Indaleko

Storage Naming Evolution

by

William Anthony Mason

S.B. Mathematics, University of Chicago, 1987

MSc. Computer Science, Georgia Institute of Technology, 2017

A THESIS PROPOSAL SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

in

The Faculty of Science
(Computer Science)

The University of British Columbia
(Vancouver)

August 2021

The following individuals certify that they have read, and recommend to the Faculty of Graduate and Postdoctoral Studies for acceptance, the thesis entitled:

Indaleko

submitted by **William Anthony Mason** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy** in **Computer Science**.

Examining Committee:

Margo I. Seltzer, Computer Science
Co-Supervisor

Ada Gavrilovska, Georgia Institute of Technology, College of Computing
Co-Supervisor

Sasha Fedorova, Electrical and Computer Engineering
Supervisory Committee Member

Norman Hutchinson, Computer Science
Supervisory Committee Member

Andrew Warfield, Computer Science
Supervisory Committee Member

TBD
TBD

Abstract

Modern computer storage systems have evolved from a simple model in which names denoted the information necessary to know both *where* a file was stored as well as *what* the file contains. As computer storage systems have evolved we have seen an increase in the diverse functional behavior offered for storing files. This has led to a situation in which users use distinct storage silos. Each of these storage silos uses its own naming scheme, provides support for specific usage, and has its own unique metadata support.

The size, type, location, and capabilities of storage silos continues to proliferate. Storage silos can be local to a computer, shared with other local computers, form geo-replicated distributed storage, consist of object storage or traditional hierarchical file organization. Non-traditional storage systems have emerged through the use of collaborative tools such as Teams, Slack, and Discord, each of which creates another location for storing information shared between human users.

These storage silos do not provide mechanisms for tracking how information is stored across these various silos. This means is a user trying to locate a document she saved while collaborating with a colleague will need to search through each of these storage silos that she routinely uses: was the document attached as an e-mail, sent via a collaboration service, stored in a shared cloud storage location will have a difficult time locating that document six months later.

This work proposes to develop a model to separate naming from location, which enables the construction of dynamic cross-silo human usable namespaces and show how that model extends the utility of computer storage to better meet the needs of human users.

something odd wtih the temp

Revision: September 17, 2021

Glossary

This glossary uses the handy `acroynym` package to automatically maintain the glossary. It uses the package's `printonlyused` option to include only those acronyms explicitly referenced in the LaTeX source. To change how the acronyms are rendered, change the `\acsfont` definition in `diss.tex`.

AFS	Andrew File System
CIFS	Common Internet File System
FTP	File Transfer Protocol
HCI	Human-Computer Interface
HDFS	Hadoop File System
HPC	High-Performance Computing
NFS	Sun's Network File System
NVMe	Non-Volatile Memory Express
RAID	Redundant Array of Inexpensive Devices
WebDAV	Web Distributed Authoring and Versioning

Chapter 1

Motivation and Problem Statement

For every particular thing to have a name is impossible. — First, it is beyond the power of human capacity to frame and retain distinct ideas of all the particular things we meet with: every bird and beast men saw; every tree and plant that affected the senses, could not find a place in the most capacious understanding. — **John Locke**, *An Essay Concerning Human Understanding* [1]

Naming is an essential human task. We use names to describe identity, relationship, and property. Thus, naming is quite broad, as it can be quite concrete — such as naming for *identity* — to abstract concepts such as *relationship*.

Computer storage focuses on *content*, which is one form of *identity* such as is found in “content addressing” where the specific identity of a file is described using a hash value computed from the content of an object. Some storage systems provide the ability to encode a limited amount of information about relationships and properties. *Relationships* in file systems are typically expressed using directories (or folders) and *context* is captured via a name.

The goal of making computer storage more flexible is not a new one. Memex was first proposed in 1945 by describing how computers might help human users by serving as “augmented memory” [illegible]. Computer storage has evolved dramatically in the 76 years since yet our storage systems have failed to progress towards, let alone realize Memex’s associative relationship model.

1.1 Thesis Statement

Separating the location in computer storage systems from the naming of objects within those storage systems enables creation of dynamic cross-silo namespaces that better meet the needs of human users.

TM ▶ This is what I had before: "To address the demands to organize, locate, and manage the rapidly growing body of data collected, stored, and organized by modern computer systems we must rethink the existing naming model of current computer storage systems and implement a richer, dynamic, and functional naming scheme capable of satisfying significantly more of those needs."◀

1.2 Naming

The file system is a commonly used type of computer storage system. The framework of modern file systems differs little from the framework used in Multics in 1965, itself based upon the familiar file cabinet metaphor described for electronic storage and data organization in the 1950s. The technological components that are used to construct file systems have changed dramatically in the ensuing decades, as storage devices have become smaller, higher density, more capable and less expensive. In 1965 the total amount of data storage was less than the capacity of a single Non-Volatile Memory Express (NVMe) storage device, yet the *framework* for how we organize, present, managed, and manipulate data has not changed. If anything, additional aspects of how we build file systems have ossified into common models: tight integration with the operating system, a presumption of block structured storage and memory page management.

The Human-Computer Interface (HCI) community has shown an abiding interest in improving the usability of computer storage systems for human users. TheHCI community has described their need for additional information about the objects within computer storage systems to provide context to the storage object. This "additional information" is not bound to a single application but rather relates to the manner in which the objects are created, accessed, and modified as well as the context in which these operations occur. This information must be provided by the underlying computer operating system and computer storage system.

1.3 Challenges

Paradigm paralysis refers to the refusal or inability to think or see outside or beyond the current framework or way of thinking or seeing or perceiving things. Paradigm paralysis is often used to indicate a general lack of cognitive flexibility and adaptability of thinking. — The Oxford Review Encyclopedia of Terms (2021).

Computer storage systems continue to evolve and change at an increasing rate. The difference between file systems, which provide the basic abstraction of unstructured data, and database systems, which provide the basic abstraction of structured data, is now filled with a growing array of *semi-structured* mechanisms including: key-value stores, object stores, document stores, no-SQL databases, data warehouses, and data lakes.

Each new mechanism invented for storing data creates a new "silo" of that specific storage system. Each new storage system comes with its own semantics and meta-data, and seldom with any explicit way of tying related information together across such silos. Storage silos are often designed with specific usage patterns in mind. For example:

Hadoop File System (HDFS) — Hadoop was created to solve the problem of large, dynamically written write-once files that are typical of machine data captured from multiple sources for additional analysis.

Google Drive — Google uses a common storage mechanism for its various services including e-mail, documents, spreadsheets, photos, and videos. It allows collaborate work, both by sharing access to the object as well as providing tools for simultaneous editing between human collaborators.

Intel DAOS — Intel's solution for large parallel High-Performance Computing (HPC) storage needs that splits meta-data and data, with meta-data stored in persistent memory and data stored on NVMe devices [1].

Qumulo — Seattle-based start-up company that provides a large data storage management product that is used in specialized big-data industries. For example, Qumulo combines both local and cloud storage to provide video post-production work in which file sizes are quite large due to

[1] https://www.intel.com/content/www/us/en/high-performance-computing/daos-high-performance-storage-brief.html

the size of high resolution video files. They claim to support secure storage of petabytes of file data across multiple tiers of storage.

TM ►*I'm not sure these examples are specific enough.*◄

Each of these storage products attempts to address specific storage needs; each new product creates a new storage silo. A human user that can limit themselves to using a single storage silo can exploit the functionality of that specific silo and simplifies the challenging task of finding disparate but related content spread across storage silos.

The proliferation of storage silos exacerbates the easy ability of humans to navigate their own storage to find a specific file. One of our own HCI researchers told me that they found their ability to find things on their own computer broke when they added multiple distinct hard disks [2] The challenge of finding a specific file or set of files is one that consistently resonates with many of those with whom I have discussed my research, both inside and outside the computer science field.

The most commonly presented model of naming files (or objects) within a storage silo is the hierarchical namespace. The hierarchical namespace confounds the storage *location* with the information *relationship.* Note that location here is not relative to the logical namespace of the storage silo, not the physical storage device.

Ordinary applications familiar with using hierarchical file systems expect related objects, of whatever type, to be in or close to the same directory. The hierarchical file system model has been highly successful for more than a half-century, though that success is from the perspective of the storage community. The HCI community has been pointing out that this model is deficient for many users.

Storage silos are not a new invention. The problem of cross-silo management, including naming is not a new problem. UNIX addressed multiple silos using "mount points". A *mount point* is a location in an hierarchical name space where another namespace is logically connected to the existing namespace. For example, UNIX mounts a new file system instance on top of an existing directory. On the Linux system where I am writing this document there are 33 distinct namespaces mounted on top of the base file system namespace — the "root" namespace that starts at "/". This model has served well as evidenced by the fact that all mainstream consumer operating systems at present (Linux, MacOS X, and Windows) support mount points.

NFS on UNIX is implemented using explicit mount points, much like media file system instances while AFS used a global namespace mechanism

[2] Private conversation with Joanna McGrenere at UBC CS-50 reception.

to connect its silos ("volumes") together so that users were given a single consistent namespace. Converting the hierarchical name of internet services has been used to create internet-based namespaces, such as using Web Distributed Authoring and Versioning (WebDAV), which converts the internet `GET/PUT` model into an hierarchical file systems namespace model. Amazon's AWS S3 service provides an object store model that is accessed using HTTP GET/PUT operations as well. This can be accessed using the hierarchical storage model using any one of the numerous "S3" FUSE file system implementations on github.com. These FUSE file system implementations of S3 exist not because they are fast or efficient but because supporting the hierarchical model understood by existing applications makes them *usable* by existing applications.

The hierarchical name space is definitely powerful: it is a specific construction of individual name space silos that are composed into a single uniform namespace. However, the benefits of this single namespace have always been limited: they tend to follow the *computer* and not the *user*. Plan 9 was one of the few systems that focused on personal name spaces, introducing key ideas such as context defined names — such as the platform architecture of the computer on which a program is currently running and *unions* in which multiple namespaces are not joined via a mount point but rather via a merge of multiple namespaces mounted at the same location. Xerox's Placeless research project was a system where documents "are organized and managed according to their properties, rather than according to their location." [3]

In the past two decades the complex multi-silo storage environment has exploded:

WebDAV — the ability to map internet web servers into the hierarchical file system;

Dropbox — maintaining a mapping between a public internet remote storage solution and a portion of the local hierarchical namespace on existing storage across multiple devices;

1.3.1 Computer Storage Naming

MIS ▶*What is this? What am I getting and putting? It seems to me that there is a high level narrative missing: Different silos us different kinds of names: HNS, key-based put/get, keyword-based search (e.g., the web), I don't even know exactly what I*

[3]https://web.archive.org/web/20020210170621/http://www.parc.xerox.com/csl/projects/placeless/

would call github, hierarchical? So the work here is figuring out what the categories of namespaces are, how they are similar, how they are different and why having many is a problem (e.g., if everything were simply explosed as mount points, multi-silos might not be a problem; we'd still have problems finding things, but not due to how names are constructed). ◄

TM ►*This is a placeholder for this information. I need to figure out where the right place to put this is.* ◄

In addition, network namespaces have proliferated, often distinct, sometimes providing an hierarchical interface (e.g., the File Transfer Protocol (FTP)) and sometimes providing a key-value interface (e.g., the HTTP GET/PUT operations). The proliferation of disjoint naming silos makes it more challenging to find related information across silos. Even with an hierarchical name space there is no simple mechanism for finding related information that is physically part of distinct silos because the relevant portion of the namespace is tied to the corresponding portion of the namespace.

Thus, the underlying location of a given data object is defined by its storage silo. The hierarchical name space obscures this somewhat but the underlying system is still a composite of the namespaces on existing storage.

Some of this is historical: the only reliable place where applications can store context is within the fully qualified path name to a file. While *some* file systems support extended attributes, while others do not. Even file systems that do support extended attributes have subtle challenges associated with them, such as differing limitations on them across file systems, the fact they are lost when moved between file systems that support extended attributes and file systems that do not, and how, unlike other file system operations, there is not even a somewhat uniform API for managing extended attributed. UNIX-like systems have the various "xattr" operations but Win32 applications on Windows have only an indirect mechanism for managing them (via backup APIs) though Windows has its own very different set of native system calls for reading, writing, and enumerating extended attributes on file systems that do support them.

Thus, the most common solution is to use the file name to embed context ("meta-data"). TM ►*I thought this was one of the insight* ◄

Using file names to embed context ("meta-data") is well-understood [3]. Yet, embedding context in file names does not solve the problem of storing dissimilar types of data in the same location, which defeats the purpose of having specialized storage. Similarly, it does not solve the problem of placing data objects in their optimal storage location while preserving their relationship.

TM ►*I'm not sure I like this example, but I'll leave it here for the time being.*◄

The need for this is increasingly clear. For example, Qumulo has created a distributed file system by constructing a cross-silo hierarchical namespace separated from actual storage location with predictive data migration to drive better support for huge data collections. Qumulo's approach attempts to address the cross-silo approach but continues to rely upon the hierarchical name space to do so.

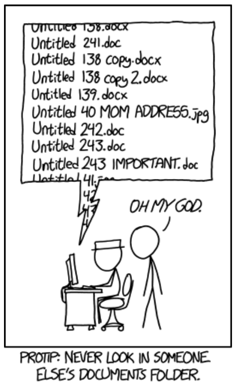

Figure 1.1: XKCD: Never Look in Someone Else's Documents Folder

Common types of file systems include:

Media — such file systems manage persistent storage, such as solid state disk ("ssd"), rotating magnetic media ("hard disk"), magnetic tape, and rotating optical storage (such as a CD-ROM). Media file systems manage the data storage provided by the media device. A media device can be constructed by combining portions of multiple media devices, such as is done with Redundant Array of Inexpensive Devices (RAID) storage. Media file systems also manage the correspondence between the file system's logical storage elements, which are typically files and directories, and

Network — such file systems utilize a network protocol for transferring data from one or more computers accessed across a network so that it can be consumed by local applications. Examples of network file systems include Sun's Network File System (NFS), Andrew File System (AFS), and Common Internet File System (CIFS). Network file systems manage moving data between computer systems over the network, presenting a compatible name space, and manage data caches and coherence between local caches and remote storage. Sometimes these are also referred to as *distributed* file systems.

Pseudo — such file systems provide structured information extracted from non-storage locations and present it via an hierarchical name space. For example, the *proc* file system (TM ►*From Plan 9?*◄) presents information about the running operating system. A pseudo file system manages its namespace as well as retrieving or storing data that relates to the corresponding system state.

MIS ►*This feels like a classic related work paragraph, but I don't know where it fits in your story line.* ◄ Media file systems have long been organized using a simple internal key-value store. The BSD UNIX Fast File System used an "index node" (inode) as a description of a data object, such as a file or directory [4]. These nodes were indexed using an identifier, which acts as a simple key. The NTFS file system is structured similarly [5]. Recent work explored the idea of separating the namespace from storage with an emphasis on high performance solid-state storage (SSD) devices [6]. This echos earlier work suggesting using object storage devices (OSD) for file systems implementations [7].

Beyond separating naming from storage, there is also a trend to separate meta-data from storage as well. One early example of this is Lustre [8], with more recent work including Ceph and Gluster [9], [10]. Each of these distributed file systems separates the service of data from meta-data, which allows data paths to focus on performance, including the use of parallel data paths. Meta-data does not benefit from high-performance parallel I/O paths and instead benefits from low latency random access. This separation does not improve naming support because that is not the goal of any of these systems.

Intel's DAOS system uses Intel DC Persistent Memory for meta-data and NVMe for data. DAOS supports two naming models: one is a POSIX-compatible hierarchical name space, and the other is a key-value store interface. Storage pools are accessed using one or the other of these interfaces but not both.

In a hierarchical name space the fully qualified path corresponds to the *logical* location of the file, that is the location of this file in the namespace itself. In most file systems that logical location corresponds to a specific file system instance and thus defines the storage device(s) on which that file is stored. Meta-data associated with that file is interpreted by the file system to determine how to retrieve the data. For a media file system that usually corresponds to the address of the location(s) of the file data on the corresponding media. For a network file system that usually corresponds to the network address used to request the correct file data from the computer storage system where the data is actually stored.

Separating the file system meta-data from the file data is known to be beneficial. [11] This approach logically makes sense when considered in the multi-silo context as well, since a human user looking for something actually does not care *where* that the thing is located they care *what* the thing is.

This raises an intriguing question, one that is at the heart of my own research: *what is the purpose of naming?* Applications are quite happy to use names that are meaningless to humans, such as content hashes, UUIDs, or randomly generated string names, such as are used for temporary application files. Indeed, if an application knows that a name has a fixed format, the knowledge of that fixed format can be used to simplify the implementation or to separate application named files from other files.

Humans use file names to provide *meaning* as to what is in the named object. Librarians assisting people in organizing data routinely suggest embedding contextual information in the file names; directories are then used to create additional organizational structure (or "context"). There are numerous limitations to this approach, however, because humans do not organize things in the same fashion and even the same human does not organize things the same over time. This is such a common phenomenon even the popular press pokes fun at it (Figure 1.1.)

The inability to find specific things is *not* unique to computer data storage. Indeed the current situation for organizing digital data seems to be descended from the organizational structure of a library or a file cabinet, despite the physical limitations of organizing paper files that are not constraints for computer storage. For example, when I am organizing my long play record collection I have to chose one primary "key" to use, such as the time when the album was released. However, digital storage does not have this same restriction - I can easily ask that they be presented to me by genre, artist, album title, or even instrument type. Further, I can change my organizational scheme dynamically. Both Bell Lab's Plan 9 [12] and Xerox Placeless Documents [13] observed that there is a dynamic nature to how information is organized. The hierarchical name space is basically static, like my album collection is static: I can choose a different organizational scheme but to do so I have to physically move them around.

Indeed, the system we have evolved works *in spite* of the obvious artificial limitations imposed by decades old decisions [14]. Rather than being a deep insight that the current system works because it is the best of all possible models, it is a testament to human ingenuity and a tolerance for primitive, sub-standard systems.

There are a body of recommendations about file naming standards[4] [5] [6] [7]. Harvard Data Management suggests [8]:

- Think about your files
- Identify metadata (e.g., date, sample, experiment)
- Abbreviate or encode metadata
- Use versioning
- Think about how you will search for your files
- Deliberately separate metadata elements
- Write down your naming conventions

Thus, there are common themes: a name provides context for *what* the given thing represents. Uniformity of information is also important — the "naming convention" permits not only identifying similarity but key elements of *difference* between any two named things. Thus, to return to the original question: "what is the purpose of naming?"

TM ▶*It seems that this Harvard list is a serious indictment of the existing system: it pushes the cognitive load onto the users, talks about meta-data , versioning, encoding,* and *capturing the naming convention.*◀

It seems clear that human naming is not the same as data storage naming, though we often treat them as the same. This can be seen in the data storage tendency to either use names that are entirely about locating a specific object, such as is the case with a key-value store, for example, or it combines location with identity, such as is typified by the Uniform Resource Identifier (URI) [16].

Thus, it would seem that naming is about:

Identity — the name of a thing should be sufficient to verify that it is the specific thing we seek. An example of this is biometric data for humans or a cryptographic hash for a storage object;

[4] Data Management for Researchers [15]

[5] Smithsonian: https://library.si.edu/sites/default/files/tutorial/pdf/filenamingorganizing20180227.pdf

[6] Stanford: https://library.stanford.edu/research/data-management-services/data-best-practices/best-practices-file-naming

[7] NIST: https://www.nist.gov/system/files/documents/pml/wmd/labmetrology/ElectronicFileOrganizationTips-2016-03.pdf

[8] https://datamanagement.hms.harvard.edu/collect/file-naming-conventions

Location — the name of a thing could provide information that directly or indirectly specifies where it is located. A human example is how an identity card might specify a current residential address. A data storage example is the U.S. Library of Congress Classification System, which can be used to find a particular work within very large bodies of work even though books are typically not thought of as a storage media;

Relationships — the concept that one thing can be derived from another, whether in a direct form, such as a version of the same logical thing, or in a more indirect form, such as a provenance relationship showing that one thing was derived from another thing by applying some set of transformations to it;

Characteristics — this relates to something that is a property of the thing. For humans this might be the date of birth ("creation date"), height, weight, eye color, etc. For storage objects this might include timestamps, size, data type;

Context — at some some level, we often rely upon names to provide us with *context*. For humans, we assume that people with the same family name are likely to be members of the same family, even though "Aki Smith" is distinguished from "Dagon Smith".

When considered from this perspective, it seems clear why "naming is hard" — it plays multiple distinct roles, sometimes overlapping, sometimes interfering. These challenges are not well-served by existing naming support in computer storage systems.

TM ▶*I feel that I should capture the dynamic nature of naming, which really does differ from the traditional static model of it.* ◀

1.4 Contributions

[I don't let the cleaners in b]ecause ***I*** *know where everything in this room is. All the books, the papers — and the moment they start cleaning, those things get hopelessly organized and tucked away and I can never find them again.* — Crown of Midnight, Sarah J. Maas

This thesis proposal describes what I intend to contribute to our understanding of how to evolve naming in computer storage:

1. An explicit model for what computer storage naming is: how and why we name data objects, what a name *should be* in a computer storage context to correspond how names are used and formed; and

2. An architecture and design of a potential naming system that supports the naming model (item 1 that is sufficiently flexible to work across the full range of current and prospective computer storage needs: in-memory compute systems, small devices, computer workstations, enterprise scale storage systems, and distributed, geo-replicated cloud storage systems, which permits the construction of new purpose-built storage systems with the strong naming support as well as integration of existing storage systems; and

3. Implement and evaluate a novel storage system implementation based on this design (item 2) that demonstrate strong naming support within a single storage silo model; and

4. Implement and evaluate an implementation of the richer naming system (item 2) on an existing computer storage system; and

5. Implement and evaluate the combination of both the novel storage system implementation (item 3) and the enhanced legacy system (item 4) into a unified system consistent with the model (item 2).

Chapter 2

Background

TM ▶*This should come from the existing text. It should explain how we got to this point.* ◀

Chapter 3

Model

In Section 1.3 I attempted to capture key aspects of how naming is used by human users and from that proposed a basic list of key elements to consider in a comprehensive naming model: identity, location, relationship, characteristic, and context. While this is a good place to begin my analysis, more is needed to construct a robust model. For example, one of the challenges of naming is that it is *dynamic*: the storage location of a given data object might change. This often means that the *name* also changes, because the name encodes location. This is no more intuitive to a human than it would be to insist someone change their name each time they moved to a new home.

In other words, the *object* is not different even though its storage location changes. Neither users nor applications care about this specific detail until they go to actually retrieve it. In the current model, rather than bein forwarded to the correct object an error is returned indicating that the object is no longer accessible.

To facilitate developing my naming model, I rely upon several use-cases that have arisen during the course of my research around this topic, both working with collaborators as well as discussions for ordinary users that are unfamiliar with my research area. **TM** ▶*Margo suggested that I be specific here by pointing to particular people in footnotes for the time being. I defer that for the moment but it is a good thing to add as I continue editing.*◀

3.1 Use Cases

To motivate the model I propose for *Indaleko*, I first start with a series of potential use cases. **TM** ▶*Note that I've started with the two from the original*

Feature	Existing Technologies	No Solution
ACTIVITY CONTEXT	timestamps and geo-location, image recognition, browsing history, ticketing systems, application-specific solutions like Burrito [3].	Link related activity across apps, record browsing history and chat conversations relevant to the creation of the data object, storing it in ways that are secure and compact.
CROSS-SILO SEARCH	Search by name, creator, content across silos, app-specific searches (e.g., Spotlight)	Unified search across all kinds of storage, including file systems, object stores, apps and devices
DATA RELATIONSHIPS	De-duplication of documents, versioning of specific files, git ancestor relation	Explicit notion of data identity, tracking different versions across different silos as data is transformed
NOTIFICATIONS	File watchers (INotify), synchronization status, manually inspecting modified time	Ability to subscribe to specific changes on attributes
PERSONALIZED NAMESPACE	Hierarchy plus hard/soft links. Use of tags.	Creating personalized namespaces with with flexible data organization and views

Table 3.1: Use-case driven functional requirements.

Indaleko *paper, but I think it might be worthwhile to add one or two more to provide a more well-rounded model.*◀

Data Processing

Compliance

Memex

Asset Management

Using these uses cases as motivation, I propose that *Indaleko* support the features as shown in Table 3.1 in greater detail in Section 3.2.

3.1.1 Data Processing

TM ▶*This is from the HotStorage paper submission*◀

Aki and Fenix are preparing a report summarizing their work on a data analysis project for a customer. Fenix sends an email to Aki containing a CSV file with original data. Aki opens this document in Excel, formats and filters it, adds additional data from a corporate storage silo, and then returns the Excel document to Fenix on Slack. Fenix is away from their desk when it arrives, so they open it on their phone, uploading it to a cloud drive. Fenix then sends the link to Aki for editing with update notifications. Finally, Fenix sends a PDF of the report to the compliance officer who promptly asks, “Where did this data come from?” Aki and Fenix are preparing a report summarizing their work on a data analysis project for a customer. Fenix sends an email to Aki containing a CSV file with original data. Aki opens this document in Excel, formats and filters it, adds additional data from a corporate storage silo, and then returns the Excel document to Fenix on Slack. Fenix is away from their desk when it arrives, so they open it on their phone, uploading it to a cloud drive. Fenix then sends the link to Aki for editing with update notifications. Finally, Fenix sends a PDF of the report to the compliance officer who promptly asks, “Where did this data come from?”

3.1.2 Compliance

Delete Request: Some time later, the compliance officer requests that all documents containing a customer’s data must be deleted. To help with finding all relevant customer data, Dagon joins the project and examines the report and requests the original data from which it was produced. Aki remembers that they gave the original data to Fenix shortly after collecting it, but does not remember the name, location, or even how the relevant files were transmitted. Thus, Aki has to manually search possible locations and applications, sendsing references to documents to Dagon, who then starts organizing these files to methodically identify the ones that might contain the customer’s data. In the process, many of the other team members’ references to the documents stop working.

TM ▶ *Discussion with Ada: how do people do this already? Why are those solutions insufficient? Some sites are accessible and others are not. How do they prove they are GDPR compliant? What about when people move from dynamic to static memory? Could I use storing the hash value as a mechanism for motivating this because it facilitates finding things. Much like the Apple content hash for child p0rn.*

How about extracting text from pictures to avoid censorship? ◀

3.1.3 Memex

The "memex" is a device posited by Vannevar Bush in 1945 [2]:

> *"Consider a future device for individual use, which is a sort of mechanized private file and library. It needs a name, and, to coin one at random, "memex" will do. A memex is a device in which an individual stores all his books, records, and communications, and which is mechanized so that it may be consulted with exceeding speed and flexibility. It is an enlarged intimate supplement to his memory."*

While the world wide web is certainly one interpretation of his forward thinking article, the system he describes is also highly personal and appears to focus not only on finding things but also capturing the context in which they are found.

TM ▶*Seems like the key here is to extract the salient factors that would impact this model.* ◀

Thus, when considering my naming model, I can draw upon the needs of Memex to assist in ensuring the model is sufficient to meet the issues raised by Bush [2]:

Selection — "The prime action of use is selection, and here we are halting indeed." The key here is being able to *filter* items of interest at a given time. This becomes important as the number of objects being considered grows. While computers are somewhat faster than they were in 1945, he points out the general problem of scaling and the need to be able to limit the actual size of the search space. This is a very real consideration for our naming system: scalability. Brute force search is not sufficient, as that is what we have *now* and even as fast as storage systems are today this is not tenable.

Association — "Our ineptitude in getting at the record is largely caused by the artificiality of systems of indexing. When data of any sort are placed in storage, they are filed alphabetically or numerically, and information is found (when it is) by tracing it down from subclass to subclass. It can be in only one place, unless duplicates are used; one has to have rules as to which path will locate it, and the rules are cumbersome. Having found one item, moreover, one has to emerge from the system and re-enter on a new path.

"The human mind does not work that way. It operates by association. With one item in its grasp, it snaps instantly to the next that is suggested by the association of thoughts, in accordance with some intricate web of trails carried by the cells of the brain. It has other characteristics, of course; trails that are not frequently followed are prone to fade, items are not fully permanent, memory is transitory. Yet the speed of action, the intricacy of trails, the detail of mental pictures, is awe-inspiring beyond all else in nature."

Thus, the key take-away here is to find associations. This is part of the motivation for *activity context*.

Trails — "And his trails do not fade. Several years later, his talk with a friend turns to the queer ways in which a people resist innovations, even of vital interest. He has an example, in the fact that the outraged Europeans still failed to adopt the Turkish bow. In fact he has a trail on it. A touch brings up the code book. Tapping a few keys projects the head of the trail."

Thus, the key take-away here is to be able to show at least one kind of relationship: a "trail," which seems to be similar to provenance.

3.1.4 Asset Management

One recurring theme in my conversations with the creative community has revolved around the management of *assets*. This term is broadly used: web pages are constructed of assets, video sequences consist of a unique rendering of independently combined asets, computer games include audio and video assets. There is no single hierarchical organizational structure for assets that satisfies the needs of this type of creative endeavor: a single game asset could be classified by a myriad of characteristics. When a game developer is looking for a particular asset, they are often focused on those characteristics. When an audio engineer is attempting to construct specific sound effects they could be looking for: the duration, number of channels, channel mapping, sampling frequency, bit depth, instrument, or dynamic range for example. It is clear that what doesn't work in such a situation is a single directory filled with all of the assets: that isn't useful.

Indeed, this use case seems to focus on being able to identify and use the *properties* of a given data object. Simpler examples of this might be how one organizes documents for accounting purposes: bank statements could be sorted by the bank from which they came or the month that they cover. In my experience, accounting users actually will store copies of such a file because they need to be able to identify it in *both* formats. Another similar example is "how do you organize your music collection?" A music collection could be organized by artist, or album, or year of release, or publisher or lyricist, or genre. When we have a physical copy of recorded music (e.g., an eight-track tape [1] we are limited to the ways in which we can organize it. Digital files have no such limitations; that limitation is an artifact of the current naming system.

TM ▶*What requirements are imposed by this use case? It's relatable, but does it really impact the design?* ◀

3.2 Features

Indaleko must provide the following features to meet the needs of the use cases:

Activity Context — this is a mechanism by which we capture information for understanding the context in which data is created, transformed, and accessed. This is not application or storage silo specific information. Examples of this might include timestamps, application specific meta-data, history information, provenance information, etc.

Search — while search is not the only use we envision for *Indaleko*, it is one of the mechanisms that we expect to enable and should support searching across storage silos and exploiting the richer meta-data

3.3 Architecture

TM ▶*I'm starting with the* Kwishut *architecture from HotStorage 2021.*◀

Indaleko is a family of services that enable sophisticated search and naming capabilities. The key features that differentiate Indaleko from prior work are:

1) incorporating object relationships as first class meta-data because prior work has not done so which interfers with the ability to ensure meta-data and naming services work together; and

[1] Invented in 1964 and thus as old as Multics, which adopted the hierarchical file system structure.

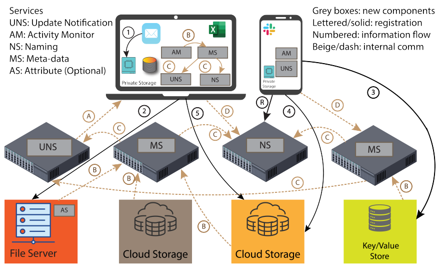

Figure 3.1: Indaleko Architecture (see Section 3.3.1).
Grey boxes indicate new components. AM="Activity Monitor", MS="Meta-data Server", UNS="Update Notification Server", NS="Namespace Server".

2) federating meta-data services, which is necessary to ensure efficiency, scalability, and security; and

3) recording activity context, which is necessary to provide insight into how the data is used dynamically (over its lifetime) rather than statically (at the point of creation or last update); and

4) integrating storage from multiple silos, which is necessary to clearly distinguish the storage characteristics that are focused on storage service optimization from the usage characteristics that establish associative context; and

5) enabling customizable naming services, which are needed because we need the flexibility to support a broad range of existing as well as innovative new storage services.

Data continues to reside in existing and to-be-developed storage silos. Indaleko interacts with these silos, collects and captures metadata, and provides a federated network of metadata and naming services to meet the needs of users with the use cases in §3.1 being our initial evaluation of our architecture and design.

3.3.1 Indaleko Services

Figure 3.1 illustrates the Indaleko architecture. Indaleko allows for different deployment scenarios. The services can be run independently, they can be co-located and bundled together to run on a local device, integrated into an OS, or available as network-based services.

In the discussion below, parenthesized numbers and letters refer to the arrows in Figure 3.1. There are five main components:

1) **Metadata servers (MS)** are responsible for storing attributes and provide a superset of capabilities found in existing metadata services [17], [18]. Users can register an MS with activity monitors or attribute services, which allows the MS to receive updated attributes from storage objects and activities (B). Thus, there can be multiple sources of attributes including the user itself. Metadata servers may retain the full or partial history of attribute updates or maintain only the most recent value.

2) **Namespace servers (NS)** connect to one or more MS and use the metadata to provide users with a personalized namespace that allows both manual organization (i.e., a hierarchical namespace) and rich search capabilities. Users can register with an NS (R) that uses one or more MS to obtain relevant attributes from them (C). Additionally, users can be part of a corporate NS that allows sharing of their select metadata with other users via standard enterprise public-key cryptography.

3) **Activity monitors (AM)** run on the user's devices. Their main function is to observe temporal relations, activity context, and relationships between objects on a user's device and transmit them to an MS (D).

4) **Attribute services (AS)** extract attributes from storage objects and transmit them to an MS (B). An AS might be invoked on updates, run once or periodically. For example, a file system AS would update the object's metadata with basic attributes such as size or modification time. There can be many AS that extract more "interesting" attributes, e.g., image recognition, similarity, or other classifiers.

5) **Update notification server (UNS)** provides notification mechanisms. Users can register interest in changes of attributes or underlying storage and will receive a message on change events (A) to which they have access.

3.3.2 Indaleko working example

To make the Indaleko architecture concrete, we revisit our use-cases from §3.1 and walk through parts of it to illustrate how Indaleko supports the various actions and events.
Storing the e-mail attachment. Aki's act of saving the CSV file that Fenix sent in email corresponds to the creation of a new object on the file

server silo, i.e., the file system (4). The file server is Indaleko-aware, so the AS co-located with it extracts attributes from the document and forwards them to the MS (B).

The AM on Aki's laptop detects that the CSV file came via company email from Fenix. It then captures the activity context identifying the relationship between the e-mail and the CSV file and transmits it as additional metadata about the CSV file to the MS (that already contains metadata extracted by the AS). Moreover, because there is a company-wide namespace service, Indaleko establishes that the e-mail attachment, the CSV in the file server, and the one on Fenix's laptop (from which the file was sent) are exact copies of each other.

Many applications already record some form of activity context, e.g., chat history, browsing history. Such histories provide a rich source of additional metadata. Other activity context, specifically the relationship between objects, such as the fact that a particular file was saved to a local storage device from an email message, requires more pervasive monitoring as found in, e.g., whole provenance capture systems [19]. Indaleko is agnostic about the precise data that comprises activity context, but allows for storing and accessing activity context as metadata.

Creating the Excel file. Next, Aki opens the CSV file using Excel and stores it as a spread sheet. This creates a new object. The AM detects that the newly created spreadsheet is a conversion from the CSV file, either via a notification from Indaleko-aware Excel or by monitoring the system calls executed on the local system. Aki proceeds to modify the data by filtering it in Excel and saving the changes. The AM records this event and updates the meta-data of the spreadsheet to record the derivation-relationship. Ideally a Indaleko-aware version of Excel specifies to the AM the exact type of the relationship (in this case a derivation); otherwise the AM informs the MS about an unspecified data relationship by observing the opening of a CSV file and a subsequent creation of the Excel file.

Sharing the spreadsheet. As Fenix receives the Excel file from Aki via Slack on their phone, a sequence of metadata events similar to those described earlier takes place, except the phone does not run a local NS or MS. Fenix now uploads the file to the company's cloud drive (4). The MS (by way of AS) reflects the creation of a new object and records its remote location. The use of a company-wide namespace and metadata service enables Indaleko to record that the file in the cloud drive is, in fact, a copy of the one received via Slack. Further, Fenix informs their personal NS that they wish to notify Aki about all updates to the file on the cloud drive. Thus, whenever an AS sends updated attributes to the MS, Fenix receives a notification.

Data origin and delete requests. When the compliance officer asks about the origin of the data, Fenix can query the corporate NS to obtain the complete history of the report. This includes the spreadsheet from which the report was derived and the e-mail or Slack messages that transmitted the files. The corporate NS was configured to be aware of the locations of the collaborating users' personal NS. Moreover, because of the activity contexts captured by the AM, Indaleko is able to identify documents that were created during any activity involving the customer whose data must be deleted. Starting from these documents, and by using the relationship of documents, Dagon was able to find all relevant objects and delete them, including the e-mail and Slack messages.

Note that unlike existing systems, Indaleko is able to efficiently find related objects across storage silos. Operating systems already provide users with indexing services to accelerate search of local files. This search can be made cross-silo by mounting and enabling indexing on network shares (e.g., Windows Desktop Search), or by interfacing with specific applications such as e-mail (e.g., MacOS Spotlight, or Android search). The problems with indexing on a large network storage repository are resource limitations such as bandwidth and local storage that may render the system unusable during indexing. In contrast, Indaleko addresses these limitations by delegating indexing and storage to one or more services.

NS are responsible for providing efficient search functionality. Indaleko uses AS to keep attributes up to date with object modifications. Lastly, Indaleko supports coordinated search among one or more local and remote NS, allowing, for example, a user to search across both their local NS as well as their employer's NS.

TM ▶*Would it be useful to resurrect some of the excluded text from this section?*◀

Chapter 4

Plan

The contributions that I proposed in Section 1.4 are substantial. To achieve this I must implement a number of novel new approaches to naming in computer storage systems. In this chapter I provide a high level overview of my plan to accomplish this.

4.1 Research Questions

This plan is focused on answering the following research questions:

Can a storage silo implementation that separates naming, meta-data, and storage manag
It is easy to talk about *functionality* but there is a real risk with a radically new architecture that it will not be usable due to the performance costs. Part of the challenge in answering this research question is to pick appropriate benchmarks. My goal is to demonstrate there is *no* significant cost associated with I/O performance (less than 5%), and that there is at most modest cost associated with meta-data performance (creating, opening, deleting, renaming) versus existing systems (less than 10%). While I think it would be useful to work with someone interested in data visualization to demonstrate improvements in data analysis and access, I do not propose doing this as part of my own work. I discuss this in more detail in Section 4.2.

Can a federated meta-data service be constructed to provide meaningful security guaran
By increasing the exposure of information via enhanced/augmented meta-data there are clear benefits for compliance measures, but these are achieved at the expense of making additional meta-data available. Part of the original design was to address these concerns and this

research question evaluates the efficacy of that design. To do this will require examining the security model my system uses, the potential information exposure, and at least a first-order analysis of the impact of information leaks. Depending upon my analysis of the answer to this question it may be necessary to augment the *Indaleko* security model to better protect the information.

How can *Indaleko*, implemented as a cross-silo naming and meta-data service, provide m
The research question itself is vague at this point because while I can suggest anticipated benefits in terms of cross-silo data location it is difficult to identify the "benefits" let alone quantify them. Despite this, both will be necessary, which makes this a "high risk" research question. **TM** ►*I don't think this is a good research question as written, so I need to re-work this...again. The difficulty here is that measuring this is not going to be particularly easy, unless we turn this into an HCI activity, such as by looking at "abandonment rates" when people start searching for things. That* may *be what is required, though, to make this argument because a performance eval of the software doesn't make much sense — to what would I compare this?*◄

At present, I only have a basic architecture and high level design of the system. While developing all of the pieces required to realize the design would take considerable effort, I can construct a prototype implementation using existing technology for many of the components. In the following sections, I have attempted to identify potential existing technologies that can be used. My goal in identifying existing technologies that I can use is sufficient to validate my goal of prototype construction while leaving sufficient flexibility to replace components as it proves necessary during the implementation phase of my research.

One key area in which I expect to develop new techniques is meta-data extraction and construction from existing data and storing that meta-data into pre-existing technology components. I describe this further in the following sections.

4.2 Storage Silo Implementation

While my objective is to construct *Indaleko*, which will support multiple silos combined with federated naming, activity, and meta-data services, my proposed initial step is to do this on a single system consisting of two storage silos.

This section constitutes an initial "straw person" proposal for what I anticipate constructing. In doing so I expect to evaluate at least two of the following research questions:

- **Can the separation of naming from the file system(s) be achieved without compromising performance?** This is related to the broader question articulated earlier in this chapter (Section 4.1.)

- **Can this two-silo naming system be used to demonstrate more effective data location?** The original motivation is that rich, dynamic naming will enable “finding my stuff”. One approach to this would be to build on top of prior work done in our department related to *data curation* to see if a richer naming system can be used to simplify that task [20].

- **Can a non-traditional storage mechanism be effective when used as a storage silo within this separated naming and meta-data service model?** This specifically speaks to the non-hierarchical name system idea that was previously proposed in “Hierarchical File Systems Are Dead” [7]. While not a primary motivator for the main thesis, I am intrigued by the idea of taking a persistent memory key-value store along the lines of “Modernizing File System through In-Storage Indexing” [6] where the key-value store is in persistent memory, which allows me to leverage my own prior work.

- **Does the explicit separation of meta-data and naming from the file system enable the creation of dynamic per-user name-spaces?** Traditional namespaces are static in that the names of objects within them are not changed by the system, yet relationship based namespaces could be dynamic. Further, the namespace of interest to me now may not be the same as the namespace of interest to someone using the same data but in the context of a different role — by not tying the namespace to the data does it provide a more functional namespace? **TM** ▶*This question is not fleshed out enough at this point.*◀

- **Does capturing activity context related to the way that data is constructed and used allow construction of enhanced data management mechanisms?** One obvious way of demonstrating this might be to show where it does permit that type of enhanced data management mechanism. For example, this might be related to the data curation aspect of prior work. It is likely there are other similar data management activities that might yield a useful basis for finding information of interest.

I anticipate that this work will require developing specific components, which largely correspond to the services described in Section 3.3.1.

One key aspect that is not well-defined in the model section but is important to do as part of this work is to define what an *activity context* is, how it is created, and how it is used within the system. This will be an important aspect of this work, even if the actual implementation is quite simple, possibly capturing only a tiny amount of information. Initial techniques for doing this can likely be based upon existing information gathering systems, such as *eBPF* TM ▶*add to glossary, define out*◀ in Linux or *ETW* TM ▶*add to glossary, define out*◀ on Windows, both of which are pre-existing systems with well-defined mechanisms for extracting information from production systems.

To review, the services that will be implemented in this phase of my work are:

Metadata Servers — there are existing models for meta-data servers, such as Egeria [1] or Amundsen [2] (a demonstrative, but not definitive list) that might be sufficient. If not, using a key-value store such as WiredTiger [3] to construct a metadata server is also viable. TM ▶*I'd rather not build anything I don't have to but my concern is that this is going to be a core bit of the work; the right thing to do is to do a more thorough search for these technologies and figure out which one looks like a good fit.* Building *one is possible but seems like a big project on its own.*◀

Namespace Servers — my expectation is that this will need to be constructed; it must interoperate with the metadata servers. Work here will include defining the interface to constructing a new namespace or retrieving a previously constructed namespace. While the ultimate goal is to have a federated namespace service, this initial effort need only provide a single, non-federated namespace even though that is more limited than the actual design.

Activity Monitors — these create the activity context information I described previously. I would expect that only a single activity monitor will be built for this initial system prototype and the details of this will be driven by the chosen definition of the activity context and relevant activity providers.

[1] https://github.com/odpi/egeria
[2] https://github.com/amundsen-io/amundsen
[3] https://github.com/wiredtiger/wiredtiger

Attribute Services — because *Indaleko* is a naming system, we need a mechanism that retrieves storage attributes. Initially, this will consist of a source for scanning current content and then a monitoring mechanism (likely similar to the activity provider) that notifies the attribute service to update the attributes of files that have changed.

Update Notification Server — existing systems provide change notifications. In a cross-silo system this sort of dynamic state monitoring will also be useful. An initial implementation for a local system could consist simply of using the existing change notification mechanism(s) for watching changes. A more robust implementation could use information from the meta-data service(s) to determine if a given change is of interest. This dynamic change tracking could then be federated as part of the later planned work.

This proposal is quite broad and consists of no working software at the present time. To achieve this I propose taking the initial architecture and constructing a design based upon this architecture. To evaluate the design, I propose choosing at least two distinct storage silos and at least one target operating system. To the extent possible, initial implementation would focus on combining existing components as much as possible. For example, Using MinIo and Sparkle Share as storage silos, with Windows Cloud Sync would provide documented and generally well-understood technologies on which to construct these services. The Meta-data server can be constructed using one of the available key-value stores (e.g., WiredTiger). The Notification service could start with Emitter. Initial activity context work can be built using eBPF or other inbuilt tracing mechanisms. This leaves how to build the attribute services as an open area to be further refined. TM ▶*TBD*◀

4.3 Federated Naming/Meta-data Security

TM ▶*TBD*◀

4.4 Cross-Silo Naming/Meta-data Benefits

TM ▶*TBD*◀

Chapter 5

Timeline

The purpose of the schedule shown in Table 5.1 is subject to adjustment as the research progresses.

Table 5.1: Proposed Schedule for competion of Doctoral Thesis

Month	**Activity**
2021	
September	Draft thesis proposal to supervisors
October	Updated draft thesis proposal to supervisors
November	Final thesis proposal to committee
December	Defend thesis proposal - reach candidacy
2022	
January	Begin proposed research Part 1 (Storage Silo)
April	Milestone 1 Storage Silo (Alpha)
June	Milestone 2: Storage Silo (Beta)
August	Milestone 3: Storage Silo (Complete)
September	Storage Silo paper submission ready
September	Begin proposed research Part 2 (Federated Naming/Security)
December	Milestone 1: Federated Naming/Security (Alpha)
2023	
March	Milestone 2: Federated Naming/Security (Beta)
June	Milestone 3: Federated Naming/Security (Complete)
July	Federated Naming/Security Paper submission ready.
September	Milestone 1: Cross-Silo Naming (Alpha)
December	Milestone 2: Cross-Silo Naming (Beta)
2024	
March	Milestone 3: Cross-Silo Naming (Complete)
April	Cross-silo Naming Paper submission ready
June	Draft thesis.
August	Final thesis.
October	Defend thesis.

Chapter 6

Related Work

Ways to organize data within a storage silo have been extensively studied from multiple perspectives. That there are so many different approaches to evaluating the optimal way to organize things is a testament to both the importance and complexity involved in approaching this topic. This section will consider the following related work:

- The *storage* perspective, which is primarily rooted in the systems perspective. Storage in this context includes file systems and databases, each of which then has multiple different manifestations.

- The *provenance* perspective, which is related to systems but focuses on creating a context of explainability that is distinct from storage.

- The *human-computer interface* (HCI) perspective, which is related to how humans organize and find information within storage systems. There are a number of distinct perspectives to this work including: hieararchical data organization, personal information management, enhanced search, and cognitive data organization.

Each of these perspectives is complementary and assist in better understanding the problem: systems focuses on being able to efficiently and reliably store and recover information though often it is agnostic to the specifics of the information; provenance focuses on accountability and explainability; and HCI focuses on the human facing problems that need to be solved and tends to ignore how those solutions are implemented.

[3], [12]–[14], [17]–[19], [21]–[47]

6.1 Storage

Much of storage work is dominated by the realities of how media and networks function, with a goal towards increasing both capacity and performance. The basic model of data organization within storage systems tends to separate into *structured* data — typically what is found in databases — and *unstrutured* data — typically what is found in file systems or object stores.

TM ▶*Describe structured data: why do people use databases, what are the strengths and weaknesses. How does this relate to data organiztion?* ◀

TM ▶*Describe unstructured data: why do people use file systems or object stores, what are the strengths and weaknesses? How does this relate to data organization?* ◀

TM ▶*Describe the work that has been done in semantic file systems, metadata augmentation, and non-hierarchical organization structure (e.g., Ground and Placeless).* ◀

TM ▶*This is a fairly large quotation from the Placeless Documents Project Archive that should probably be edited down, but for the moment I want to leave it here because I think the insight provides is quite useful.* ◀

From the Placeless Documents Project Archive [1]:

> Placeless Documents are documents that are organized and managed according to their properties, rather than according to their location. Document properties can be things you already know about your documents, like that they're published, or notes, or about the budget,or drafts, or source code, or important, or shared with your colleagues, or from your manager, or big, or from the Web, or... whatever suits you. Document properties can also be things that you want to be true about your documents, like that they are backed up, or replicated on your laptop, or can be purchased for a small fee. These latter properties carry the code to implement or interface with the desired functionality. Document properties are statements about your documents that make sense to you, and affect what you're going to do with the documents.
>
> What does it mean?
>
> We live and work in an information-filled world. The project focuses on helping people cope with the large and diverse information spaces that are part of life in the networked world. Our approach is to provide information consumers with a unifying model for organizing and manipulating their information.

[1] https://web.archive.org/web/20020210170621/http://www.parc.xerox.com/csl/projects/placeless/

Much of the information that we commonly use is in the form of documents, both physical and electronic. In fact, the use of electronic documents on our computer desktops is pervasive, and even though they unify much of desktop computing, the document metaphor falls short of providing an accessible and readily understood way to interact with all forms of information, whether electronic or physical. Instead, we resort to specialized applications for much of our computer-based work. Electronic documents are managed through different systems like mail, WWW, and file systems. Similarly, the many devices we use in the course of our work (including fax machines, scanners, televisions, VCRs, telephones) manipulate and store information, yet they do not integrate seamlessly with the rest of our on-line information.

Furthermore, the means available for individuals or groups to organize and customize their information spaces are extremely poor and driven mostly by storage and distribution models, not user needs. The most common model for information organization is hierarchical. The use of folders as a fundamental organizing principle, and the restriction that documents (mail messages, files, URLs, etc.) appear in only one folder at a time, force users to create strict categorizations, resulting in inflexible organizations that tend to persist over time even though their needs evolve. Similarly, customization of the organization and the behavior of information according to individual requirements is cumbersome, if not impossible. Access to a shared document that one individual deems as corresponding to the budget and another project cannot be easily tailored for both specialized functional requirements are equally hard to achieve. If a user can express that a document is read-only, and that it is a Word document, why cant heexpress that updated copies should be faxed to a colleague once a week?

This project is about removing these hurdles by using a novel infrastructure and proving its benefits through applications that exploit its capabilities. We plan not only to change the way people interact with their currently segregated world of documents, but we plan to exploit a single concept the document and its properties to allow users to interact with arbitrary information.

Our vision is one of customizable, context-aware management of integrated information spaces, which:

- integrate information components from many sources: repositories

(WWW, mail, file systems), devices (scanners, video-cameras, television, phone), and dynamic processes (workflow, source code management systems, search engines, and dynamic document content),

- allow customizable organization of the information based on properties of that information, e.g., budget related, read at home, shared with John, and From: petersen@parc.xerox.com,
- allow information properties to be arbitrary objects specified through many different mechanisms: explicitly by the users themselves, captured by physical context sensors, inferred from usage, automatically generated by content analysis, etc.
- allow information properties to be active and carry behaviors to automate information work, enabling functionality like fax to John at 5pm each day, translate to English, notarized, backed-up in Utah for safety, etc.
- scale to sizes anywhere between an individual and the enterprise,
- are available at all locations required by the users, and
- protect the privacy and intellectual property of users.

In this world the focus is on information, customization, and functionality that extends beyond the abilities of monolithic applications. Essentially, information carries the behaviors and semantics needed to operate on it. Information is independent of location and becomes responsive to the environments it is used in and the contexts of individual users, and it is managed independently by both its consumers and providers.

TM ▶*The big hole that I can see is their comment that they want to prove "its benefits through applications that exploit its capabilities." The* ◀

6.2 Provenance

TM ▶*Explain provenance. It is a more recent area of study, but it also captures more of the kinds of information that will be useful to me.* ◀

6.3 Human-Computer Interface

The traditional primary organizational model provided to users has been the file/folder metaphor TM ▶*Need citation to 1965 Daley and the 1956 storage* *papers*◀, which was itself derived from physical filing cabinets. However, electronic storage is not bound by the same restrictions as a physical filing cabinet, as can be seen even in early work **TM** ▶*Again, this is Daley*◀ where the model added *links*; the closest equivalent to this in filing cabinets would be to make duplicate copies of a document — I have seen exactly this routinely practiced by bookkeeping and accounting professional, whether using physical or electronic filing systems. This works because the objects are generally *immutable*, but for electronic storage systems without deduplication it is not particularly space efficient.

The Human-Computer Interface community has been studying human-centered data organization models for decades. For example, the HCI community observed that hierarchical file structure is challenging for users with low spatial abilities [48]. This suggests why storage developers would not even see there is a problem here: the study of computer science correlates well with the development of spatial abilities [49]. In my own discussions with even senior computer scientists it is the introduction of *silos* that seems to make the hierarchical abstraction break. "Did I store that in Dropbox, or Google Drive, or was it on my laptop or my desktop computer?"

The wealth of research here is astonishing, yet does not appear to influence the design of storage systems to exploit the results of that research. Indeed, the systems community seems to be focused on a *search based* solution while the HCI community research suggests that search is *not* preferred by users — instead, users want to *navigate*:

> *When retrieving a file the user needs to choose between folder based navigation and query based search. There are obvious intuitive advantages of search for both retrieval and organization. Search seems to be more flexible and efficient for retrieval. It is flexible because it does not depend on users remembering the correct storage location; instead, in their query users can specify any file attribute they happen to remember...*
>
> *In fact, regardless of search engine quality, people consistently use search only as a 'last resort' for that minority of cases where they cannot remember file locations...* [50]

Thus, from an HCI perspective it would seem that one potential research direction would be to consider potential interfaces that mimic navigation over a search interface.

For example, recent work around data curation explores the idea of a "data dashboard" since the first step of curation is finding specific data to curate [20]. This work builds upon prior research showing that when presenting data to users it is important to ignore storage silo boundaries. Indeed, my reading of this work is that it presents what seems to be navigation even though it is implemented using a search mechanism. Equally important, this work also points to the benefits of not changing the *location* of data, but rather allowing construction of useful relationships via metadata. Thus, this work supports my ideas of ignoring silo boundaries and providing metadata for use by a similar tool. What it does not explore is a way for data storage to provide enhanced metadata, dynamic update, and notification of changes, which are important elements to making such a dashboard more useful for data visualization.

Bibliography

[1] J. Locke, *Locke's essays: An essay concerning human understanding, and A treatise on the conduct of the understanding (Complete in 1 volume with the author's last additions and corrections).* 1844 (**backrefpage** 1).

[2] V. Bush **andothers**, "As we may think," *The atlantic monthly*, **jourvol** 176, **number** 1, **pages** 101–108, 1945 (**backrefpages** 1, 18).

[3] P. J. Guo **and** M. Seltzer, "Burrito: Wrapping your lab notebook in computational infrastructure," **in***TaPP'12 Proceedings of the 4th USENIX conference on Theory and Practice of Provenance* 2012, **pages** 7–7 (**backrefpages** 6, 16, 33).

[4] M. K. McKusick, W. N. Joy, S. J. Leffler **and** R. S. Fabry, "A fast file system for unix," *ACM Transactions on Computer Systems*, **jourvol** 2, **number** 3, **pages** 181–197, 1984 (**backrefpage** 8).

[5] H. Custer, *Inside the Windows NT File System.* 1994 (**backrefpage** 8).

[6] J. Koo, J. Im, J. Song, J. Park, E. Lee, B. S. Kim **and** S. Lee, "Modernizing file system through in-storage indexing," **in***15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)* 2021, **pages** 75–92 (**backrefpages** 8, 27).

[7] M. Seltzer **and** N. Murphy, "Hierarchical file systems are dead," **in***HotOS'09 Proceedings of the 12th conference on Hot topics in operating systems* 2009, **pages** 1–1 (**backrefpages** 8, 27).

[8] P. Braam, "The lustre storage architecture," *arXiv preprint arXiv:1903.01955*, 2019 (**backrefpage** 8).

[9] R. Noronha **and** D. Panda, "Imca: A high performance caching front-end for glusterfs on infiniband," **in***2008 37th International Conference on Parallel Processing* 2008, **pages** 462–469 (**backrefpage** 8).

[10] S. A. Weil, S. A. Brandt, E. L. Miller, D. D. E. Long **and** C. Maltzahn, "Ceph: A scalable, high-performance distributed file system," **in***Proceedings of the 7th symposium on Operating systems design and implementation* 2006, **pages** 307–320 (**backrefpage** 8).

[11] Y. Kawai, A. Hasan, G. Iwai, T. Sasaki **and** Y. Watase, "A method for reliably managing files with rns in multi data grids," **in***Procedia Computer Science* **volume** 4, 2011, **pages** 412–421 (**backrefpage** 9).

[12] R. Pike, D. Presotto, K. Thompson, H. Trickey **and** P. Winterbottom, "The use of name spaces in plan 9," *SIGOPS Oper. Syst. Rev.*, **jourvol** 27, **number** 2, **pages** 72–76, **april** 1993, ISSN: 0163-5980. DOI: 10.1145/155848.155861. **url**: https://doi.org/10.1145/155848.155861 (**backrefpages** 9, 33).

[13] P. Dourish, R. Bentley, R. Jones **and** A. MacLean, "Getting some perspective: Using process descriptions to index document history," **in***Proceedings of the international ACM SIGGROUP conference on Supporting group work* 1999, **pages** 375–384 (**backrefpages** 9, 33).

[14] P. Dourish, "The appropriation of interactive technologies: Some lessons from placeless documents," *conference on computer supported cooperative work*, **jourvol** 12, **number** 4, **pages** 465–490, 2003 (**backrefpages** 9, 33).

[15] K. Briney, *Data Management for Researchers: Organize, maintain and share your data for research success.* 2015 (**backrefpage** 10).

[16] T. Berners-Lee, R. Fielding **and** L. Masinter, "Uniform resource identifiers (uri): Generic syntax," *RFC*, **jourvol** 2396, **pages** 1–40, 1998 (**backrefpage** 10).

[17] Y. Hua, H. Jiang, Y. Zhu, D. Feng **and** L. Tian, "Smartstore: A new metadata organization paradigm with semantic-awareness for next-generation file systems," **in***Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis* 2009, **pages** 1–12. DOI: 10.1145/1654059.1654070 (**backrefpages** 22, 33).

[18] C. Wang, K. Thareja, M. Stealey, P. Ruth **and** I. Baldin, "Comet: A distributed metadata service for federated cloud infrastructures," **in***2019 IEEE High Performance Extreme Computing Conference (HPEC)* 2019, **pages** 1–7. DOI: 10.1109/HPEC.2019.8916536 (**backrefpages** 22, 33).

[19] T. Pasquier, X. Han, M. Goldstein, T. Moyer, D. Eyers, M. Seltzer **and** J. Bacon, "Practical whole-system provenance capture," **in***Symposium on Cloud Computing (SoCC'17)* ACM, 2017 (**backrefpages** 23, 33).

[20] F. Vitale, "Personal data curation in the cloud age : Individual differences and design opportunities," phdthesis, University of British Columbia, 2020. DOI: http://dx.doi.org/10.14288/1.0392427. **url**: https://open.library.ubc.ca/collections/ubctheses/24/items/1.0392427 (**backrefpages** 27, 38).

[21] S. R. Mashwani **and** S. Khusro, "360° semantic file system: Augmented directory navigation for nonhierarchical retrieval of files," *IEEE Access*, **jourvol** 7, **pages** 9406–9418, 2019 (**backrefpage** 33).

[22] M. .-. Vef, R. Steiner, R. Salkhordeh, J. Steinkamp, F. Vennetier, J. .-. Smigielski **and** A. Brinkmann, "Delvefs - an event-driven semantic file system for object stores," **in***2020 IEEE International Conference on Cluster Computing (CLUSTER)* 2020, **pages** 35–46. DOI: 10.1109/CLUSTER49012.2020.00014 (**backrefpage** 33).

[23] M.-A. Vef, R. Steiner, R. Salkhordeh, J. Steinkamp, F. Vennetier, J.-F. Smigielski **and** A. Brinkmann, "Delvefs - an event-driven semantic file system for object stores," **in***2020 IEEE International Conference on Cluster Computing (CLUSTER)* 2020, **pages** 35–46 (**backrefpage** 33).

[24] S. Harrison **and** P. Dourish, "Re-place-ing space: The roles of place and space in collaborative systems," **in***Proceedings of the 1996 ACM conference on Computer supported cooperative work* 1996, **pages** 67–76 (**backrefpage** 33).

[25] D. Barreau **and** B. A. Nardi, "Finding and reminding: File organization from the desktop," *ACM Sigchi Bulletin*, **jourvol** 27, **number** 3, **pages** 39–43, 1995 (**backrefpage** 33).

[26] P. Dourish, W. K. Edwards, A. LaMarca, J. Lamping, K. Petersen, M. Salisbury, D. B. Terry **and** J. Thornton, “Extending document management systems with user-specific active properties,” *ACM Trans. Inf. Syst.*, **jourvol** 18, **number** 2, **pages** 140–170, **april** 2000, ISSN: 1046-8188. DOI: 10.1145/348751.348758. **url**: https://doi.org/10.1145/348751.348758 (**backrefpage** 33).

[27] D. K. Gifford, P. Jouvelot, M. A. Sheldon **and** J. W. O’Toole, “Semantic file systems,” **in***Proceedings of the Thirteenth ACM Symposium on Operating Systems Principles* **jourser** SOSP ’91, Pacific Grove, California, USA: Association for Computing Machinery, 1991, **pages** 16–25, ISBN: 0897914473. DOI: 10.1145/121132.121138. **url**: https://doi.org/10.1145/121132.121138 (**backrefpage** 33).

[28] M. A. Olson, “The design and implementation of the inversion file system,” **in***USENIX Winter 1993 Conference (USENIX Winter 1993 Conference)* San Diego, CA: USENIX Association, **january** 1993. **url**: https://www.usenix.org/conference/usenix-winter-1993-conference/design-and-implementation-inversion-file-system (**backrefpage** 33).

[29] S. Bloehdorn, O. Görlitz, S. Schenk **and** M. Völkel, “Tagfs - tag semantics for hierarchical file systems,” **in***Proceedings of the 6th International Conference on Knowledge Management (I-KNOW 06), Graz, Austria, September 6-8, 2006* **september** 2006 (**backrefpage** 33).

[30] D. Di Sarli **and** F. Geraci, “Gfs: A graph-based file system enhanced with semantic features,” **in***Proceedings of the 2017 International Conference on Information System and Data Mining* **jourser** ICISDM ’17, Charleston, SC, USA: Association for Computing Machinery, 2017, **pages** 51–55, ISBN: 9781450348331. DOI: 10.1145/3077584.3077591. **url**: https://doi.org/10.1145/3077584.3077591 (**backrefpage** 33).

[31] S. Shah, C. A. N. Soules, G. R. Ganger **and** B. D. Noble, “Using provenance to aid in personal file search,” **in***2007 USENIX Annual Technical Conference on Proceedings of the USENIX Annual Technical Conference* **jourser** ATC’07, Santa Clara, CA: USENIX Association, 2007, ISBN: 9998888776 (**backrefpage** 33).

[32] A. A. Rasel **and** M. E. Ali, "Uprove2: Privacy-aware, scalable, ubiquitous provenance to enhance file search," **in***2016 International Conference on Networking Systems and Security (NSysS)* IEEE, 2016, **pages** 1–5 (**backrefpage** 33).

[33] J. Liu, D. Feng, Y. Hua, B. Peng, P. Zuo **and** Y. Sun, "P-index: An efficient searchable metadata indexing scheme based on data provenance in cold storage," **in***International Conference on Algorithms and Architectures for Parallel Processing* Springer, 2015, **pages** 597–611 (**backrefpage** 33).

[34] L. Page, S. Brin, R. Motwani **and** T. Winograd, "The pagerank citation ranking: Bringing order to the web.," Stanford InfoLab, techreport, 1999 (**backrefpage** 33).

[35] R. Sandberg, D. Goldberg, S. Kleiman, D. Walsh **and** B. Lyon, "Design and implementation of the sun network filesystem," **in***Proceedings of the Summer USENIX conference* 1985, **pages** 119–130 (**backrefpage** 33).

[36] D. Bermbach, M. Klems, S. Tai **and** M. Menzel, "Metastorage: A federated cloud storage system to manage consistency-latency tradeoffs," **in***2011 IEEE 4th International Conference on Cloud Computing* 2011, **pages** 452–459. DOI: 10.1109/CLOUD.2011.62 (**backrefpage** 33).

[37] J. H. Howard, M. L. Kazar, S. G. Menees, D. A. Nichols, M. Satyanarayanan, R. N. Sidebotham **and** M. J. West, "Scale and performance in a distributed file system," *ACM Transactions on Computer Systems*, **jourvol** 6, **number** 1, **pages** 51–81, 1988 (**backrefpage** 33).

[38] J. H. Morris, M. Satyanarayanan, M. H. Conner, J. H. Howard, D. S. Rosenthal **and** F. D. Smith, "Andrew: A distributed personal computing environment," *Commun. ACM*, **jourvol** 29, **number** 3, **pages** 184–201, **march** 1986, ISSN: 0001-0782. DOI: 10.1145/5666.5671. **url**: https://doi.org/10.1145/5666.5671 (**backrefpage** 33).

[39] A. Adya, W. J. Bolosky, M. Castro, G. Cermak, R. Chaiken, J. R. Douceur, J. Howell, J. R. Lorch, M. Theimer **and** R. P. Wattenhofer, "Farsite: Federated, available, and reliable storage for an incompletely trusted environment," *SIGOPS Oper. Syst. Rev.*, **jourvol** 36, **number** SI, **pages** 1–14, **december** 2003,

ISSN: 0163-5980. DOI: 10.1145/844128.844130. **url**: https://doi.org/10.1145/844128.844130 (**backrefpage** 33).

[40] D. M. Ritchie **and** K. Thompson, “The unix time-sharing system,” *Commun. ACM*, **jourvol** 17, **number** 7, **pages** 365–375, **july** 1974, ISSN: 0001-0782. DOI: 10.1145/361011.361061. **url**: https://doi.org/10.1145/361011.361061 (**backrefpage** 33).

[41] A. N. Bessani, R. Mendes, T. Oliveira, N. F. Neves, M. Correia, M. Pasin **and** P. Verissimo, “Scfs: A shared cloud-backed file system.,” **in***USENIX Annual Technical Conference* Citeseer, 2014, **pages** 169–180 (**backrefpage** 33).

[42] Y. Demchenko, C. Ngo, C. de Laat **and** C. Lee, “Federated access control in heterogeneous intercloud environment: Basic models and architecture patterns,” **in***2014 IEEE International Conference on Cloud Engineering* 2014, **pages** 439–445. DOI: 10.1109/IC2E.2014.84 (**backrefpage** 33).

[43] J. Benet, “Ipfs - content addressed, versioned, p2p file system.,” *arXiv preprint arXiv:1407.3561*, 2014 (**backrefpage** 33).

[44] M. L. Mazurek, Y. Liang, W. Melicher, M. Sleeper, L. Bauer, G. R. Ganger, N. Gupta **and** M. K. Reiter, “Toward strong, usable access control for shared distributed data,” **in***Proceedings of the 12th USENIX conference on File and Storage Technologies* 2014, **pages** 89–103 (**backrefpage** 33).

[45] Y. Li, N. S. Dhotre, Y. Ohara, T. M. Kroeger, E. L. Miller **and** D. D. E. Long, “Horus: Fine-grained encryption-based security for large-scale storage,” **in***Proceedings of the 11th USENIX conference on File and Storage Technologies* 2013, **pages** 147–160 (**backrefpage** 33).

[46] A. Adya, W. J. Bolosky, M. Castro, G. Cermak, R. Chaiken, J. R. Douceur, J. Howell, J. R. Lorch, M. Theimer **and** R. P. Wattenhofer, “Farsite: Federated, available, and reliable storage for an incompletely trusted environment,” **in***Proceedings of the 5th symposium on Operating systems design and implementation (Copyright restrictions prevent ACM from being able to make the PDFs for this conference available for downloading)* **volume** 36, 2002, **pages** 1–14 (**backrefpage** 33).

[47] L. Carata, S. Akoush, N. Balakrishnan, T. Bytheway, R. Sohan, M. Seltzer **and** A. Hopper, "A primer on provenance," *Commun. ACM*, **jourvol** 57, **number** 5, **pages** 52–60, **may** 2014, ISSN: 0001-0782. DOI: 10.1145/2596628. **url**: https://doi.org/10.1145/2596628 (**backrefpage** 33).

[48] K. J. Vicente **and** R. C. Williges, "Accommodating individual differences in searching a hierarchical file system," *International Journal of Human-computer Studies International Journal of Man-machine Studies*, **jourvol** 29, **number** 6, **pages** 647–668, 1988 (**backrefpage** 37).

[49] J. Parkinson **and** Q. Cutts, "Investigating the relationship between spatial skills and computer science," **august** 2018, **pages** 106–114. DOI: 10.1145/3230977.3230990 (**backrefpage** 37).

[50] O. Bergman, T. Israeli **and** S. Whittaker, "Search is the future? the young search less for files," **in***Proceedings of the Association for Information Science and Technology* **volume** 56, 2019, **pages** 360–363 (**backrefpage** 37).

[51] M. Todesco, G. L. Owens, N. Bercovich, J.-S. Légaré, S. Soudi, D. O. Burge, K. Huang, K. L. Ostevik, E. B. M. Drummond, I. Imerovski, K. Lande, M. A. Pascual-Robles, M. Nanavati, M. Jahani, W. Cheung, S. E. Staton, S. Muños, R. Nielsen, L. A. Donovan, J. M. Burke, S. Yeaman **and** L. H. Rieseberg, "Massive haplotypes underlie ecotypic differentiation in sunflowers," *Nature*, **jourvol** 584, **number** 7822, **pages** 602–607, 2020.

[52] R. JESHION, "The significance of names," *Mind & Language*, **jourvol** 24, **number** 4, **pages** 370–403, 2009. DOI: https://doi.org/10.1111/j.1468-0017.2009.01367.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1468-0017.2009.01367.x. **url**: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1468-0017.2009.01367.x.

[53] J. Koetsier, *Digg pokes up its head and says, 'i'm not dead yet'*, (accessed January 6, 2019), **january** 2013. **url**: https://tinyurl.com/yam3xyp4.

[54] D. M. Ritchie **and** K. Thompson, "The unix time-sharing system," **in***ACM SIGOPS Operating Systems Review* ACM, **volume** 7, 1973, **page** 27.

[55] M. Wilkes, “A programmer’s utility filing system,” *The Computer Journal*, **jourvol** 7, **number** 3, **pages** 180–184, 1964.

[56] D. Reinsel, J. Gantz **and** J. Rydning, “Data age 2025: The digitization of the world from edge to core,” Seagate, techreport, **november** 2018, (accessed January 6, 2019). **url**: https://tinyurl.com/y95ogat4.

[57] D. MacDonald, “Datafile: A new tool for extensive file storage,” **in***Papers and discussions presented at the December 10-12, 1956, eastern joint computer conference: New developments in computers* ACM, 1956, **pages** 124–128.

[58] F. Halasz **and** T. P. Moran, “Analogy considered harmful,” **in***Proceedings of the 1982 conference on Human factors in computing systems* ACM, 1982, **pages** 383–386.

[59] J. S. Shapiro, “Extracting the lessons of multics,” 2004.

[60] Microsoft, *Data add-in interfaces*, (accessed January 6, 2019). **url**: https://tinyurl.com/y83elc34.

[61] Apple, *Search and spotlight*, (accessed January 6, 2019). **url**: https://tinyurl.com/yasf7kf2.

[62] J. Chou, “Findfs: Adding tag-based views to a hierarchical filesystem,” phdthesis, University of British Columbia, 2015.

[63] G. Marsden **and** D. E. Cairns, “Improving the usability of the hierarchical file system,” **in***Proceedings of the 2003 annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology* South African Institute for Computer Scientists **and** Information Technologists, 2003, **pages** 122–129.

[64] W. Jones, A. J. Phuwanartnurak, R. Gill **and** H. Bruce, “Don’t take my folders away!: Organizing personal information to get things done,” **in***CHI’05 extended abstracts on Human factors in computing systems* ACM, 2005, **pages** 1505–1508.

[65] D. R. Karger **and** W. Jones, “Data unification in personal information management,” *Communications of the ACM*, **jourvol** 49, **number** 1, **pages** 77–82, 2006.

[66] S. Ma **and** S. Wiedenbeck, “File management with hierarchical folders and tags,” **in***CHI’09 Extended Abstracts on Human Factors in Computing Systems* ACM, 2009, **pages** 3745–3750.

[67] C. Jensen, H. Lonsdale, E. Wynn, J. Cao, M. Slater **and** T. G. Dietterich, “The life and times of files and information: A study of desktop provenance,” **in***Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* ACM, 2010, **pages** 767–776.

[68] A. K. Karlson, G. Smith **and** B. Lee, “Which version is this?: Improving the desktop experience within a copy-aware computing ecosystem,” **in***Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* ACM, 2011, **pages** 2669–2678.

[69] W. Odom, A. Sellen, R. Harper **and** E. Thereska, “Lost in translation: Understanding the possession of digital things in the cloud,” **in***Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* ACM, 2012, **pages** 781–790.

[70] E. Thereska, O. Riva, R. Banks, S. Lindley, R. Harper **and** W. Odom, “Beyond file systems: Understanding the nature of places where people store their data,” Microsoft Research, techreport, 2013, (accessed March 29, 2018). **url**: https://www.microsoft.com/en-us/research/wp-content/uploads/2013/02/MSR-TR-2013-26.pdf.

[71] R. Harper, S. Lindley, E. Thereska, R. Banks, P. Gosset, G. Smyth, W. Odom **and** E. Whitworth, “What is a file?” **in***Proceedings of the 2013 conference on Computer supported cooperative work* ACM, 2013, **pages** 1125–1136.

[72] E. Méndez **and** J. Greenberg, “Linked data for open vocabularies and hive’s global framework,” *El profesional de la información*, **jourvol** 21, **number** 3, **pages** 236–244, 2012.

[73] C. Bothorel, J. D. Cruz, M. Magnani **and** B. Micenkova, “Clustering attributed graphs: Models, measures and methods,” *Network Science*, **jourvol** 3, **number** 3, **pages** 408–444, 2015.

[74] Z. Huo, L. Xiao, Q. Zhong, S. Li, A. Li, L. Ruan, S. Wang **and** L. Fu, “Mbfs: A parallel metadata search method based on bloomfilters using mapreduce for large-scale file systems,” *The Journal of Supercomputing*, **jourvol** 72, **number** 8, **pages** 3006–3032, 2016.

[75] T. Jayalakshmi **and** C. Chethana, “A semantic search engine for indexing and retrieval of relevant text documents,” 2016.

[76] N. Carvalho, H. Kim, M. Lu, P. Sarkar, R. Shekhar, T. Thakur, P. Zhou, R. H. Arpaci-Dusseau **and** I. Datos, "Finding consistency in an inconsistent world: Towards deep semantic understanding of scale-out distributed databases.," **in***HotStorage* 2016.

[77] Y. Gao, F. Song, X. Xie **and** X. Gao, "Implicit semantic text retrieval and distributed implementation for rural medical care," **in***Cloud Computing and Intelligence Systems (CCIS), 2016 4th International Conference on* IEEE, 2016, **pages** 264–267.

[78] Y. Hua, H. Jiang **and** D. Feng, "Real-time semantic search using approximate methodology for large-scale storage systems," *IEEE Transactions on Parallel and Distributed Systems*, **jourvol** 27, **number** 4, **pages** 1212–1225, 2016.

[79] D. Di Sarli **and** F. Geraci, "Gfs: A graph-based file system enhanced with semantic features," **in***Proceedings of the 2017 International Conference on Information System and Data Mining* ACM, 2017, **pages** 51–55.

[80] S. E. Lindley, G. Smyth, R. Corish, A. Loukianov, M. Golembewski, E. A. Luger **and** A. Sellen, "Exploring new metaphors for a networked world through the file biography," **in***Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* ACM, 2018, **page** 118.

[81] M. T. Khan, M. Hyun, C. Kanich **and** B. Ur, "Forgotten but not gone: Identifying the need for longitudinal data management in cloud storage," **in***Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* ACM, 2018, **page** 543.

[82] F. Vitale, I. Janzen **and** J. McGrenere, "Hoarding and minimalism: Tendencies in digital data preservation," **in***Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* ACM, 2018, **page** 587.

[83] A. Laursen, "A novel, tag-based file-system," (accessed March 29, 2018), mathesis, Macalester College, 2014. **url**: https://digitalcommons.macalester.edu/mathcs%5C_honors/34/.

[84] R. Boardman, R. Spence **and** M. A. Sasse, "Too many hierarchies? the daily struggle for control of the workspace," **in***Proceedings of HCI international* **volume** 1, 2003, **pages** 616–620.

[85] nayuki, *Designing better file organization around tags, not hierarchies*, (accessed October 1, 2018), **march** 2017. **url**: https://www.nayuki.io/page/designing-better-file-organization-around-tags-not-hierarchies.

[86] B. Martin, "Formal concept analysis and semantic file systems," **in***International Conference on Formal Concept Analysis* Springer, 2004, **pages** 88–95.

[87] ——, "Formal concept analysis and semantic file systems," phdthesis, University of Wollongong, 2008. **url**: https://ro.uow.edu.au/theses/260/.

[88] ——, "Open source libferris: Chasing the "everything is a file system" dream," *linux.com*, **january** 2014, (accessed September 4, 2018). **url**: https://www.linux.com/learn/open-source-libferris-chasing-everything-file-system-dream.

[89] C. A. Soules **and** G. R. Ganger, "Toward automatic context-based attribute assignment for semantic file systems," *Parallel data laboratory, Carnegie Mellon University. June*, 2004.

[90] F. J. Corbató **and** V. A. Vyssotsky, "Introduction and overview of the multics system," **in***Proceedings of the November 30–December 1, 1965, fall joint computer conference, part I* ACM, 1965, **pages** 185–196.

[91] S. Faubel **and** C. Kuschel, "Towards semantic file system interfaces," *CEUR Workshop Proceedings*, **jourvol** 401, 2008, ISSN: 16130073.

[92] M. Harlan **and** G. Parmer, "Joinfs: A semantic file system for embedded systems," **in***Proceedings of the International Conference on Embedded Systems and Applications (ESA)* The Steering Committee of The World Congress in Computer Science, Computer Engineering **and** Applied Computing (WorldComp), 2011, **page** 1.

[93] S. Jan, M. Li, G. Al-Sultany, H. Al-Raweshidy **and** I. A. Shah, "Semantic file annotation and retrieval on mobile devices," *Mobile Information Systems*, **jourvol** 7, **number** 2, **pages** 107–122, 2011, ISSN: 1574017X. DOI: 10.3233/MIS-2011-0113.

[94] C. Soules **and** G. Ganger, “Toward automatic context-based attribute assignment for semantic file systems,” *Parallel data laboratory, Carnegie Mellon ...*, **number** June, 2004. **url**: http://shiftleft.com/mirrors/www.hpl.hp.com/personal/Craig%7B%5C_%7DSoules/papers/CMU-PDL-04-105.pdf.

[95] M. Suguna **and** T. Anand, “Dynamic Metadata Management in Semantic File Systems,” **jourvol** 5, **number** 3, **pages** 44–47, 2015.

[96] P. Andrews, I. Zaihrayeu **and** J. Pane, “A classification of semantic annotation systems,” *Semantic Web*, **jourvol** 3, **number** 3, **pages** 223–248, 2012, ISSN: 15700844. DOI: 10.3233/SW-2011-0056.

[97] O. Eck **and** D. Schaefer, “A semantic file system for integrated product data management,” *Advanced Engineering Informatics*, **jourvol** 25, **number** 2, **pages** 177–184, 2011, ISSN: 14740346. DOI: 10.1016/j.aei.2010.08.005.

[98] Y. Hua, H. Jiang, Y. Zhu **and** D. Feng, “Rapport: Semantic-sensitive Namespace Management in Large-scale File Systems,” *CSE Technical reports*, 2010. **url**: http://digitalcommons.unl.edu/cgi/viewcontent.cgi?article=1123%7B%5C&%7Dcontext=csetechreports.

[99] A. Joshi, “Orion File System : File-level Host-based Virtualization,”

[100] R. Mander, G. Salomon **and** Y. Y. Wong, “A “Pile” Metaphor for Supporting Casual Organization of Information,” *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI ’92*, **pages** 627–634, 1992. DOI: 10.1145/142750.143055. **url**: http://portal.acm.org/citation.cfm?doid=142750.143055.

[101] B. Martin, “Formal concept analysis and semantic file systems,” *Concept Lattices*, 2004. **url**: http://link.springer.com/chapter/10.1007/978-3-540-24651-0%7B%5C_%7D9.

[102] B.-H. Ngo, C. Bac **and** F. Silber-Chaussumier, “Integrating ontology into semantic file systems,” **in***Huitièmes Journées Doctorales en Informatique et Réseaux (JDIR’07)* 2007, **pages** 139–142.

[103] P. Omvlee, “A novel idea for a new Filesystem,” *June*, 2009. **url**: http://referaat.cs.utwente.nl/conference/11/paper/6966/a-novel-idea-for-a-new-filesystem.pdf.

[104] G. Wang, H. Lu, G. Yu **and** B. YuBao, "Managing very large document collections using semantics," *Journal of Computer Science and Technology*, **jourvol** 18, **number** 3, **pages** 403–406, 2003.

[105] A. Shah, "Knowledge Management Enviroments for High Throughput Biology," **number** September, 2007.

[106] L. Up, "Tag , no di !," **pages** 1–23, 2016.

[107] B. Gopal **and** U. Manber, "Integrating content-based access mechanisms with hierarchical file systems," **in***OSDI* **volume** 99, 1999, **pages** 265–278.

[108] B. Martin **and** P. Eklund, "Applying formal concept analysis to semantic file systems leveraging wordnet," *ADCS 2005 - Proceedings of the Tenth Australasian Document Computing Symposium*, **pages** 56–63, 2005.

[109] H. Reiser, "Name spaces as tools for integrating the operating system rather than as ends in themselves," 2007, (accessed October 28, 2016 via archive.org). **url**: http://www.namesys.com/whitepaper.html.

[110] I. M. Benefits, R. Current, F. Systems, D. Anyone, D. Space, I. Approach, A. Approach, D. Canonical **and** A. Comparison, "Semantic File Systems," **pages** 1–17, 2016.

[111] V. Codocedo **and** A. Napoli, "Formal Concept Analysis and Information Retrieval – A Survey," *Formal Concept Analysis: 13th International Conference, ICFCA 2015*, **pages** 61–77, 2015. DOI: 10.1007/978-3-319-19545-2_4. **url**: http://dx.doi.org/10.1007/978-3-319-19545-2%7B%5C_%7D4.

[112] S. Ghemawat, H. Gobioff **and** S.-t. Leung, "Google_File_System," 2003, ISSN: 01635980. DOI: 10.1145/1165389.945450. arXiv: z0024.

[113] I. Jones, W. You **and** C. A. N. Do, "Tagxfs is a semantic file system. It extends the user space file system to a tag based hierarchy.," **pages** 1–5, 2016.

[114] M. Mahalingam, C. Tang **and** Z. Xu, "Towards a semantic, deep archival file system," *Proceedings of the IEEE Computer Society Workshop on Future Trends of Distributed Computing Systems*, **jourvol** 2003-Janua, **pages** 115–121, 2003. DOI: 10.1109/FTDCS.2003.1204321.

[115] M. Os, O. Xtagfs, M. Os, S. Comment, M. Os, T. F. Files, S. Comments, G. Info, S. Comments, I. Macfuse, D. Xtagfs **and** R. Xtagfs, "Archive," **pages** 1–2, 2016.

[116] A. Parker-Wood, D. D. E. Long, E. Miller, P. Rigaux **and** A. Isaacson, "A File By Any Other Name: Managing File Names with Metadata," *Proceedings of International Conference on Systems and Storage*, **pages** 1–11, 2014.

[117] V. Prabhakaran, A. C. Arpaci-Dusseau **and** R. H. Arpaci-Dusseau, "Analysis and evolution of journaling file systems," *Proceedings of the International Conference on Dependable Systems and Networks*, **page** 8, 2005. DOI: 10.1109/DSN.2005.65. **url**: http://dl.acm.org/citation.cfm?id=1247360.1247368.

[118] B. Schandl, "Representing linked data as virtual file systems," *CEUR Workshop Proceedings*, **jourvol** 538, 2009, ISSN: 16130073.

[119] P. Cellier, M. Ducassé, S. Ferré **and** O. Ridoux, "Formal concept analysis," *Lecture notes in computer science (R. Medina, S. Obiedkov, eds.)*, **jourvol** 4933, 2008.

[120] T. Strong **and** P. Akundi, "A semantic cloud for file system annotation," **in***Proceedings of the International Conference on Semantic Web and Web Services (SWWS)* The Steering Committee of The World Congress in Computer Science, Computer Engineering **and** Applied Computing (WorldComp), 2013, **page** 24.

[121] L. Xu, Z. Huang, H. Jiang, L. Tian **and** D. Swanson, "VSFS: A searchable distributed file system," *Proceedings of PDSW 2014: 9th Parallel Data Storage Workshop - Held in Conjunction with SC 2014: The International Conference for High Performance Computing, Networking, Storage and Analysis*, **pages** 25–30, 2014. DOI: 10.1109/PDSW.2014.10.

[122] M. A. Olson **andothers**, "The design and implementation of the inversion file system," **in***USENIX Winter* 1993, **pages** 205–218.

[123] A. Leung, A. Parker-Wood **and** E. Miller, "Copernicus: A scalable, high-performance semantic file system," *University of California, Santa ...*, **number** October, 2009. **url**: http://www.ssrc.ucsc.edu/papers/ssrctr-09-06.pdf.

[124] B. Schandl **and** B. Haslhofer, “The sile model - A semantic file system infrastructure for the desktop,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, **jourvol** 5554 LNCS, **pages** 51–65, 2009, ISSN: 03029743. DOI: 10.1007/978-3-642-02121-3_8.

[125] A. Shah **and** L. Caves, “ConceptOntoFs : A Semantic File System for Inferno,” *1st International Workshop on Plan 9*, 2006.

[126] T. Wiki, G. P. Open **and** B. Explore, “oniony / TMSU,” **pages** 1–2, 2016.

[127] K. J. Vicente, B. C. Hayes **and** R. C. Williges, “Assaying and isolating individual differences in searching a hierarchical file system,” *Human factors*, **jourvol** 29, **number** 3, **pages** 349–359, 1987.

[128] T. W. Malone, “How do people organize their desks?: Implications for the design of office information systems,” *ACM Transactions on Information Systems (TOIS)*, **jourvol** 1, **number** 1, **pages** 99–112, 1983.

[129] I. G. Terrizzano, P. M. Schwarz, M. Roth **and** J. E. Colino, “Data wrangling: The challenging yourney from the wild to the lake.,” **in***CIDR* 2015.

[130] R. Watson, S. Dekeyser **and** N. Albadri, “Exploring the design space of metadata-focused file management systems,” **in***Proceedings of the Australasian Computer Science Week Multiconference* ACM, 2017, **page** 20.

[131] P. J. Guo **and** M. Seltzer, “Burrito: Wrapping your lab notebook in computational infrastructure,” **in***Proceedings of the 4th USENIX Workshop on the Theory and Practice of Provenance* **jourser** TaPP’12, Boston, MA: USENIX Association, 2012. **url**: http://dl.acm.org/citation.cfm?id=2342875.2342882.

[132] P. Macko, V. J. Marathe, D. W. Margo **and** M. I. Seltzer, “Llama: Efficient graph analytics using large multiversioned arrays,” **in***Proceedings of the 31st IEEE International Conference on Data Engineering* IEEE, 2015.

[133] D. Margo **and** M. Seltzer, “A scalable distributed graph partitioner,” *Proceedings of the VLDB Endowment*, **jourvol** 8, **number** 12, **pages** 1478–1489, 2015.

[134] P. Macko, D. Margo **and** M. Seltzer, “Performance introspection of graph databases,” **in***Proceedings of the 6th International Systems and Storage Conference* ACM, 2013, **page** 18.

[135] I. Amazon Web Services, *Object tagging*, (accessed January 8, 2019), 2018. **url**: https://docs.aws.amazon.com/AmazonS3/latest/dev/object-tagging.html.

[136] I. V. Balabine, R. Kandasamy **and** J. A. Skier, *File system interface to a database*, US Patent 5,937,406, **august** 1999.

[137] M. Stonebraker, P. M. Aoki, R. Devine, W. Litwin **and** M. Olson, “Mariposa: A new architecture for distributed data,” **in***Data Engineering, 1994. Proceedings. 10th International Conference* IEEE, 1994, **pages** 54–65.

[138] I. V. Balabine, R. Kandasamy **and** J. A. Skier, *Database interface for database unaware applications*, US Patent 6,442,548, **august** 2002.

[139] Z. Xu, M. Karlsson, C. Tang **and** C. T. Karamanolis, “Towards a semantic-aware file store.,” **in***HotOS* 2003, **pages** 181–187.

[140] A. Kashyap **and** A. Kashyap, “File system extensibility and reliability using an in-kernel database,” phdthesis, Stony Brook University, 2004.

[141] S. Ames, N. Bobb, K. M. Greenan, O. S. Hofmann, M. W. Storer, C. Maltzahn, E. L. Miller **and** S. A. Brandt, “Lifs: An attribute-rich file system for storage class memories,” **in***Proceedings of the 23rd IEEE/14th NASA Goddard Conference on Mass Storage Systems and Technologies* IEEE, 2006.

[142] J. Koren, A. Leung, Y. Zhang, C. Maltzahn, S. Ames **and** E. Miller, “Searching and navigating petabyte-scale file systems based on facets,” **in***Proceedings of the 2nd international workshop on Petascale data storage: held in conjunction with Supercomputing’07* ACM, 2007, **pages** 21–25.

[143] H. H. Huang, N. Zhang, W. Wang, G. Das **and** A. S. Szalay, “Just-in-time analytics on large file systems,” *IEEE Transactions on Computers*, **jourvol** 61, **number** 11, **pages** 1651–1664, 2012.

[144] Z. Xu, M. Karlsson, C. Tang **and** C. Karamanolis, *Semantic file system*, US Patent 7,617,250, **november** 2009.

[145] N. Murphy, M. Tonkelowitz **and** M. Vernal, *The design and implementation of the database file system*, 2002.

[146] A. Leung, I. Adams **and** E. L. Miller, “Magellan: A searchable metadata architecture for large-scale file systems,” *University of California, Santa Cruz, Tech. Rep. UCSC-SSRC-09-07*, 2009.

[147] O. Frieder **and** S. Kapoor, *Hierarchical structured abstract data organization system*, US Patent 8,209,358, **june** 2012.

[148] K.-Y. Whang **and** R. Krishnamurthy, “The multilevel grid file-a dynamic hierarchical multidimensional file structure.,” **in***DASFAA* World Scientific, **volume** 1991, 1991, **pages** 449–459.

[149] H. P. Kumar, C. Plaisant **and** B. Shneiderman, “Browsing hierarchical data with multi-level dynamic queries and pruning,” *International journal of human-computer studies*, **jourvol** 46, **number** 1, **pages** 103–124, 1997.

[150] F. Van Ham **and** J. J. van Wijk, “Beamtrees: Compact visualization of large hierarchies,” *Information Visualization*, **jourvol** 2, **number** 1, **pages** 31–39, 2003.

[151] S. Dumais, E. Cutrell, J. J. Cadiz, G. Jancke, R. Sarin **and** D. C. Robbins, “Stuff i’ve seen: A system for personal information retrieval and re-use,” **in***ACM SIGIR Forum* ACM, **volume** 49, 2016, **pages** 28–35.

[152] C. A. Soules **and** G. R. Ganger, “Why can’t i find my files?: New methods for automating attribute assignment,” **in***HotOS* 2003, **pages** 115–120.

[153] J. MacCormick, N. Murphy, M. Najork, C. A. Thekkath **and** L. Zhou, “Boxwood: Abstractions as the foundation for storage infrastructure.,” **in***OSDI* **volume** 4, 2004, **pages** 8–8.

[154] G. Marchionini, “Exploratory search: From finding to understanding,” *Communications of the ACM*, **jourvol** 49, **number** 4, **pages** 41–46, 2006.

[155] S. Brandt, C. Maltzahn, N. Polyzotis **and** W.-C. Tan, “Fusing data management services with file systems,” **in***Proceedings of the 4th Annual Workshop on Petascale Data Storage* ACM, 2009, **pages** 42–46.

[156] A. W. Leung, M. Shao, T. Bisson, S. Pasupathy **and** E. L. Miller, “Spyglass: Fast, scalable metadata search for large-scale storage systems.,” **in***FAST* **volume** 9, 2009, **pages** 153–166.

[157] K. Ren **and** G. A. Gibson, "Tablefs: Enhancing metadata efficiency in the local file system.," **in***USENIX Annual Technical Conference* 2013, **pages** 145–156.

[158] S. Niazi, M. Ismail, S. Haridi, J. Dowling, S. Grohsschmiedt **and** M. Ronström, "Hopsfs: Scaling hierarchical file system metadata using newsql databases.," **in***FAST* 2017, **pages** 89–104.

[159] R. V. H. Van Staereling, R. Appuswamy, D. C. van Moolenbroek **and** A. S. Tanenbaum, "Efficient, modular metadata management with loris," **in***Networking, Architecture and Storage (NAS), 2011 6th IEEE International Conference on* IEEE, 2011, **pages** 278–287.

[160] R. Appuswamy, D. van Moolenbroek **and** A. Tanenbaum, "Loris - a redundant array of independent physical layers," English, **in***Proceedings of the sixteenth annual conference of the Advanced School for Computing and Imaging (ASCI'10)* Advanced School for Computing **and** Imaging (ASCI), 2010.

[161] R. Appuswamy, "Building a file-based storage stack: Modularity and flexibility in loris," 2014.

[162] M. Harlan, "JOIN FS : a Semantic File System for Embedded Systems," 2011.

[163] C. Min, S. Kashyap, B. Lee, C. Song **and** T. Kim, "Cross-checking Semantic Correctness : The Case of Finding File System Bugs," *Sosp '15*, **pages** 361–377, 2015. DOI: 10.1145/2815400.2815422.

[164] P. Open **and** B. Explore, "Fuse::TagLayer," **pages** 1–2, 2016.

[165] J. Chou, "FindFS - Adding Tag-Based Views to a Hierarchical Filesystem by," **number** August, 2015.

[166] B.-h. Ngo, C. Bac, B.-h. Ngo, C. Bac **and** F. Silber-chaussumier, "To cite this version : Integrating Ontology into Semantic File Systems," 2015.

[167] B. Gopal, "Integrating Content-Based Access Mechanisms with Hierarchical File Systems 1 Introduction 2 The Design of the HAC File System,"

[168] G. Beydoun, "Formal concept analysis for an e-learning semantic web," *Expert Systems with Applications*, **jourvol** 36, **number** 8, **pages** 10 952–10 961, 2009, ISSN: 09574174. DOI: 10.1016/j.eswa.2009.02.023.

[169] D. Gifford, P. Jouvelot, M. Sheldon, J. O. Toole, D. K. Gifford, M. A. Sheldon **and** J. W. O. Toole, "13 . Paper : Semantic File Systems Semantic File Systems," *Computer*, **pages** 16–25, 2006.

[170] S. L. Hyvonen **and** S. Louise, "Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.," *Dissertation*, **number** May, 2004.

[171] J. Siekmann, "Formal Concept Analysis," **pages** 0–45,

[172] T. Strong **and** P. Akundi, "A Semantic Cloud for File System Annotation," **pages** 1–4,

[173] R. Escriva **and** E. Gun Sirer, "The Design and Implementation of the WAVE Transactional File System," **number** February, **pages** 1–14, 2015. arXiv: 1509.07821. **url**: http://arxiv.org/pdf/1509.07821v1.pdf.

[174] B. Agee, C. Daly **and** J. Brown, "Ubuntu A tag-based filesystem for ubuntu," **pages** 1–3, 2016.

[175] D. K. Gifford, P. Jouvelot, M. a. Sheldon, J. W. O'Toole, P. Jouvelotl, M. a. Sheldon **and** W. J. O'Toole, Jr, "Semantic file systems," *ACM SIGOPS Operating Systems Review*, **jourvol** 25, **number** 5, **pages** 16–25, 1991, ISSN: 01635980. DOI: 10.1145/121133.121138. **url**: http://portal.acm.org/citation.cfm?doid=121133.121138%7B%5C%%7D5Cnhttp://dl.acm.org/citation.cfm?id=121133.121138.

[176] V. U. Guide, "Archive," **pages** 1–2, 2016.

[177] X. Liu, G. Niv, P. Shenoy, K. Ramakrishnan **and** J. Van der Merwe, "The case for semantic aware remote replication," *Proceedings of the second ACM workshop on Storage security and survivability - StorageSS '06*, **page** 79, 2006. DOI: 10.1145/1179559.1179575. **url**: http://dl.acm.org/citation.cfm?id=1179559.1179575.

[178] R. M. Needham **and** A. D. Birrell, "The cap filing system," **in***ACM SIGOPS Operating Systems Review* ACM, **volume** 11, 1977, **pages** 11–16.

[179] M. Arenas, B. C. Grau, E. Kharlamov, Š. Marciuška **and** D. Zheleznyakov, "Faceted search over rdf-based knowledge graphs," *Web Semantics: Science, Services and Agents on the World Wide Web*, **jourvol** 37, **pages** 55–74, 2016.

[180] D. Tunkelang, “Faceted search,” *Synthesis lectures on information concepts, retrieval, and services*, **jourvol** 1, **number** 1, **pages** 1–80, 2009.

[181] M. Hearst, “Design recommendations for hierarchical faceted search interfaces,” **in***ACM SIGIR workshop on faceted search* Seattle, WA, 2006, **pages** 1–5.

[182] V. Klungre **and** M. Giese, “Evaluating a faceted search index for graph data,” **in***OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"* Springer, 2018, **pages** 573–583.

[183] R. Walton, “Searching high and low: Faceted navigation as a model for online archival finding aids (a literature review).,” *Journal for the Society of North Carolina Archivists*, **jourvol** 12, 2015.

[184] ——, “Looking for answers: A usability study of online finding aid navigation,” *The American Archivist*, **jourvol** 80, **number** 1, **pages** 30–52, 2017.

[185] P. H. Cleverley **and** S. Burnett, “Retrieving haystacks: A data driven information needs model for faceted search,” *Journal of Information Science*, **jourvol** 41, **number** 1, **pages** 97–113, 2015.

[186] H. C. Huurdeman, M. L. Wilson **and** J. Kamps, “Active and passive utility of search interface features in different information seeking task stages,” **in***Proceedings of the 2016 ACM on Conference on Human Information Interaction and Retrieval* ACM, 2016, **pages** 3–12.

[187] R. Sandberg, “The sun network file system: Design, implementation and experience,” **in***in Proceedings of the Summer 1986 USENIX Technical Conference and Exhibition* Citeseer, 1986.

[188] B. Sidebotham **andothers**, *Volumes: the andrew file system data structuring primitive.* Carnegie Mellon University, Information Technology Center, 1986.

[189] M. K. Sreenivas, S. R. Steiner, A. Brjazovski **and** S. H. Agarwal, *Bypass of the namespace hierarchy to open files*, US Patent 7,925,681, **april** 2011.

[190] O. van Rest, S. Hong, J. Kim, X. Meng **and** H. Chafi, “Pgql: A property graph query language,” **in***Proceedings of the Fourth International Workshop on Graph Data Management Experiences and Systems* ACM, 2016, **page** 7.

[191] N. Francis, A. Green, P. Guagliardo, L. Libkin, T. Lindaaker, V. Marsault, S. Plantikow, M. Rydberg, P. Selmer **and** A. Taylor, “Cypher: An evolving query language for property graphs,” **in***Proceedings of the 2018 International Conference on Management of Data* ACM, 2018, **pages** 1433–1445.

[192] R. Angles, M. Arenas, P. Barcelo, P. Boncz, G. Fletcher, C. Gutierrez, T. Lindaaker, M. Paradies, S. Plantikow, J. Sequeda **andothers**, “G-core: A core for future graph query languages,” **in***Proceedings of the 2018 International Conference on Management of Data* ACM, 2018, **pages** 1421–1432.

[193] M. Rudolf, M. Paradies, C. Bornhövd **and** W. Lehner, “The graph story of the sap hana database.,” **in***BTW* Citeseer, **volume** 13, 2013, **pages** 403–420.

[194] J. Masci, M. M. Bronstein, A. M. Bronstein **and** J. Schmidhuber, “Multimodal similarity-preserving hashing,” *IEEE transactions on pattern analysis and machine intelligence*, **jourvol** 36, **number** 4, **pages** 824–830, 2014.

[195] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numerische mathematik*, **jourvol** 1, **number** 1, **pages** 269–271, 1959.

[196] M. Conover, “Analysis of the windows vista security model,” *Symantec Advanced Threat Research*, 2006.

[197] R. Pike, D. Presotto, K. Thompson, H. Trickey **and** P. Winterbottom, “The use of name spaces in plan 9,” **in***Proceedings of the 5th workshop on ACM SIGOPS European workshop: Models and paradigms for distributed systems structuring* ACM, 1992, **pages** 1–5.

[198] M. J. Spier, T. N. Hastings **and** D. N. Cutler, “An experimental implementation of the kernel/domain architecture,” **in***Proceedings of the Fourth ACM Symposium on Operating System Principles* **jourser** SOSP ’73, New York, NY, USA: ACM, 1973, **pages** 8–21. DOI: 10.1145/800009.808043. **url**: http://doi.acm.org/10.1145/800009.808043.

[199] L. Rosenfeld **and** P. Morville, *Information architecture for the world wide web.* ” O’Reilly Media, Inc.”, 2002.

[200] D. Beaver, S. Kumar, H. C. Li, J. Sobel, P. Vajgel **andothers**, "Finding a needle in haystack: Facebook's photo storage.," **in***OSDI* **volume** 10, 2010, **pages** 1–8.

[201] S. Muralidhar, W. Lloyd, S. Roy, C. Hill, E. Lin, W. Liu, S. Pan, S. Shankar, V. Sivakumar, L. Tang **andothers**, "F4: Facebook's warm blob storage system," **in***Proceedings of the 11th USENIX conference on Operating Systems Design and Implementation* USENIX Association, 2014, **pages** 383–398.

[202] Y. Mao, E. Kohler **and** R. T. Morris, "Cache craftiness for fast multicore key-value storage," **in***Proceedings of the 7th ACM european conference on Computer Systems* ACM, 2012, **pages** 183–196.

[203] D. Borthakur, *Rocksdb: A persistent key-value store*, 2014.

[204] H. Lim, B. Fan, D. G. Andersen **and** M. Kaminsky, "Silt: A memory-efficient, high-performance key-value store," **in***Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles* ACM, 2011, **pages** 1–13.

[205] M. K. McKusick, W. N. Joy, S. J. Leffler **and** R. S. Fabry, "A fast file system for unix," *ACM Transactions on Computer Systems (TOCS)*, **jourvol** 2, **number** 3, **pages** 181–197, 1984.

[206] M. Cao, S. Bhattacharya **and** T. Ts'o, "Ext4: The next generation of ext2/3 filesystem.," **in***LSF* 2007.

[207] N. Bronson, Z. Amsden, G. Cabrera, P. Chakka, P. Dimov, H. Ding, J. Ferris, A. Giardullo, S. Kulkarni, H. Li, M. Marchukov, D. Petrov, L. Puzar, Y. J. Song **and** V. Venkataramani, "TAO: Facebook's distributed data store for the social graph," **in***Presented as part of the 2013 USENIX Annual Technical Conference (USENIX ATC 13)* San Jose, CA: USENIX, 2013, **pages** 49–60, ISBN: 978-1-931971-01-0. **url**: https://www.usenix.org/conference/atc13/technical-sessions/presentation/bronson.

[208] P. J. Shetty, R. P. Spillane, R. R. Malpani, B. Andrews, J. Seyster **and** E. Zadok, "Building workload-independent storage with vt-trees," **in***Presented as part of the 11th USENIX Conference on File and Storage Technologies (FAST 13)* San Jose, CA: USENIX, 2013, **pages** 17–30, ISBN: 978-1-931971-99-7. **url**:

https://www.usenix.org/conference/fast13/technical-sessions/presentation/shetty.

[209] T. Pasquier, X. Han, M. Goldstein, T. Moyer, D. Eyers, M. Seltzer **and** J. Bacon, "Practical whole-system provenance capture," **in***Proceedings of the 2017 Symposium on Cloud Computing* **jourser** SoCC '17, Santa Clara, California: ACM, 2017, **pages** 405–418, ISBN: 978-1-4503-5028-0. DOI: 10.1145/3127479.3129249. **url**: http://doi.acm.org/10.1145/3127479.3129249.

[210] K.-K. Muniswamy-Reddy, D. A. Holland, U. Braun **and** M. Seltzer, "Provenance-aware storage systems," **in***Proceedings of the Annual Conference on USENIX '06 Annual Technical Conference* **jourser** ATEC '06, Boston, MA: USENIX Association, 2006, **pages** 4–4. **url**: http://dl.acm.org/citation.cfm?id=1267359.1267363.

[211] S. Bloehdorn, O. Görlitz, S. Schenk, M. Völkel **andothers**, "Tagfs-tag semantics for hierarchical file systems," **in***Proceedings of the 6th International Conference on Knowledge Management (I-KNOW 06), Graz, Austria* **volume** 8, 2006, **pages** 6–8.

[212] J. Webber **and** I. Robinson, *A programmatic introduction to neo4j*. Addison-Wesley Professional, 2018.

[213] Microsoft, *Welcome to azure cosmos db*, (accessed January 17, 2019), **january** 2019. **url**: https://docs.microsoft.com/en-us/azure/cosmos-db/introduction.

[214] A. Kyrola, G. E. Blelloch **and** C. Guestrin, "Graphchi: Large-scale graph computation on just a pc," USENIX, 2012.

[215] Y. Low, J. E. Gonzalez, A. Kyrola, D. Bickson, C. E. Guestrin **and** J. Hellerstein, "Graphlab: A new framework for parallel machine learning," *arXiv preprint arXiv:1408.2041*, 2014.

[216] D. Nguyen, A. Lenharth **and** K. Pingali, "A lightweight infrastructure for graph analytics," **in***Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles* ACM, 2013, **pages** 456–471.

[217] S. Salihoglu **and** J. Widom, "Gps: A graph processing system," **in***Proceedings of the 25th International Conference on Scientific and Statistical Database Management* ACM, 2013, **page** 22.

[218] G. Malewicz, M. H. Austern, A. J. Bik, J. C. Dehnert, I. Horn, N. Leiser **and** G. Czajkowski, “Pregel: A system for large-scale graph processing,” **in***Proceedings of the 2010 ACM SIGMOD International Conference on Management of data* ACM, 2010, **pages** 135–146.

[219] J. E. Gonzalez, R. S. Xin, A. Dave, D. Crankshaw, M. J. Franklin **and** I. Stoica, “Graphx: Graph processing in a distributed dataflow framework.,” **in***OSDI* **volume** 14, 2014, **pages** 599–613.

[220] J. Shun **and** G. E. Blelloch, “Ligra: A lightweight graph processing framework for shared memory,” **in***ACM Sigplan Notices* ACM, **volume** 48, 2013, **pages** 135–146.

[221] M. Factor, K. Meth, D. Naor, O. Rodeh **and** J. Satran, “Object storage: The future building block for storage systems,” **in***Local to Global Data Interoperability-Challenges and Technologies, 2005* IEEE, 2005, **pages** 119–123.

[222] G. Watumull, J. Gerend, L. Poggemeyer, A. Hansen **and** K. Ainapure, *Resilient file system (refs) overview*, (accessed January 17, 2019). **url**: https://tinyurl.com/y832s3ky.

[223] G. A. I. Barnard **and** L. Fein, “An information filing and retrieval system for the engineering and management records of a large-scale computer development project,” English, *American Documentation (pre-1986)*, **jourvol** 9, **number** 3, **page** 208, **july** 1958, Copyright - Copyright Wiley Periodicals Inc. Jul 1958; Last updated - 2010-08-27. **url**: http://prx.library.gatech.edu/login?url=https://search.proquest.com/docview/195441816?accountid=11107.

[224] Amazon Web Services, *Editing Object Metadata*, Online. Accessed 2021-02-01. https://docs.aws.amazon.com/AmazonS3/latest/user-guide/add-object-metadata.html, 2021.

[225] A. authors, *Anonymized title.*

[226] A. R. Fox, A. Satyarnarayan, P. Guo, H. Xia **and** J. D. Hollan, *Chs: Medium: A human-centered information space:designing dynamic personalized visual information*, 2019. **url**: http://hci.ucsd.edu/220/NSF_Hollan_220.pdf.

[227] F. Vitale, “Personal data curation in the cloud age: Individual differences and design opportunities,” phdthesis, University of British Columbia, 2020.

[228] L. Orr, M. Balazinska **and** D. Suciu, *Sample debiasing in the themis open world database system (extended version)*, 2020. arXiv: 2002.09799 [cs.DB].

[229] R. Arpaci-Dusseau **and** A. Arpaci-Dusseau, "Naming and binding of objects," **in***Operating Systems: Three Easy Pieces* 2021, Chapter 41. **url**: http://pages.cs.wisc.edu/~remzi/OSTEP/file-ffs.pdf.

[230] Apple, Inc., *Search drives user engagement*, 2016.

[231] Google Inc., *Google Cloud Storage API - Blobs/Objects*, Version 1.35.0. Online. Accessed 2021-02-01. https://googleapis.dev/python/storage/latest/blobs.html, 2019.

[232] Amazon Web Services, Inc, *Amazon Simple Storage Service Developer Guide*, API Version 2006-03-01. Online. Accessed 2021-02-01. https://docs.aws.amazon.com/AmazonS3/latest/dev/s3-dg.pdf, 2021.

[233] Google Inc., *Google cloud storage documentation - object naming guidelines*, Online. Accessed 2021-02-01. https://cloud.google.com/storage/docs/naming-objects, 2020.

[234] IBM Corporation, *WebSphere Application Server (Distributed operating systems) Namespace Federation*, Version 8.5.5. Online. Accessed 2021-02-01. https://www.ibm.com/support/knowledgecenter/SSEQTP_8.5.5/com.ibm.websphere.base.doc/ae/cnam_federation.html, 2020.

[235] Oracle, *Java Platform Standard Ed. 7 - Package org.omg.CosNaming*, Online. Accessed 2021-02-01. https://docs.oracle.com/javase/7/docs/api/org/omg/CosNaming/package-summary.html, 2020.

[236] Object Management Group, Inc., *Naming Service Specification*, Version 1.3. Online. Accessed 2021-02-01. https://www.omg.org/spec/NAM/1.3/PDF, 2004.

[237] The Kubernetes Authors, *Kubernetes Concepts: Namespaces*, Version 1.16. Online. Accessed 2021-02-01. https://v1-16.docs.kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/, 2021.

[238] HUAWEI TECHNOLOGIES CO., LTD., *Multi-cloud container platform: Namespaces*, Issue 01. Online. Accessed 2021-02-01. https://support.huaweicloud.com/intl/en-us/usermanual-mcp/mcp_01_0025.html, 2020.

[239] S. Srinivas, *An Introduction to HDFS Federation*, Online. Accessed 2021-02-01.
https://blog.cloudera.com/an-introduction-to-hdfs-federation/, 2011.

[240] A. Bakre, D. Krishnamurthy, K. Muthyala, C. Sharma **and** R. Talwadker, *Federated namespace of heterogeneous storage system namespaces*, Patent US10812313B2. Online. Accessed 2021-02-01.
https://patents.google.com/patent/US10812313B2/en, 2021.

[241] R. Daley **and** P. Neumann, "A general-purpose file system for secondary storage," **in***Proceedings of the November 30–December 1, 1965, fall joint computer conference, part I* ACM, 1965, **pages** 213–229.

[242] M. Seltzer **and** N. Murphy, "Hierarchical file systems are dead," **in***Proceedings of the 12th Conference on Hot Topics in Operating Systems* **jourser** HotOS'09, Monte Verità, Switzerland: USENIX Association, 2009, **page** 1.

[243] J. C. Mogul, "Representing information about files," phdthesis, Stanford University, 1986.

[244] O. Bergman, T. Israeli **and** S. Whittaker, "Factors hindering shared files retrieval," *Aslib Journal of Information Management*, **jourvol** 72, **number** 1, **pages** 130–147, 2019.

[245] H. Sim, A. Khan, S. S. Vazhkudai, S. Lim, A. R. Butt **and** Y. Kim, "An integrated indexing and search service for distributed file systems," *IEEE Transactions on Parallel and Distributed Systems*, **jourvol** 31, **number** 10, **pages** 2375–2391, 2020. DOI: 10.1109/TPDS.2020.2990656.

[246] V. Atlidakis, J. Andrus, R. Geambasu, D. Mitropoulos **and** J. Nieh, "Posix abstractions in modern operating systems: The old, the new, and the missing," **in***Proceedings of the Eleventh European Conference on Computer Systems* **jourser** EuroSys '16, London, United Kingdom: Association for Computing Machinery, 2016, ISBN: 9781450342407. DOI: 10.1145/2901318.2901350. **url**: https://doi.org/10.1145/2901318.2901350.

[247] N. Albadri, R. Watson **and** S. Dekeyser, "Treetags: Bringing tags to the hierarchical file system," **in***Proceedings of the Australasian Computer Science Week Multiconference* **jourser** ACSW '16, Canberra, Australia: Association for Computing Machinery, 2016,

ISBN: 9781450340427. DOI: 10.1145/2843043.2843868. **url**: https://doi.org/10.1145/2843043.2843868.

[248] A. Parker-Wood, D. D. E. Long, E. Miller, P. Rigaux **and** A. Isaacson, "A file by any other name: Managing file names with metadata," **in***Proceedings of International Conference on Systems and Storage* **jourser** SYSTOR 2014, Haifa, Israel: Association for Computing Machinery, 2014, **pages** 1–11, ISBN: 9781450329200. DOI: 10.1145/2611354.2611367. **url**: https://doi.org/10.1145/2611354.2611367.

[249] P. H. Lensing, T. Cortes **and** A. Brinkmann, "Direct lookup and hash-based metadata placement for local file systems," **in***Proceedings of the 6th International Systems and Storage Conference* **jourser** SYSTOR '13, Haifa, Israel: Association for Computing Machinery, 2013, ISBN: 9781450321167. DOI: 10.1145/2485732.2485741. **url**: https://doi.org/10.1145/2485732.2485741.

[250] I. Beckwith, "Id3fs," **url**: https://erislabs.net/ianb/projects/id3fs/id3fs-index.html.

[251] D. Rahn, *CVS log for src/sys/ufs/ufs/Attic/extattr.h*, Revision 1.2. Online. Accessed 2021-02-01. https://cvsweb.openbsd.org/src/sys/ufs/ufs/Attic/extattr.h, 2005.

[252] H. Kang **and** B. Shneiderman, "Mediafinder: An interface for dynamic personal media management with semantic regions," **in***CHI '03 Extended Abstracts on Human Factors in Computing Systems* **jourser** CHI EA '03, Ft. Lauderdale, Florida, USA: Association for Computing Machinery, 2003, **pages** 764–765, ISBN: 1581136374. DOI: 10.1145/765891.765977. **url**: https://doi.org/10.1145/765891.765977.

[253] C. Lee, D. Sim, J. Hwang **and** S. Cho, "F2fs: A new file system for flash storage," **in***13th {USENIX} Conference on File and Storage Technologies ({FAST} 15)* 2015, **pages** 273–286.

[254] M. Rosenblum **and** J. K. Ousterhout, "The design and implementation of a log-structured file system," *ACM Transactions on Computer Systems (TOCS)*, **jourvol** 10, **number** 1, **pages** 26–52, 1992.

[255] R. Pike, D. Presotto, K. Thompson, H. Trickey **and** P. Winterbottom, "The use of name spaces in plan 9," *SIGOPS Oper. Syst. Rev.*, **jourvol** 27, **number** 2, **pages** 72–76, **april** 1993, ISSN: 0163-5980. DOI: 10.1145/155848.155861. **url**: https://doi.org/10.1145/155848.155861.

[256] R. Pike **and** P. Weinberger, "The Hideous Name," **in***USENIX Summer 1985 Conference Proceedings* Portland Oregon, 1985.

[257] D. K. Gifford, P. Jouvelot, M. A. Sheldon **and** J. W. O'Toole, "Semantic file systems," **in***Proceedings of the Thirteenth ACM Symposium on Operating Systems Principles* **jourser** SOSP '91, Pacific Grove, California, USA: Association for Computing Machinery, 1991, **pages** 16–25, ISBN: 0897914473. DOI: 10.1145/121132.121138. **url**: https://doi.org/10.1145/121132.121138.

[258] M. L. Kazar, B. W. Leverett, O. T. Anderson, V. Apostolides, B. A. Bottos, S. Chutani, C. Everhart, W. A. Mason, S.-T. Tu **and** E. R. Zayas, "Decorum file system architectural overview.," *USENIX Summer*, **pages** 151–164, 1990.

[259] K. Birman **and** T. Joseph, "Exploiting virtual synchrony in distributed systems," **in***Proceedings of the eleventh ACM Symposium on Operating systems principles* **volume** 21, 1987, **pages** 123–138.

[260] T. W. Malone, "How do people organize their desks?: Implications for the design of office information systems," *ACM Transactions on Information Systems*, **jourvol** 1, **number** 1, **pages** 99–112, 1983.

[261] J. H. Saltzer, "Naming and binding of objects," **in***Operating Systems: An Advanced Course* R. Bayer, R. M. Graham **and** G. Seegmüller, **editors**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1978, **pages** 99–208, ISBN: 978-3-540-35880-0. DOI: 10.1007/3-540-08755-9_4. **url**: https://doi.org/10.1007/3-540-08755-9_4.

[262] F. Revol, "Universal file system extended attributes namespace," **in***International Conference on Dublin Core and Metadata Applications* 2011, **pages** 69–73.

[263] Linux man-pages Project, *Xattr - extended attributes*, Linux Programmer's Manual, Online. Accessed 2021-02-01. https://man7.org/linux/man-pages/man7/xattr.7.html, 2020.

[264] A. Grünbacher, “POSIX Access Control Lists on Linux,” **in***Proceedings of the FREENIX Track: 2003 USENIX Annual Technical Conference* San Antonio, TX, USA, 2003.

[265] D. Kasatkin **and** Z. Miriam, “Integrity measurement architecture (ima),” 2021. **url**: https://sourceforge.net/p/linux-ima/wiki/Home/.

[266] G. Linux, “How selinux controls file and directory accesses,” 2020. **url**: https://wiki.gentoo.org/wiki/SELinux/Tutorials/How_SELinux_controls_file_and_directory_accesses.