H-0318 (HAI1301-006) January 20, 2013
Computer Science

IBM Research Report

Preservation DataStores in the Cloud (PDS Cloud): Long Term Digital Preservation in the Cloud

Simona Rabinovici-Cohen, John Marberg, Kenneth Nagin
IBM Research Division
Haifa Research Laboratory
Mt. Carmel 31905
Haifa, Israel

Research Division
Almaden - Austin - Beijing - Cambridge - Haifa - India - T. J. Watson - Tokyo - Zurich

Preservation DataStores in the Cloud (PDS Cloud): Long Term Digital Preservation in the Cloud

Simona Rabinovici-Cohen, John Marberg, Kenneth Nagin
IBM Research – Haifa
Haifa 31905, Israel
{simona,marberg,nagin}@il.ibm.com

ABSTRACT

The emergence of the cloud and advanced object-based storage services provides opportunities to support novel models for long term preservation of digital assets. Among the benefits of this approach is leveraging the cloud's inherent scalability and redundancy to dynamically adapt to evolving needs of digital preservation. Preservation DataStores in the Cloud (PDS Cloud) is an OAIS-based preservation-aware storage service employing multiple heterogeneous cloud providers. It materializes the logical concept of a preservation information-object into physical cloud storage objects. Preserved information can be interpreted by deploying virtual appliances in the compute cloud, built from readily available components and provisioned with data objects together with their designated rendering software. PDS Cloud has a hierarchical data model and resource naming structure, supporting independent tenants whose assets are organized in multiple aggregations based on content and value. Each aggregation has a separate preservation profile that is reconfigurable as requirements keep changing over the long term. Continuous changes to data objects, life-cycle activities, virtual appliances and cloud providers are applied in a manner transparent to the client. PDS Cloud is being developed as an infrastructure component of the European Union ENSURE project, where it is used for preservation of medical and financial data.

Keywords

clouds, data storage systems, platform virtualization, information management

1. INTRODUCTION

Cloud technology is emerging as an infrastructure suitable for building large and complex systems. Storage and compute resources provisioned from converged infrastructure and shared resource pools present a cost-effective alternative to the traditional in-house data center. The cloud provides new levels of scalability, elasticity and availability, and enables simple access to data from any location and any device. Moreover, the cloud exposes a data model of objects that include data with inherent user-defined key-value metadata that is treated as one unit. Thus, the cloud is an attractive platform for a new type of scalable fixed content applications that require rich metadata.

Long Term Digital Preservation (LTDP) is the ability to sustain the understandability and usability of digital objects in the distant future regardless of changes in technologies and in the "designated communities" that create and consume these digital objects. A growing number of organizations now have a requirement to preserve large volumes of digital content for decades while maintaining access to it. Regulatory compliance and legal issues require preservation of email archives, medical records, financial accounts, aircraft designs, oil-field data, and more.

The LTDP challenge can be divided into *bit preservation* and *logical preservation*. Bit preservation is the ability to retrieve the bits in the face of physical media degradation or obsolescence, corruption or destruction due to errors or malicious attacks, or even environmental catastrophes such as fire and flooding. Logical preservation involves preserving the understandability and usability of the data, despite changes that will take place in servers, operating systems, data management products, applications and even users over the long term. Additionally, logical preservation needs to maintain the provenance of the data, along with its authenticity and integrity, so that current and future systems can ensure that only legitimate users access that data. Most data subject to long term preservation is fixed content that hardly changes once written. Due to its nature, this kind of data is typically accessed infrequently.

The core standard for digital preservation systems is Open Archival Information System (OAIS), an ISO standard since 2003 (ISO 14721:2003 OAIS) [1]. OAIS specifies the terms, concepts and reference models for a system dedicated to preserving digital assets for a designated community. OAIS defines a functional model that describes the entities in a long term digital preservation system and the flows among these entities. One of

the main concepts of OAIS is the Archival Information Package (AIP), which is the basic object stored in the archival storage of a preservation system. The AIP is a composite object that includes the core data being preserved and additional large metadata needed in the preservation services. Archival storage based on OAIS contains the services and functions used for storage, retrieval and management of AIPs.

LTDP can benefit from cloud technology. With its many vendors, open interfaces and subscription payment model, cloud storage offers the flexibility needed to address the dynamically evolving requirements of preservation. The potentially unlimited capacity provides inherent scalability and redundancy. Likewise, the traditional notion of investing in high end storage may not always be economically feasible or desirable. In the long run, the ability to easily switch between different vendors is a key factor in ensuring the economical viability of any preservation solution. Finally, cloud storage is sometimes positioned as well suited for latency-tolerant applications such as backup and archiving, thus making it attractive for digital preservation repositories.

A key notion of LTDP is that preserved information content needs to be interpretable and understandable in the future (logical preservation). The importance of most digital data is not its original state. Rather its value lies in the information that it conveys. In other words, its bits are mutable as long as it retains the valued information. Indeed, continuously transforming data formats to keep pace with evolving standards is a reasonable and probably desirable preservation strategy, as data that proves its mutability in the present is more likely to be mutable and relevant in the future.

For the purpose of interpreting evolving data formats, virtual appliances (VAs) running in the compute cloud and built from readily available components are a viable alternative to specialized local emulation environments. With this paradigm, VAs will evolve in the same ecosystem as the data being supported. They are maintained in a manner transparent to users, who are no longer required to be involved in altering the way they access the data as it keeps transforming. Support of VAs thereby becomes an integrated aspect of the preservation environment in the cloud.

Our main contribution is the definition of PDS Cloud (Preservation DataStores in the cloud), an OAIS-based preservation-aware storage service in a multi-cloud environment. Compared to existing cloud storage, or even traditional archival systems, PDS Cloud supports logical preservation and materializes the concept of logical preservation information object into physical cloud storage objects. It constitutes a cloud broker that interconnects between the OAIS entities and the multiple diverse clouds. It defines a way to ensure grouping of metadata with data for the long term that helps automate preservation processes and perform them close to the data.

PDS Cloud has a hierarchical data model supporting independent tenants whose assets are organized in multiple aggregations based on content and value. Each aggregation has a separate preservation profile that is reconfigurable dynamically and transparently as requirements keep changing. The Preserved content can be accessed using virtual appliances provisioned with data objects from the storage cloud together with the designated rendering software.

The research on PDS Cloud was initiated in the ENSURE project (Enabling kNowledge, Sustainability, Usability and Recovery for Economic Value) [2]. This is a European Union FP7 project which aims at extending the state of the art in digital preservation, focusing on business and scientific use cases, such as health care and financial data. PDS Cloud is implemented as the storage infrastructure component of the ENSURE experimental prototype.

The rest of this paper is organized as follows. In section 2 we provide gap analysis of several cloud platforms in regard to preservation. Section 3 gives a high-level view of the PDS Cloud architecture. The data model of PDS Cloud is introduced in Section 4. Section 5 describes how we map PDS Cloud entities to cloud storage. Virtual appliances are discussed in Section 6. In section 7 we overview the demonstration prototype of PDS Cloud. Section 8 surveys related work. We conclude in Section 9 with a summary and future plans.

2. CLOUD USABILITY GAP ANALYSIS

We have surveyed several existing cloud platforms, in regard to their usability for digital preservation, considering the needs of PDS Cloud as a preservation storage services layer. The purpose of this study was to evaluate the capabilities of the candidate platforms, identify the important differences among them, and understand what functionality can be exploited for preservation needs. The main focus of the study is storage cloud capabilities, but our requirements include also compute cloud functionality, specifically the ability to publish and deploy virtual appliances that are used for interpretation of preservation objects.

2.1 Analysis of Cloud Capabilities

It became evident from our study that many of the relevant features of the multiple platforms are very similar, if not identical. consequently, the platforms also share similar shortcomings.

- **Bit reliability:** Guarantees of bit reliability in the cloud are insufficient for preservation systems. Storage cloud platforms generally perform a fixity check upon storing an object, but do not have an option to repeat this check periodically. Also,

regulatory requirements for digital preservation may entail performing fixity using multiple algorithms, whereas cloud platforms usually support one predefined method.

- **Data lock-in:** Cloud systems currently suffer from data lock-in, where there are no easy means to get the data out of the system in its entirety, reliably and efficiently. This poses a great risk as services providers may go out of business or become unreliable over time.
- **Certification and trust:** Storage clouds lack support for auditing, certification and trust, including secure access. This is critical in preservation of commercial and business oriented data where there is a need to provide evidence of regulatory compliance. Specifically, preservation related regulatory requirements entail support for data encryption, anonymization, periodic auditing (including fixity), replication, versioning, and more.
- **Metadata** One of the key concepts in the OAIS model for preservation is the extensive use of metadata, strongly coupled with the raw data as part of the AIP. Moreover, metadata is likely to change and grow significantly in size during the extended lifetime of the AIP. Storage clouds today have rather limited support of metadata. The allowable space for metadata (per object) is much too small for the extensive size of preservation metadata. A related issue is the lack of capabilities in most clouds for search on metadata, i.e., the ability to filter objects by tags.
- **Event tracking:** Storage clouds do not capture events that are part of the object provenance and need to be recorded for preservation, such as access to objects, media refresh events, etc. This is particularly crucial in the cloud, because data in the cloud can be shared widely. Provenance is a means for consumers to verify data authenticity or identity. It is of importance to keep the provenance together with the data, to guarantee consistency.
- **Workload management:** Today, most uses of storage clouds are for backup or for applications requiring a high level of sharing, downloads and streaming. Therefore, the cost models today are well suited to this type of usage. Preservation systems pose a different type of workload, requirements, and SLAs resulting in possibly different cost models. For example, consumer access to preserved data may be infrequent, resulting in less demand for streaming and downloading, On the other hand, preservation maintenance may utilize more efficient (and less expensive) access to the preserved objects directly on the cloud, and this is typically performed in bulk on many objects.
- **Storage and compute synergy:** Computational support is needed for preservation, as storage is active over time. Data Management functionalities may be offered transparently in the cloud (e.g., handling data replication and disaster recovery). This is insufficient for a digital preservation solution, since data migration and transformation are an integral part of the Preservation Digital Asset Lifecycle (PDAL), and should be configurable and operable by the client. In the cloud, a viable approach would be to exploit compute cloud services in conjunction with cloud storage.
- **Logical preservation:** Preservation is more than just ensuring the bit integrity of the object content. It must also support logical data preservation, so that the content is understandable in the future. Today, the cloud environment does not have built-in support for logical preservation. As part of PDS Cloud, we are leveraging cloud based virtual appliances to assist in interpretation and visualization of digital content. One must address the problem of how to economically maximize accessibility to the content now and in the future, while reducing the probability of future obsolescence of the virtual appliance.

Notwithstanding these issues, cloud storage provides a scalable, sound and cost-effective infrastructure that is essential for preservation solutions. The selection the cloud platforms to be supported by PDS Cloud is influenced by the intent to offer clients a variety of cloud platforms with diverse deployment characteristics.

Specifically, there are trade-offs between a locally deployed private cloud and a public enterprise cloud deployed by a service provider. Among others, this affects security-related requirements, scalability and elasticity, cost, as well as performance. For example, clients with sensitive data may prefer a private cloud, or alternatively require additional security measures when using a public cloud, such as encryption and secure communication. Also, a public cloud typically has a pay-per-usage cost model, whereas a private cloud incurs equipment ownership and maintenance costs, but no direct usage cost.

The PDS Cloud architecture presented in subsequent sections attempts to mitigate many of the gaps identified here, as will be pointed out throughout this paper.

2.2 Evaluation of Cloud Platforms

The cloud platforms engaged in PDS Cloud in the scope of the ENSURE project experimental implementation include Amazon S3 and EC2 (enterprise) [3] and Openstack Swift and Nova (private, open source) [4]. For additional diversity, we may in the future engage a research platform, for example VISION Cloud [5], which is an EU project.

There are additional relevant storage cloud platforms, such as Eucalyptus [6], Rackspace [7] and EMC Atmos [8]. Largely, the basic features of such platforms are

similar, so in our study we are focusing on the platforms mentioned above. It should also be noted that Vision Cloud is still in early development stages, thus it is not immediately applicable, and is used here as a reference platform.

Another cloud service environment we have considered is the DuraCloud [9] open source platform. Its goal is to provide a fully integrated environment where services and data can be managed across multiple cloud providers, supporting data storage, replication and access, as well as services to support data preservation, such as data format transformation and fixity checking. Presently, this evolving platform does not aim to be compliant with the OAIS reference model [10], and therefore is not directly suitable for PDS Cloud. We will continue to watch DuraCloud, and may plan to integrate some aspects of PDS Cloud with it in the future.

Our evaluation of features of the selected storage clouds covers the following broad categories: data model, storage policies, security, administration, and software environment. A comparative survey of compute clouds was not performed.

One of the most obvious problems is the rather limited support of user metadata. The allowable space for metadata (per object) is much too small for the extensive size of preservation metadata. We can overcome this limitation by packaging the required metadata as part of the AIP body along with the rest of the content. A related issue is the lack of capabilities for search on metadata. Swift and S3 do not support filtering (searching) of object by tags (metadata).

Vision Cloud will provide more extensive metadata support. This is not an imminent problem for ENSURE, since PDS Cloud works in a larger environment, where ontologies are supported. In the long run, however, PDS Cloud should exist as an independent service layer. Search on object properties (metadata) should be available at the cloud level, regardless of higher level infrastructure. The goal is to avoid any dependencies on upper layers, and specifically the need to provide an explicit list of objects on which to perform preservation actions. Instead, just the properties to be searched on should be specified. This has strong impact on implementation and on development effort.

All cloud platforms offer a RESTful [11] interface, and the URI path organization is similar. However, the security (authorization) models are different. This entails different treatment of the interaction with each cloud platform. Also, some details of the requests (e.g., HTTP headers) are different. These gaps are mitigated using a two-layer multi-cloud interface between PDS Cloud and the cloud platforms. The upper layer is a common API (uniform across all platforms) used by the preservation services. The lower layer, invoked from the upper layer, handles the specifics of interaction with each cloud platform, using an individual "driver" per platform. Adding or changing a cloud provider would entail using a different driver. Ideally, preservation services would not have to know which cloud platform is being used. This is feasible as long as the data models are identical. PDS Cloud engages an existing multi-cloud interface service, to save significant development effort, and minimize our exposure to gaps among cloud providers.

An important logistical difference between cloud platforms is locality. Swift is a private cloud, and storage can be entirely under control of a single client organization. S3, on the other hand, is a public service shared by many customers, whereas storage is located and provisioned in the public domain. This influences security-related requirements of the preservation services. Clients with sensitive data may prefer a local cloud, or require additional security measures when using a public cloud, such as encryption of data, secure communication, etc. In particular, encryption entails additional infrastructure, since this is not provided by cloud platforms. It should be observed that a requirement for encryption may be imposed by the customer, regardless of whether the cloud is public or private.

In the cloud environment, it is desirable to offload preservation maintenance work to the cloud itself. This can be done on any platform that provides compute cloud services along with the storage cloud, such as Amazon EC2 or Openstack Nova.

When data is stored in the cloud in encrypted form, there are limitations on what maintenance can be offloaded to the cloud. Fixity, for example, could be done on encrypted data. On the other hand, data transformations can only be done on decrypted data, and therefore may not be performed entirely on the cloud. If security is to be genuine, encryption/decryption should not be entrusted to the cloud provider.

In a preservation system, there are references and links within the various parts of the AIP. Additionally, there are references and links between distinct AIPs, e.g., a reference from an AIP to its RepInfo. Furthermore, multi-cloud support implies that AIPs may reference each other even if they do not reside on the same cloud. In cloud storage, objects exist within containers. Such a "logical" directory structure of object names in the same container can be exploited as a partial solution and assist referencing within the various parts of the AIP. This would need to be extended in order to support referencing objects on remote clouds (for example using appropriate metadata).

The choice among cloud providers is significantly influenced by cost. A local cloud platform commonly does not need billing for consumption of storage or compute resources. However, it incurs capital expenses for

hardware, and operational expenses for maintenance of the server software. On the other extreme, a public cloud service entails billing for resource consumption (Amazon provides a variety of cost models), but no costs of hardware and maintenance. The decision as to which cost option is most suitable is in the hands of the individual customer. In ENSURE, a cost modeling component will be provided, so for versatility of this model it is relevant that PDS Cloud demonstrate support of different platform types (private and public).

3. PDS CLOUD ARCHITECTURE

The gap analysis in Section 2 reveals that simply "throwing" data onto the cloud is not an adequate solution for digital preservation repositories, and more advanced management and reliability mechanisms are needed; thus PDS Cloud is designed to overcome some of these gaps. PDS Cloud supports logical preservation and materializes the logical concept of a preservation information-object into a physical storage object. It is motivated by the idea that digital preservation systems will be more robust and have lower probability for data corruption or loss if preservation-related functionality is offloaded to the storage. Another goal is to support automation of preservation processes.

The foundations of PDS Cloud were established in PDS [12], a preservation storage architecture using Object Storage Devices (OSD). Here the scope is expanded and adapted for the cloud environment. Moreover, new cloud-specific goals and features are added. The main additional goals of PDS Cloud are:

- Support access to multiple cloud storage and cloud compute platforms, as well as enable migration of data between different clouds. This includes using multiple clouds concurrently, while taking advantage of special capabilities in each platform.
- Deploy a flexible data model for a multi-tenant multi-cloud environment, with configurable data management capabilities that can adhere to diverse aggregations of digital assets, having different requirements for preservation that can change over time.
- Enhance future understandability of content by supporting data access using cloud based virtual appliances. The virtual machine instance is created from a previously published image or from readily available components, and provisioned with the desired preservation data content and the designated software needed to render the data. The user can manage the provisioned virtual machine instance and perform termination, suspension and resumption of the instance.
- Offer advanced OAIS-based services, such as fixity (integrity) checks, provenance and auditing that complement the generic clouds capabilities. Also, support complex interrelated objects in the cloud and manage relationships and links while maintaining referential integrity.
- Provide computational storage via storlets, which are pluggable modules executing in a sandbox in the storage cloud, close to the data, utilizing the data locality to perform data-intensive functions more efficiently.

The PDS Cloud architecture and its components are illustrated in Figure 1. The dotted box components are intended for future implementation, and described here briefly to show their context in the overall picture.

PDS Cloud is architected as an intermediate service layer. It constitutes a broker that interconnects between OAIS entities and the multiple clouds. On the front end, PDS Cloud exposes to the client a set of OAIS-based preservation services such as ingest, access, delete and preservation actions on OAIS AIPs. On the back end it leverages heterogeneous storage and compute cloud platforms from different vendors.

AIPs may be stored on multiple clouds simultaneously, to exploit different storage cloud capabilities and pricing structures. This also serves to avoid lock-in, and to increase data survivability by replicating data across clouds in different locations [13].

It is assumed that user authentication and authorization to the preservation system are performed in upper levels of the runtime environment, prior to calling PDS Cloud. Thus, PDS Cloud can safely perform the requested operations.

PDS Cloud is divided into two main layers:

- **Multi-Cloud Service** – handles access to a heterogeneous set of cloud storage and compute platforms. This layer is agnostic to preservation. Its role is to hide the specific interfaces and capabilities exposed by each different cloud platform.
- **Preservation Engine** – provides preservation functionality for AIPs. It accepts requests via the PDS Cloud external interface and services them, utilizing a variety of functional handlers and eventually the multi-cloud service underneath.

3.1 Multi-Cloud Service

To mitigate data lock-in, the architecture supports deployment of multiple clouds from different vendors. Further, we emphasize the synergy between cloud storage and cloud compute for access to preserved data. Heterogeneity allows the user to experiment with diverse technologies, and to examine interoperability across space that will hopefully lead to interoperability across time, namely preservation. To implement this methodology efficiently, we separate the multi-cloud service from the preservation engine.

The National Institute of Standards and Technology (NIST) has defined four different deployment models

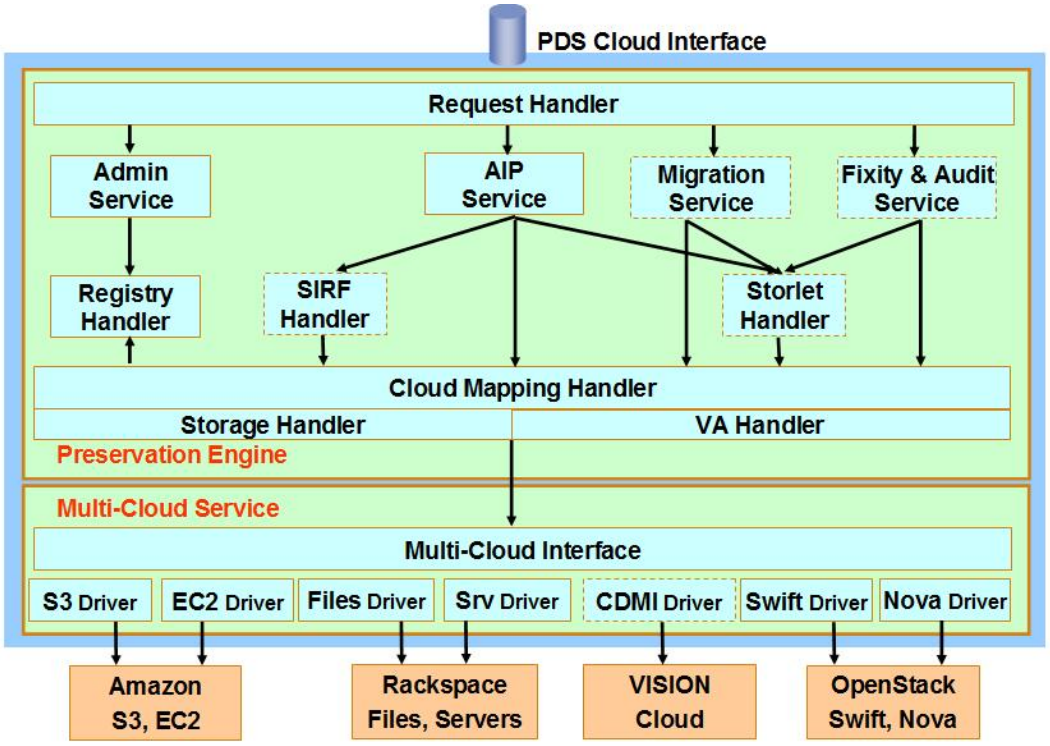

Figure 1: PDS Cloud high-level architecture

for cloud deployment: public, private, community and hybrid[14]. Public clouds are hosted by a third party and provide shared resources to multiple organizations. As they operate on a large scale, public clouds provide the most rapid elasticity, enabling better automation of preservation actions. However, the user has little control over management of the cloud. Furthermore, security issues may arise when storing the data outside its original premises. Private clouds, on the other hand, are operated by a single organization. The user has control over all aspects; yet, there are typically fewer resources with limited elasticity and speed. Community clouds are run for a consortium of cooperating organizations, and thus can have the benefits of both public and private clouds, but the participating organizations need to trust one another. Finally, hybrid clouds are a composition of the other models.

PDS Cloud supports all four deployment models. Moreover, it enables storing high value data in multiple clouds at the same time to increase resiliency over time. This is especially important for public clouds where there is dependence on a third party.

In our initial implementation, we are experimenting with the following cloud platforms:

- Amazon [3] S3 storage and EC2 compute – public cloud
- Rackspace [7] Files storage and Servers compute – public cloud
- Openstack [4] Swift storage and Nova compute open source – private cloud
- VISION Cloud [5] storage – EU research platform exposing standard CDMI[15] interface (to be used in the future).

We leverage jclouds[16], an open source cloud interface library, as the basis for the multi-cloud service. It comprises a unified interface (multi-cloud interface component) and a set of drivers that implement the interactions with the individual storage and compute clouds underneath.

3.2 Preservation Engine

The preservation engine provides all the preservation related functionality. At the top is the Request Handler which is the server side of the HTTP protocol, interacting with PDS Cloud clients. It receives each HTTP request, parses it, validates it, then hands it over to one of the other PDS Cloud services for processing. At the bottom of the preservation engine resides the Cloud Mapping Handler that handles mapping from AIPs to the cloud object model (see Section 5). It further utilizes the Storage Handler and the VA Handler, which interact with the multi-cloud service layer (jclouds), to handle all cloud operations related to storage or virtual appliances.

The core of the preservation engine comprises four main services. *AIP Service* is in charge of ingest, access

and delete of various types of AIPs (data, RepInfo, etc.) and orchestrates the management of the AIP metadata as defined in OAIS. It generates unique AIP identifiers, and manages provenance and relations among the various AIPs. Additionally, in the future it will utilize the *SIRF Handler* to support SIRF[17] containers in the cloud. SIRF (Self-contained Information Retention Format) is a storage container format for preservation objects that provides a catalog with metadata related to the entire contents of the container as well as to the individual objects and their interrelationship.

Admin Service handles definition and profiling of tenants, aggregations and policies. It engages the *Registry Handler* to maintain this operational information in a non-volatile registry (see also discussion on data model in Section 4).

Migration Service, which is intended for future implementation, coordinates transformations of AIPs from one format to another as requirements and environment evolve. According to OAIS, migration comprises four functions: refreshment, replication, repackaging and transformation. The first three concern bit preservation, which in the cloud environment is performed by the underlying cloud storage. Transformation, on the other hand, is logical preservation, and is performed by the migration service.

Fixity and Audit Service handles flexible periodic fixity (integrity) checks on AIPs using a choice of multiple fixity algorithms, to ensure the bits are not altered. This is required since, as pointed out in Section 2, the bit reliability guaranteed by the underlying storage clouds is generally not strong and durable enough for long term bit preservation. The service can also be used for system audits by a third party.

The AIP Service, Migration Service and Fixity Service sometimes require performing data-intensive computational tasks, such as validation, transformation, fixity checks, de-identification, and encryption. It is much more efficient, cheaper and more reliable to do these tasks near the data, namely instead of moving the data from storage to a processing machine, move the computation module to the storage server and run it there. Computational modules deployable in the storage system are called "storlets" [18]. We plan to develop extensions for open source storage clouds, such as Openstack Swift, to support storlets. The *Storlet Handler* will manage the deployment of storlets for preservation actions, which will execute periodically or when explicitly triggered.

The next three sections elaborate further on aspects of the architecture that have already been implemented. Specifically, we discuss the data model, the mapping of AIPs to the cloud object model, and advanced automatic rendering of AIPs using virtual appliances.

4. HIERARCHICAL DATA MODEL

The storage cloud is emerging as an archiving solution, in addition to being a data storage platform. However, traditional requirements associated with data management remain: data must be classified in order to define the policies and characteristics related to aspects such as replication, retention, integrity, and security. Data handling facilities can be mapped to cloud storage. Indeed, in order to take full advantage of the cloud, it is necessary to leverage multiple clouds to exploit different storage cloud capabilities and pricing structures, and to avoid lock-in. For example, multi-cloud support can enhance data durability by replicating data across clouds in different locations. Further, data assets belonging to different tenants must be logically insulated from each other even when residing in the same storage cloud. A cloud storage service would support multiple independent tenant domains, with completely separate administrative ownership and users.

Enterprises using an archiving storage service typically organize their data in multiple collections having different defined policies and facilities for their data management, based on criteria such as information type, value to the organization, and storage cost. As the needs of the organization evolve over time, the data management profile of a collection should be dynamically configurable. For example, the administrator may set policies for replication and for integrity checking according to regulatory requirements, which are likely to change over time. The storage service may continuously monitor performance aspects such as storage costs, space usage and platform availability, and decide to change storage location or synchronize among replicas. Also, as cloud technology advances and improved cloud services by new providers become available, it should be possible to migrate data to another cloud platform. Ideally, any changes in data management should be completely transparent to the user of the storage service.

The idea is to enable transparent and dynamic configuration of data management in a multi-cloud multi-tenancy environment. Users should be able to access the data without needing to know underlying details such as the identity of the storage cloud providers, and should not have to adapt continuously to changes in configuration. Moreover, the storage service could comprise an engine that initiates data management actions autonomously and transparently, for example replication across storage providers, integrity checks, and recovery.

Data stored in the cloud is commonly accessed using a hierarchical naming path. The data model typically consists of containers and objects, but may vary significantly among platforms. This data model is adequate for a simple data storage platform, where the user works with resources in a specific storage cloud, but it does

not address the requirements of a multi-cloud/multi-tenancy storage service.

PDS Cloud uses a logical data model and uniform hierarchical resource naming path for entities in a preservation storage system. Details of data management configuration are hidden from the user, yet integrated into the model. The model is logical in the sense that it is not tied to any specific implementation. It lends itself to different realizations, depending on the capabilities of the cloud storage platforms being used.

4.1 Model Definition

The top-down hierarchy in the data model consists of: tenants, aggregations, dockets, and objects. This is shown in Figure 2.

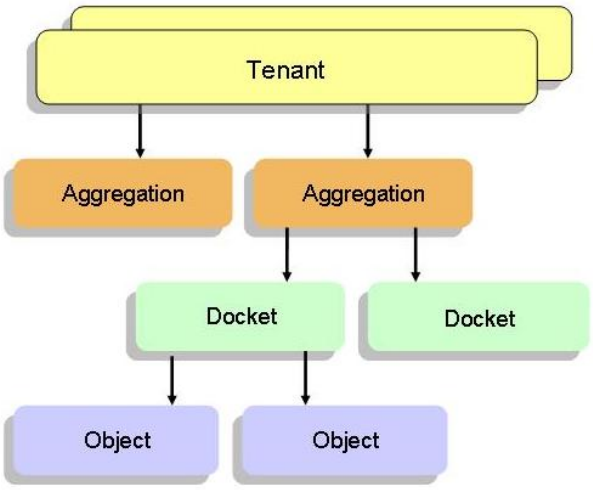

Figure 2: Data model in PDS Cloud

Tenant is an enterprise or organization that engages in storing data in the cloud. Each tenant constitutes an independent information domain, having separate administrative ownership, policies and users. Data assets belonging to different tenants are logically insulated from each other.

Aggregation is a configuration profile, defining the policies and capabilities for managing the data in storage. It specifies the details of one or more cloud platforms (address, access credentials, etc.) that are being used for physical storage. It also designates various characteristics for maintaining and accessing data objects, such as integrity checking procedures or rendering properties, as relevant for the specific use case. Each aggregation belongs to a single tenant, and its configuration is tailored to the tenant's requirement and regulations.

Docket is a grouping of objects analogous to a directory in a file system. A docket name is not unique, and may be reused under different aggregations.

Object is the fundamental preserved entity. In the context of OAIS-based preservation, this refers to an AIP. A given object belongs in a single docket and a single aggregation. An object is replicated in all the cloud platforms configured by the aggregation in which it belongs. An object has a name (as specified in the hierarchical path) and a *Logical Id*. Each Logical Id is globally unique. When an object is moved to a different aggregation or docket, its Logical Id remains the same. This can be exploited for maintaining referential integrity.

Aggregations are configured based on the needs of the tenant. The objects that belong in a given aggregation can be viewed as a collection of information assets that share the same characteristics and are managed together and in the same fashion. In that sense, aggregations can be considered as classes of service.

Dockets and objects are logical entities. A docket is distinct from a cloud container (also called bucket in some platforms), whereas a container has physical existence. The mapping between logical docket and physical container need not be one-to-one. Further, it does not have to be universal, and may be tailored for the requirements and limitations of each individual cloud platform. Similarly, an object in the naming hierarchy may or may not correspond to a single stored object in the cloud (for example, depending on object size limits). Details of the physical organization could be specified in the aggregation.

The data model can be extended to allow object names to be non-unique. Multiple objects having the same name and belonging in the same docket are considered versions of the same base object, and are identifiable and distinguishable by a globally unique *Version Id*. In PDS Cloud, versions can be used to represent transformations of the base AIP, necessitated by evolving preservation conditions. All versions of a base object share the same Logical Id. Version Ids may induce total order among versions. Referencing an object by name defaults to one specific version, typically the last (most recent). Further, each object may be associated with a *Parent Id*, denoting the Version Id of the object from which the given version is derived, thereby inducing a genealogy tree structure on all versions of a base object.

Users access the data without being aware of configuration details in the aggregation. A storage service layer, such as PDS Cloud, is responsible for interpreting the aggregation profile and engaging the relevant data management facilities. This includes accessing the specific cloud platforms designated by the aggregation and mapping the logical dockets and objects to the physical name space of each specific cloud. Changes in aggregation configuration over time affect the handling in the storage service layer, but remain transparent to the user application interface.

4.2 Example Use Case

We illustrate our approach by means of an example use case from the health care domain. See Figure 3.

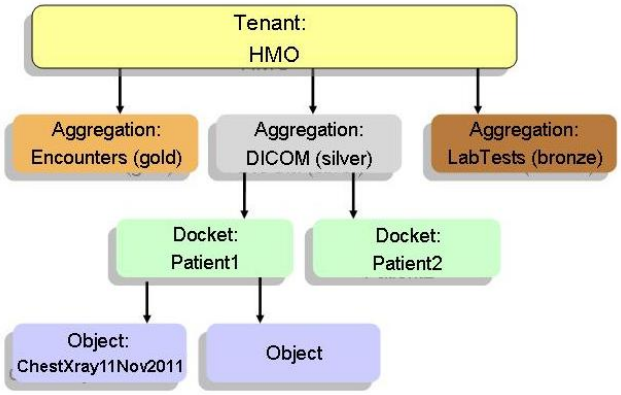

Figure 3: Example application of the PDS Cloud data model

Let us consider a tenant called HMO, a health care provider that is required to preserve medical records of patients according to elaborate government regulations, for long periods of time that may extend beyond the lifetime of the patient.

A separate aggregation is defined for each type of medical record. The aggregation embodies the preservation properties of the given type of medical record. For example, there could be an aggregation called "Encounters", intended for textual summaries of patient visits signed by the physician, which must have strong ("gold") preservation characteristics. The Encounters aggregation is configured to use two storage cloud platforms: Amazon S3 and Openstack Swift, and two fixity modules: SHAH-1 and MD5, applied semi-annually. Another aggregation, called "DICOM", is used for medical images with large raw image data and weaker ("silver") preservation requirements. Hence, the DICOM aggregation is configured to use two storage clouds, S3 and Swift, but only a single fixity module, MD5, applied annually. A third aggregation, called "LabTests" is used for laboratory results, with even weaker ("bronze") requirements, using a single storage cloud: S3, and a single fixity module, MD5, applied annually. As can be observed, the three aggregations provide different classes of preservation service.

In this example, a docket is a grouping of medical records of a specific patient. The docket name is the patient id. Considering there are two patients: Patient1 and Patient2, there would be a docket called Patient1 and a docket called Patient2 under each relevant aggregation. The type of each record (object) is determined by the aggregation.

Let a specific DICOM image of Patient1 be named chest-xray-11Nov2011. The hierarchical naming path of this object is /HMO/DICOM/Patient1/chest-xray-11Nov2011. Preservation policies applicable to the object, and specifically the multi-cloud configuration, are derived from the aggregation DICOM. The user need not be aware of the physical locations of the object. The object may have several versions, resulting from transformations due to changes in the rendering technology. The hierarchical path references by default the most recent version. Other versions are accessible using a specific Version Id.

5. MAPPING TO CLOUD STORAGE

PDS Cloud provides a brokering service that intermediates between OAIS entities and the diverse storage clouds. On the client side, PDS Cloud exposes a logical hierarchical data model that comprises tenants, aggregations, dockets and AIPs, as discussed in Section 4. On the storage side, it leverages multiple cloud platforms that typically use a hierarchical model consisting of of users, containers (or buckets) and objects with key-value pairs (metadata).

The Cloud Mapping Handler relates the two models. Tenants are mapped to users; dockets are mapped to cloud containers. Specifically in case of Amazon S3, the container name encodes the tenant name, geographical location, and docket name. This is because in Amazon the container name needs to be unique across all users and geographical locations. The aggregation name is represented as metadata (key-value pair) associated with cloud objects that implements each AIP. Mapping the AIP, the basic artifact in archival storage, is more involved since an AIP is a composite logical object with multiple sub-parts. We first describe the structure of the AIP, then the mapping to the cloud.

5.1 AIP Logical Structure

The logical structure of the AIP as defined in the OAIS standard [1] is illustrated in Figure 4. The AIP contains zero or one Content Information compartments and one or more Preservation Description Information (PDI) compartments.

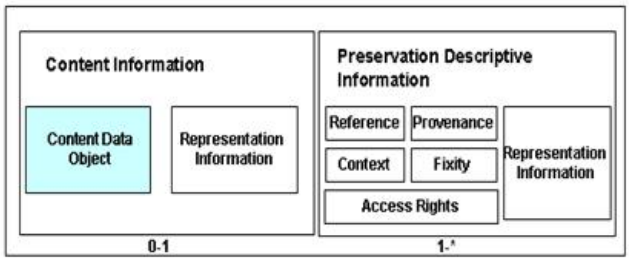

Figure 4: OAIS AIP logical structure

More specifically, Content Information comprises the Content Data Object (CDO) and the Representation

Information (RepInfo). CDO is the raw data being the focus of the preservation. RepInfo is information on the hardware and software environment needed to render the CDO intelligible to its designated community.

The PDI compartment maintains metadata describing the past and present states of the Content Information, covering five aspects: Identifying the AIP uniquely and permanently (reference); documenting its history and origin (provenance); describing the relationship to environment (context); ensuring bits have not been altered in an undocumented manner (fixity); and specifying access restriction (access rights).

5.2 AIP Mapping

The AIP logical structure reveals a composite entity with multiple integral sub-parts. If any sub-part is missing, access to the entire AIP might be lost. Thus, all the sub-parts should best be co-located in storage. This also simplifies migration of the AIP across storage services.

The AIP is materialized by the following physical organization. At the root is a manifest, which references all the other sub-parts. The manifest also contains the AIP metadata. Sub-parts include one or more CDOs, and optional metadata sub-parts in case the manifest cannot contain all the required metadata.

To maintain co-location of all the physical sub-parts we have considered two approaches. The first approach is to create a tarball of all the AIP sub-parts and store it as a single cloud object. Key-value pairs of this cloud object are utilized to keep frequently accessed small-sized metadata, such as AIP identifiers, aggregation name, and fixity modules and values. The advantage of this approach is that it spares PDS Cloud from having to synchronize among the various sub-parts (referential integrity). The disadvantage is that usability and performance are compromised, as clients need to extract the tarball when accessing the preserved data. Moreover, clients may need to download the entire tarball to retrieve just the metadata; this, however, is viewed as an infrequent situation because the most needed metadata is kept in key-value pairs of the cloud object, which are accessible separately. It should be noted that typically it is not feasible to save all the AIP metadata in key-value pairs because of imposed size limits. Also, in some clouds (e.g., Amazon S3), in order to update just the key-value pairs, the entire object must be uploaded again. We consider this a limitation of those clouds that is expected to be relaxed in the future.

In the second approach, each AIP sub-part is mapped to a separate cloud object, such that the object names of all sub-parts share the same prefix, thereby indicating their inclusion in the same AIP. In this case, the frequently used AIP metadata is kept in key-value pairs of the manifest object. The main advantage of this approach is that each sub-part can be stored and accessed in its native format, and only the relevant sub-parts needs to be downloaded. The disadvantage is that PDS Cloud has to synchronize among the various sub-parts and maintain referential integrity, whereas the cloud is not aware of the connection between the sub-parts.

While the second approach is probably more flexible and efficient, it is harder to implement because of the need to maintain multiple synchronized cloud objects. Thus, in PDS Cloud we have adopted the first approach. It is anticipated that in the future some cloud platforms will support compound objects, i.e. sets of cloud objects treated as one entity with internal referential integrity.

5.3 RepInfo AIP

Representation Information (RepInfo) is an important class of preservation metadata, describing how to interpret the content data. An AIP may reference multiple RepInfos; this allows several view paths to interpret the same data.

In PDS Cloud, RepInfo is kept as a separate AIP, with its own unique identifier. Thus, RepInfo can be used as a shared resource, referenced from multiple data AIPs.

An important example is the RepInfo for a virtual appliance (VA). This RepInfo contains all the information needed to instantiate a certain VA in the compute cloud. As any AIP, the VA RepInfo AIP is kept in the storage cloud, and is referenced by relevant data AIPs (typically all AIPs of a given aggregation). This is used by PDS Cloud to provide access to the data by means of a VA (see Section 6).

RepInfo is a recursive entity and may have one or more additional RepInfos to interpret itself. This creates a network of RepInfos that ends when facing a RepInfo that is non-digital, which the designated community is able to trustfully preserve through time.

By representing RepInfo as an AIP, we can utilize the same mechanism to preserve also the RepInfo, keeping the design simple.

5.4 AIP Versions

OAIS defines the concept of *version AIP*, which is an AIP resulting from a digital migration that involves transformation of an existing AIP, namely causing changes to either the Content Information bits or the PDI bits. The new AIP has a new identifier and metadata, and is viewed as a replacement of the source AIP, where the information has been preserved to the maximum extent practical. The PDI needs to identify the source AIP and its version, and document what changes were made and why. Typically, the previous version is also kept, e.g. for copyright or legal reasons. Additionally, several copies of each version AIP may be kept, all of which are bitwise identical. This is to improve the bit preservation of the data and allow

automatic fixes to damaged parts of the data.

In PDS Cloud, multiple AIPs having the same name and belonging in the same docket are considered versions of the same base AIP, and are identifiable and distinguishable by a globally unique persistent *Version Id*. All versions of a base AIP share the same *AIP Name* and *Logical Id*. Further, each AIP may be associated with a *Parent Id*, denoting the Version Id of the AIP from which the given version is derived, thereby inducing a genealogy tree structure on all versions of the base AIP.

PDS Cloud stores the different versions of the base AIP in one directory under the associated cloud container. The genealogy trees are kept in the cloud via key-value pairs of the version AIP objects, enabling clients to query and navigate the trees.

6. VIRTUAL APPLIANCES

Typically, current systems for long term digital data preservation provide a simple, but not very friendly access path to the stored data. Users are expected to download the data from the storage server and then examine it within their local environment. PDS Cloud enhances future access to preserved content by leveraging a storage/compute cloud synergy to automate the provisioning of virtual appliances running on a compute cloud with data objects preserved in the storage cloud.

6.1 Data Rendering and Transformation

A virtual appliance (VA) is a virtual machine image (VM image) with an operating system and application packaged together as a pre-installed system image for a virtualized environment such as KVM, Xen or VMware. The user runs the VA on a compute cloud, e.g. Amazon EC2, Rackspace Cloud Servers or Openstack Nova. In order to make a VA useful it is necessary to provision its running instances with the user's unique data. When this data is stored in cloud storage, as is the case with PDS Cloud, the user provisions the running instances by copying the data from cloud storage to the VA. After copying the data, the VA application operates on the data and presents the results to the user.

Having separate steps in which the user is required to interact with compute and storage clouds is both inconvenient and inefficient. PDS Cloud addresses these deficiencies by combining the compute and storage cloud related operations and providing methods to optimize the storage access.

PDS Cloud automates the provisioning of VAs running on a compute cloud with its data objects preserved in the storage cloud. The user interacts only with the data preservation storage system, using a single transaction, rather than issuing multiple transactions to the compute and storage clouds.

Instead of the user/application having to execute multiple steps:

1. Access the storage cloud;
2. Retrieve data from storage cloud;
3. Access the VA on compute cloud;
4. Store data in the VA;

PDS Cloud automates these steps into a single action. This simplifies the application, yields better performance and improves robustness.

This enhancement is analogous to when an OS supports double clicking on a file and the OS then automatically opens a window with the file inside its associated application. However, in our case the data, application, and the application's running environment reside in the cloud and not on the user's desktop.

Moreover, PDS Cloud can exploit its knowledge of the topology of the compute and storage cloud to collocate the VA instances with the data that it must process or bring the VA closer to the user. Further, the method may be extended to add filtering capabilities, like data security, not provided by the underlying clouds.

In providing the solution described above, PDS Cloud leverages several innovative features.

- A generic interface is provided, that hides the complexity of interacting with multiple heterogeneous compute and storage clouds.
- Cloud elasticity is exploited by creating many instances of the same VA, collocating the VAs with data to be transformed, and transforming large volumes of data in parallel. Typically, data transformation is done on a large volume of data of the same format. As the volume of data to be transformed increases, optimization related to and collocating compute and storage clouds becomes more important. Collocation is accomplished by creating VA instances in the same region as the cloud storage of the cloud provider.
- When allowing VA direct access to the storage cloud (see "VA Copy" below), the system can dramatically reduce data transfer time by collocating the VA instance and the subject data. The system can also enhance the underlying storage cloud's authorization and security model to ensure that the VA user(s) access to storage is restricted to only the data that it is permitted to read or write.
- The system automatically transforms data when the cloud storage data organization or format is not compatible with the VA. For instance, Amazon S3 organizes its data with buckets, objects and folders. The typical VA operating system is organized as files and directories. But copying a folder directly from S3 to a VA does not create a directory; rather the engine creates directories itself and reorganizes the other storage objects accordingly. When writing files and directories back to S3 the conversion process must be

reversed. The problem is further complicated since there is no single standard organization for data in the many cloud storage vendors so the system must treat each storage cloud's separately. Likewise, the system transforms the data formats when the cloud storage data is not compatible with the format required by VA application.

6.2 VA Provisioning Automation

A VA is ingested into the system as an OAIS AIP designated with the role of "Representation Information" (RepInfo), containing a description of the Virtual Appliance. Data AIPs can be associated with a RepInfo VA AIP. When users request access to the data, the system transparently provisions the AIPs on the VA.

PDS Cloud provides a generic interface that hides the complexity of interacting with multiple heterogeneous compute and storage clouds. The user sends a request to render data that is stored in cloud storage. The data may be tagged with the preferred VA, or the user may specify a particular VA.

The system automates provisioning as follows.

1. Create the VA instance on a compute cloud.
2. Invoke predefined initialization commands or scripts on VA.
3. Copy the data from the storage cloud to the VA.
4. Invoke predefined conclusion commands or scripts on VA.
5. Return a link to the VA to user.

In step 3 above, several data copy methods are provided: *PDS Cloud Copy*, *VA Copy*, and *Alternate Service Copy*, illustrated in Figures 5, 6, and 7.

The *PDS Cloud Copy* method is simple but less efficient. PDS Cloud copies the subject data from the storage cloud to PDS Clouds's local store and then pushes it to the VA on the compute cloud. The advantage is that PDS Cloud is able to transform the data before pushing the transformed data to the VA. For example, to comply with security regulations, data anonymization [19] may be required, which can be performed in PDS Cloud before releasing the data to a VA running on the public cloud.

The *VA Copy* method is more efficient. The VA copies the subject data directly from the storage cloud to the VA on the compute cloud. Further optimization can be achieved by collocating the VA and data to reduce copy distance. Collocation is accomplished by creating VA instances in the same region as the cloud storage provider. If necessary, another method in the VA is invoked to transform the data before releasing it to the user.

The *Alternate Service Method* method is complex, since it relies on another service to upload the data. However, in some cases this is required, where the data

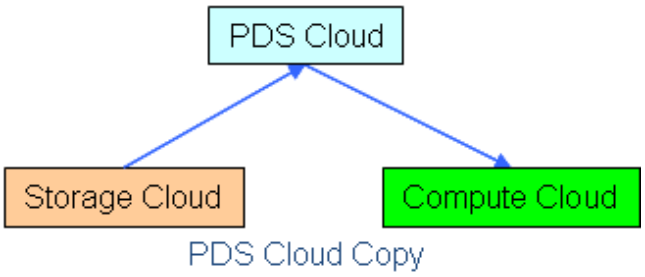

Figure 5: PDS Cloud Copy: PDS Cloud copies data from the storage cloud and then pushes it on to the VA

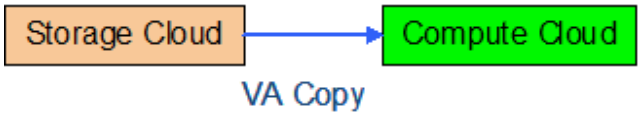

Figure 6: VA Copy: PDS Cloud invokes a predefined function in the VA to copy the data from the cloud storage to itself

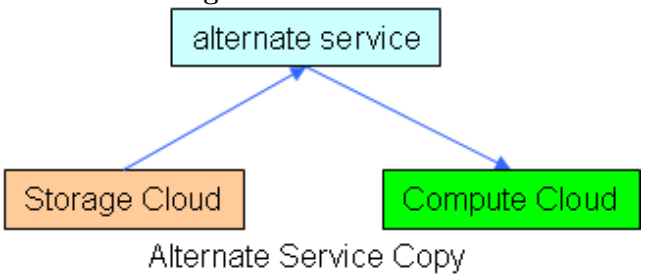

Figure 7: Alternate Service Copy: The system invokes a predefined function in the VA that calls an alternate service to upload data from the storage cloud to the VA

transformation methods are outside of PDS Cloud's immediate control. For example, the methods may be private, or may require a particular execution environment or access to an external database.

6.3 VA Security

PDS Cloud must guard against malicious intruders abusing the VAs, and legitimate users overreaching their privileges and accessing data that is not permitted to them. In order to guard against intruders, we limit the number of ports that will accept incoming traffic. Linux VAs are limited to only the ssh port 22. All data copy to a VA is done over ssh. Only password-less ssh with public keys access is supported and users are limited to non-privileged user ids. PDS Cloud does have a privileged id and uses it when copying data from cloud storage to the user's home with its *VA Copy* method.

Thus, intruders are kept away by password-less ssh and users are limited to their unprivileged user spaced. MS Windows VAs also open the RDP port 3389 for incoming traffic; it is less secure than password-less ssh only, but this is an unavoidable Windows limitation.

6.4 VA Provisioning Performance

Using the *VA Copy* method, PDS Cloud can dramatically reduce the time to transfer cloud data objects to a VA by collocating its VA instances and cloud storage. We did a *VA Copy* collocation study using AWS American S3 and E2.

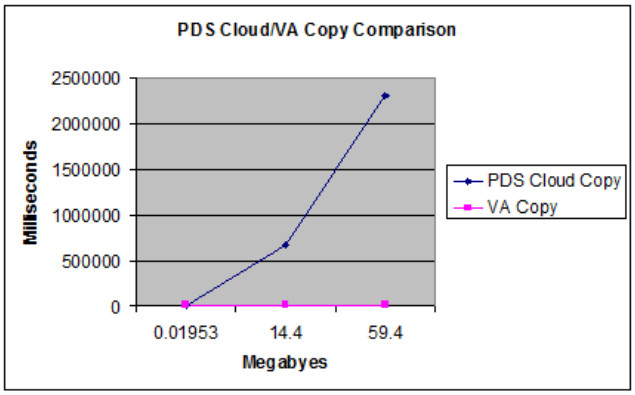

Figure 8: Comparison of VA Copy vs PDS Cloud Copy

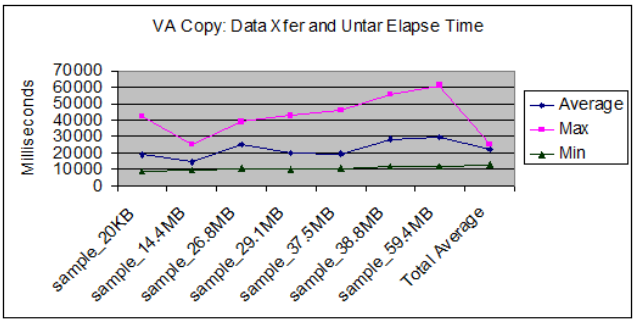

Figure 9: VA Copy: transfer and untar

In Figure 8 the *VA Copy* method is compared to the *PDS Cloud Copy* method. As would be expected, *VA Copy* transfers data much faster than *PDS Cloud Copy*. This is because *PDS Cloud Copy* copies the data twice, first from storage cloud to PDS Cloud, and then to the VA on the compute cloud, and data must traverse the distances between these locations. We also observe that as data size increases, *PDS Cloud Copy* exhibits an exponential increase in the time it takes to copy the data from the storage cloud to the VA.

A closer look at *VA Copy* performance exhibits some unexpected results. We experimented with a few tar files of different sizes ranging from 20KB to 60MB stored in AWS S3. Each file was copied to an Ubuntu VA and untared to the user's directory. It was expected that the time to copy-untar any given sample file would be more or less constant, and relative to its file size. However, we observed considerable variability when examining any given sample file's performance, and the relative performance between different samples is not always a function of file size. Figure 9 illustrates the samples and performance/size variability. For example, the 20KB sample, sample_20KB: performance ranged between 8845 and 42170 milliseconds; and the 59.4MB sample, sample_59.4MB: performance ranged between 11907 and 61295 milliseconds. We speculate that this variability is related to EC2 network congestion. Unexplained is the performance/size variability between samples sample_26.8MB, sample_29.1MB, and sample_37.5MB. Their relative completion times are the reverse of what one would expect based solely on size. However, the discrepancy in absolute terms is minimal, only a few milliseconds.

Collocating VA instances and cloud storage is not the only way that PDS Cloud can reduce VA provision time. It can also re-use VA instances.

When a VA is first launched it must be copied to a cloud compute node and brought to running state. Once in running state, the VA is initialized and all its server daemons are started. A VA instance can only be provisioned when its ssh server daemon is actively listening for new ssh client connections.

We refer to a VA prior to being launched as a Cold VA, and to a VA instance with its server daemons listening for new connections as a Hot VA. A Hot VA may be suspended, so that it does not use any CPU cycles, and subsequently resumed to its former state. A suspended VA instance is called Warm VA.

Figure 10 compares the VA provisioning startup elapsed times. We observe that PDS Cloud can decrease provisioning time significantly by re-using Hot or Warm VAs. Our experimentation was done on AWS EC2. The average time in milliseconds to transition a Cold or Warm VA to running state is similar; for Cold VAs: 35246 and for Warm VAs: 31185. We reason that the similarity indicates that EC2 does not actually copy the VA Images to its compute nodes, rather it uses Copy-On-Write. Discovering Hot VAs already in running state takes on average 3486 milliseconds, which is about one tenth of the time for the transitioning of Cold or Warm VAs.

As stated earlier, transition to running state is not sufficient to begin provisioning, the VA's ssh server daemon must be actively listening for new ssh client connections. The average elapsed time in milliseconds to complete a client ssh connection after transitioning to or discovering a running VA varies; for Cold VAs: 27653, for Warm VAs: 4176, and for Hot VAs: 2093.

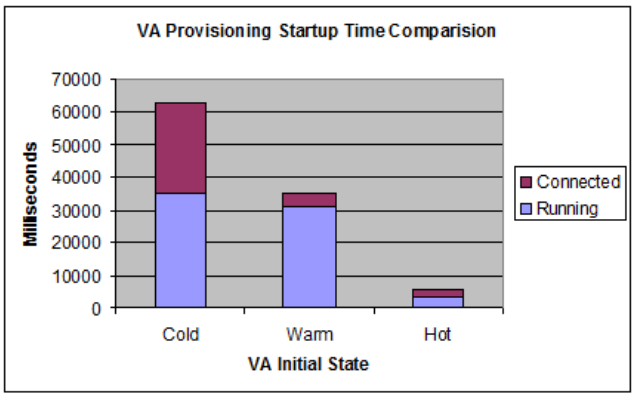

Figure 10: Comparison of VA provisioning startup time

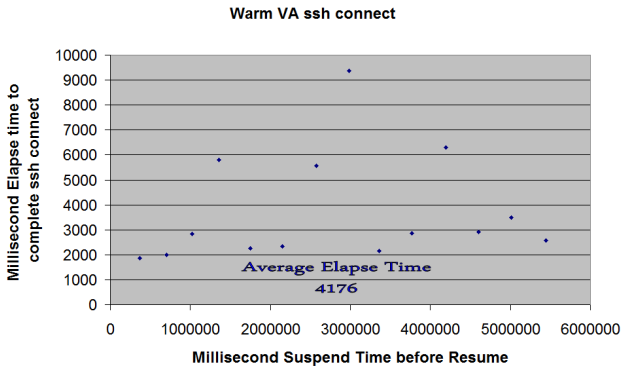

Figure 11: Warm VA: SSH connect elapse time

The ssh connection elapsed times for Cold and Hot VAs is rather constant, but Warm VAs exhibit considerable variability, as illustrated in Figure 11. We suggest that the reason that a Warm VA's transition to Hot VA takes significantly less time than a Cold VA's transition is that some of the Warm VA's state remains in memory, whereas a Cold VA starts with no state. Over time, the Warm VA's memory state is paged out to disk, but the page-out rate is a function of the load on the host. We believe that this explains the ssh connection time variability of Warm VAs.

7. DEMO OF PDS CLOUD PROTOTYPE

PDS Cloud is being developed as the storage infrastructure component of ENSURE [2], a European Union FP7 Project. In March 2012, a first prototype was demonstrated to the EU Commission as part of the annual project review.

The demonstration covered all the main features of PDS Cloud, including the hierarchical data model, configuring multiple aggregations, AIP ingest and access in the storage cloud, and deploying virtual appliances (VA) in the compute cloud.

Scenarios for the demo came from two use cases: health care and financial. In the health care use case, PDS Cloud demonstrated aggregations and versioning. Three aggregations were configured, for three types of medical records.

- *Medical Encounters:* designated as "gold" preservation service, with data saved on two storage clouds, and fixity check applied twice a year using two modules. Encounter records are XML documents.
- *Medical Images:* "silver" preservation service, using only one storage cloud, fixity check same as in "gold". Data consists of DICOM images.
- *Information Pages:* "bronze" service, using one storage cloud and one fixity module applied once yearly. These are PDF documents.

AIPs were ingested into the system in accordance with the three aggregation configurations. Simulating the passage of time and changes in requirements, DICOM images were transformed off-line, and the transformed images were ingested as new versions. PDS Clouds then provided access to both the original and transformed versions.

In the financial use case, we demonstrated access to preservation objects using VA. Two different aggregations were configured, for two types of financial records: Market Data and Customer Documents. To interpret Market Data in a business application context, a VA was built off-line from available components. The operating system is Microsoft Windows Server Amazon Machine Image (AMI). The designated rendering software installed in the VA is TradeStation [20], an application for analyzing and trading in the financial markets. The Cygwin SSH server was required to automate provisioning.

The VA was ingested into the system as a RepInfo (Representation Information) AIP associated with the Market Data aggregation. When the user requested to access some market data, PDS Cloud automatically copied the preserved data to the VA using its Service Copy method and returned an RDP(Remote Desktop Protocol)link file, which the user could connect to the Window Instance running on Amazons EC2.

8. RELATED WORK

A growing number of studies focus on the storage aspects of digital preservation. Long-term preservation systems differ from traditional storage applications with respect to goals, characteristics, threats, and requirements. Baker et al. [21, 22] examine these differences and suggest bit preservation guidelines and alternative architectural solutions that focus on replication across autonomous sites, reduced per-site engineering costs, and the ability to scale over time and technologies.

The requirements for preservation-aware storage are

characterized in a position paper by Factor et al. [23]. The PDS (Preservation DataStores) system [12, 18] was developed to these requirements. Our current work on PDS Cloud extends this methodology into the cloud environment.

Dappert and Enders [24] discuss the importance of metadata in a long term preservation solution. The authors identify several categories of metadata, including descriptive, preservation related, and structural, arguing that no single existing metadata schema accommodates the representation of all categories. The work surveys metadata specifications contributing to long-term preservation.

Offloading data maintenance functions from the application to the storage system is an ongoing trend. Functionality such as bit-to-bit data migration, block-level data integrity, and encryption are commonly carried out by advanced intelligent storage subsystems. For instance, Muniswamy-Reddy et al. [25] introduce protocols for the cloud that collect and track he provenance of data objects.

The DuraCloud open source platform [9] aims to provide a fully integrated environment where data can be managed across multiple cloud providers. DuraCloud is offered as a hosted service providing data storage, replication and access, and services to support data preservation, such as data format transformation and fixity checking. DuraCloud does not host content, but instead it stores only what is necessary to mediate storage and retrieval of content with third party storage providers.

OCLC's Digital Archive [26] provides a secure storage environment for master files and digital originals for libraries. It mainly provides bit preservation (backups, disaster recovery, and periodic fixity check) with some basic logical preservation (virus check, manifest and format verification). In [2] a comparison is made between this service and Amazon S3 cloud storage that does not include computational support. The comparison concluded that S3 with added preservation functionality is more expensive.

Bessani et. al. [27] present DepSky, a storage system in a cloud-of-clouds that overcomes the limitations of individual clouds by using an efficient set of Byzantine quorum system protocols, cryptography, secret sharing, erasure codes and the diversity that comes from using several clouds. One of the key objectives of DepSky is to reduce cost, and thus the DepSky protocols were designed to require at most two communication round-trips for each operation and store only approximately half of the data in each cloud for the typical case.

You et al. [28] present PRESIDIO, a scalable archival storage system that efficiently stores diverse data by providing a framework that detects similarity and selects from different space-reduction efficient storage methods.

DepSky and PRESIDIO suggest advanced protocols and implementations for specific data management functions. Our work, on the other hand, supports data management in a dynamic way, namely enabling protocols, technologies and implementation to change over time, and effecting the ensuing changes in a manner transparent to the user.

The goal of long term preservation is not only to maintain the bit integrity of the data, but also to guarantee that information content is understandable in the future. Emulation techniques for interpretation and visualization of digital contents have been used to assist in logical preservation [29, 30, 31, 32, 33, 34]. Yet, developing and maintaining emulation tools in each environment and for each information type may be too expensive as a common solution for long term preservation. Our approach advocates using virtual appliances in the compute cloud in place of specialized emulation systems.

Rhea et al. [35] describe an archival storage layer which uses content-addressed storage (CAS) to retain nightly snapshots of usersŠ disks indefinitely. In addition, they also snapshot the entire virtualized desktop. Their focus is on efficiently leveraging off-the-shelf storage to create a personalized archive. Arguably, preserving the entire virtualized desktop does not scale or age well. Rather, we claim that it is better to minimize the virtualized environment as much as possible by using "Just Enough OS" with only the required applications installed.

An extensive bibliography of digital preservation and curation has been compiled by Bailey [36]. It covers multiple subjects, including models. formats, an ongoing projects and research.

9. CONCLUSIONS AND FUTURE WORK

We have presented PDS Cloud, an OAIS-based preservation aware storage service that engages storage and compute clouds from diverse providers. The main objective of PDS Cloud is to maintain understandability of the digital content (logical preservation) for the long term, adhering to dynamic changes in requirements and evolving technology. The paper discussed the architecture of PDS Cloud and its novel features, in particular the data model, the mapping of AIP to the storage cloud, and the deployment and data provisioning of virtual appliances. The data model enables organizing information into multiple aggregations with separate preservation profiles that can be changed dynamically and transparently. Virtual appliances deployed in the compute cloud automate the interpretation of preservation data, as alternative to specialized local emulation. Further, we presented performance measurements of virtual appliances with data provisioning in the AWS

cloud. For data survivability in multiple clouds measurements, we reference the analytical model proposed by Li et al. [13].

PDS cloud is currently being developed as the storage infrastructure of a larger experimental runtime environment called ENSURE, modeled after OAIS, and has been demonstrated to the EU Commission. Future plans include providing advanced preservation functions. In particular, we intend to support storlets and SIRF (Self-contained Information Retention Format), as discussed in Section 3.2. We will also conduct further performance analysis in order to gain more insight into the tangible benefits of the cloud as a preservation environment.

Digital preservation systems need to perform periodical data intensive tasks, such as validation and data transformation. Instead of moving the data to a compute environment and then back to cloud storage, it is desirable to perform such data intensive procedures within the storage system, or as close as possible to it (i.e., in the same server or in the same network domain). The term storlets (similar to applets and servlets) describes deployable restricted computation modules, embedded in the cloud storage system. Storlets for purposes such as AIP transformation, data mining, or fixity computation, would be deployed by PDS Cloud and later executed either periodically or when explicitly triggered. Obviously, supporting storlets entails enhancements to the storage cloud platform. We plan to initially experiment with this feature on a public platform where the implementation is readily available and extendable, such as Openstack Swift.

Efforts to unify cloud APIs across the multitude of vendors and to cater to diverse application requirements have lead to the specification of CDMI (Cloud Data Management Interface) [15], a SNIA architecture standard. This is functional interface with which clients can store and access objects in the cloud, discover the platform capabilities, manage containers and objects, including setting of metadata, and administer accounts, security, billing, etc. In anticipation of increased popularity of this rapidly evolving standard, we intend to leverage CDMI on both ends of the PDS Cloud architecture: as a client interface into PDS cloud, and as a driver underneath the jclouds multi-cloud service to access vendor cloud platforms such as Openstack Swift. Moreover, CDMI will enable us to deploy cloud platforms that expose only this interface, such as VISION Cloud. A CDMI driver for jclouds would be a meaningful contribution to this open source library.

Another issue that merits exploration is referential integrity. In a preservation system, there are references and links within the various parts of the AIP and between distinct AIPs, e.g., a reference from an AIP to its RepInfo (representation information). Furthermore, multi-cloud support implies that AIPs may reference each other even if they do not reside on the same cloud. In cloud storage, objects exist within containers. Keeping related object in the same container can be a partial solution, to assist referencing within a single cloud. This scheme would need to be extended to support referencing objects on remote clouds, for example using appropriate metadata.

ACKNOWLEDGMENT

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement № 270000.

REFERENCES

[1] *Reference Model for an Open Archival Information System (OAIS) - Recommended Practice, CCSDS 650.0-M-2 (Magenta Book) Issue 2.* Also available as ISO Standard 14721:2012. The Consultative Committee for Space Data Systems (CCSDS), June 2012. URL http://public.ccsds.org/publications/archive/650x0m2.pdf.

[2] ENSURE: Enabling kNowledge Sustainability, Usability and Recovery for Economic value, EU FP7 project. URL http://ensure-fp7.eu.

[3] Amazon Web Services. URL http://aws.amazon.com.

[4] Openstack cloud software. URL http://openstack.org.

[5] E.K. Kolodner, S. Tal, D. Kyriazis, D. Naor, M. Allalouf, L. Bonelli, P. Brand, A. Eckert, Elmroth E, S.V. Gogouvitis, Harnik D, F. Hernandez, M.C. Jaeger, E.B.Lakew, J.M. Lopez, M. Lorenz, A. Messina, A. Shulman-Peleg, R. Talyansky, A. Voulodimos, and Y. Wolfsthal. A cloud environment for data-intensive storage services. In *CloudCom 2011: Proceedings of the IEEE Third International Conference on Cloud Computing Technology and Science*, pages 357–366, Athens, Greece, November 2011. URL http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6133164.

[6] Eucalyptus. URL http://www.eucalyptus.com.

[7] The Rackspace Cloud. URL http://www.rackspace.com/cloud.

[8] EMC Atmos. URL http://www.emc.com/storage/atmos/atmos.

[9] DuraCloud. URL http://www.duracloud.org.

[10] *Reference Model for an Open Archival Information System (OAIS) - Draft Recommended Standard, CCSDS 650.0-P-1.1 (Pink Book) Issue 1.1.* The Consultative Committee for Space Data Systems (CCSDS), August 2009. URL http://public.ccsds.org/sites/cwe/rids/Lists/CCSDS6500P11/CCSDSAgency.aspx.

[11] L. Richardrson and S. Ruby. *RESTful Web Services*. O'Reilly Media, 2007. URL http://shop.oreilly.com/product/9780596529260.do.

[12] S. Rabinovici-Cohen, M. Factor, D. Naor, L. Ramati, P. Reshef, S. Ronen, J. Satran, and D. Giaretta. Preservation DataStores: New storage paradigm for preservation environments. *IBM Journal of Research and Development, Special Issue on Storage Technologies and Systems*, 52(4/5):389–399, July/September 2008. URL http://www.research.ibm.com/haifa/projects/storage/datastores/papers/rabinovici.pdf.

[13] Y. Li, D.D.E. Long, and E.L. Miller. Understanding data survivability in archival storage systems. In *SYSTOR 2012: Proceedings of the 5th Annual International Systems and Storage Conference*, Haifa, Israel, June 2012. URL http://www.soe.ucsc.edu/~yanli/res/li-systor12.pdf.

[14] P. Mell and T. Grance. The NIST definition of cloud computing: Recommendations of the National Institute of Standards and Technology. Special Publication 800-145, National Institure of Standards and Technology, U.S. Department of Commerce, September 2011. URL http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf.

[15] *Cloud Data Management Interface (CDMI), Version 1.0.2, Technical Position*. Storage Networking Industry Association (SNIA), June 2012. URL http://cdmi.sniacloud.com.

[16] Jclouds. URL http://www.jclouds.org.

[17] S. Rabinovici-Cohen, M.G. Baker, R. Cummings, S. Fineberg, and J. Marberg. Towards SIRF: Self-contained Information Retention Format. In *SYSTOR 2011: Proceedings of the 4th Annual International Systems and Storage Conference*, Haifa, Israel, May 2011. URL http://www.research.ibm.com/haifa/projects/storage/datastores/papers/systor56-rabinovici-cohen.pdf.

[18] M. Factor, D. Naor, S. Rabinovici-Cohen, L. Ramati, P. Reshef, J. Satran, and D. Giaretta. Preservation DataStores: Architecture for preservation aware storage. In *MSST 2007: Proceedings of the 24th IEEE Conference on Mass Storage Systems and Technologies*, pages 3–15, San Diego, CA, September 2007. URL http://www.haifa.il.ibm.com/projects/storage/datastores/papers/Preservation_DataStores_MSST07_camera.pdf.

[19] L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):571–588, October 2002. URL http://www.comp.nus.edu.sg/~tankl/cs5322/readings/k-anonymity2.pdf.

[20] Tradestation. URL http://www.tradestation.com.

[21] M. Baker, K. Keeton, and S. Martin. Why traditional storage systems don't help us save stuff forever. In *HotDep 2005: Proceedings of the IEEE First Workshop on Hot Topics in System Dependability*, Yokohama, Japan, June 2005. URL http://www.hotdep.org/2005/baker_forever.pdf.

[22] M. Baker, M. Shah, D. Rosenthak, M. Roussopoulos, P. Maniatis, T.J. Giuli, and P. Bungale. A fresh look at the reliability of long-term digital storage. In *EuroSys 2006: Proceedings of the 1st ACM SIGOPS European Systems Conference*, pages 221–234, April 2006. URL http://dl.acm.org/citation.cfm?id=1141770.

[23] M. Factor, D. Naor, S. Rabinovici-Cohen, L. Ramati, P. Reshef, and J. Satran. The need for preservation aware storage - a position paper. *ACM SIGOPS Operating Systems Review, Special Issue on File and Storage Systems*, 41(1):19–23, January 2007. URL http://www.research.ibm.com/haifa/projects/storage/datastores/papers/preservation_data_store_osr07_dec_30.pdf.

[24] A. Dappert and M. Enders. Digital perservation metadata standards. *Information Standards Quarterly, Special Issue on DIgital Preservation*, 22(2):4–12, Spring 2010. URL http://www.loc.gov/standards/premis/FE_Dappert_Enders_MetadataStds_isqv22no2.pdf.

[25] K-K Muniswamy-Reddy, P. Macko, and M.I. Seltzer. Provenance for the cloud. In *FAST 2010: Proceedings of the 8th USENIX Conference on File and Storage Technologies*, pages 197–210, San Jose, CA, February 2010. URL http://www.usenix.org/events/fast10/tech/full_papers/muniswamy-reddy.pdf.

[26] Digital Archive. URL http://www.oclc.org/us/en/digitalarchive.

[27] A. Bessani, M. Correia, B. Quaresma, F. Andre, and P. Sousa. Depsky: Dependable and secure

storage in a cloud-of-clouds. In *EuroSys'11: Proceedings of the 6th ACM EuroSys Conference on Computer Systems*, pages 31–46, Salzburg, Austria, April 2011. URL http://www.gsd.inesc-id.pt/~mpc/pubs/eurosys219-bessani.pdf.

[28] L.L. You, K.T. Pollack, D.D.E. Long, and K. Gopinath. PRESIDIO: A framework for efficient archival data storage. *ACM Transactions on Storage*, 7(2), July 2011. URL http://doi.acm.org/10.1145/1970348.1970351.

[29] J.R. Van der Hoeven, H.N. van Wijngaarden, R. Verdegem, and J. Slats. Emulation – a viable preservation strategy. Technical report, Koninklijke Bibliotheek / Nationaal Archief, The Hague, Netherlands, May 2005. URL www.kb.nl/hrd/dd/dd_projecten/Emulation_research_KB_NA_2005.pdf.

[30] J.R. Van der Hoeven and H.N. van Wijngaarden. Modular emulation as a long-term preservation strategy for digital objects. In *IWAW'05: Proceedings of the 5th International Web Archiving Workshop*, Vienna, Austria, May 2005. URL http://iwaw.europarchive.org/05/papers/iwaw05-hoeven.pdf.

[31] R.A. Lorie. A methodology and system for preserving digital data. In *JCDL 2002: Proceedings of the 2nd ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 312–319, Portland, OR, July 2002. URL http://doi.acm.org/10.1145/544220.544296.

[32] R.A. Lorie. The UVC: A method for preserving digital documents: Proof of concept. Technical report, Koninklijke Bibliotheek / Nationaal Archief, The Hague, Netherlands, 2004. URL http://www.kb.nl/hrd/dd/dd_onderzoek/reports/4-uvc.pdf.

[33] T. Reichherzer and G. Brown. Quantifying software requirements for supporting archived office documents using emulation. In *JCDL 2006: Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 86–94, Chapel Hill, NC, June 2006. URL http://dl.acm.org/citation.cfm?doid=1141753.1141770.

[34] D. Von Suchodoletz. A future emulation and automation research agenda. In *Automation in Digital Preservation, Dagstuhl Seminar Proceedings 10291*, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2010. URL http://drops.dagstuhl.de/opus/volltexte/2010/2771.

[35] S. Rhea, R. Cox, and A. Pesterev. Fast, inexpensive content-addressed storage in foundation. In *USENIX'08: Proceedings of the USENIX 2008 Annual Technical Conference*, pages 143–156, Boston, MA, June 2008. URL http://static.usenix.org/events/usenix08/tech/full_papers/pucha/pucha.pdf.

[36] C.W. Bailey, Jr. Digital curation bibliography: Preservation and stewardship of scholarly works. Technical report, Digital Scholarship, Houston, TX, 2012. URL http://digital-scholarship.org/dcpb/dcb.htm.