PDS Cloud: Long Term Digital Preservation in the Cloud

Simona Rabinovici-Cohen
IBM Research – Haifa
Haifa 31905, Israel
simona@il.ibm.com

John Marberg
IBM Research – Haifa
Haifa 31905, Israel
marberg@il.ibm.com

Kenneth Nagin
IBM Research – Haifa
Haifa 31905, Israel
nagin@il.ibm.com.com

David Pease
IBM Research – Almaden
San Jose, CA 95120, USA
pease@almaden.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. 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 provisioned with cloud storage data objects together with their designated rendering software. PDS Cloud has a hierarchical data model supporting independent tenants whose assets are organized in multiple aggregations based on content and value. 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;**

I. 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 integrated with its user-defined and system-defined metadata as a single 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, revised in 2012 (ISO 14721:2012) [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 metadata needed in the preservation services. PDS Cloud serves as an enhanced archival storage that 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. 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 an 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.

II. 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. 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 at defined intervals. 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. Further, some clouds do not support update of metadata alone without also incurring the data transfer cost of its data.
- **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.
- **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, 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.

Notwithstanding these issues, cloud storage provides a scalable, sound and cost-effective infrastructure that is essential for preservation solutions. The selection of 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.

The PDS Cloud architecture presented in subsequent sections attempts to mitigate many of the gaps identified here, as will be pointed out in the discussion. Additional aspects of our comparative evaluation are presented in the expanded version of this work [3].

III. PDS Cloud Architecture

The gap analysis in Section II 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 [4], 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.
- Provide 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.
- 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.

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.

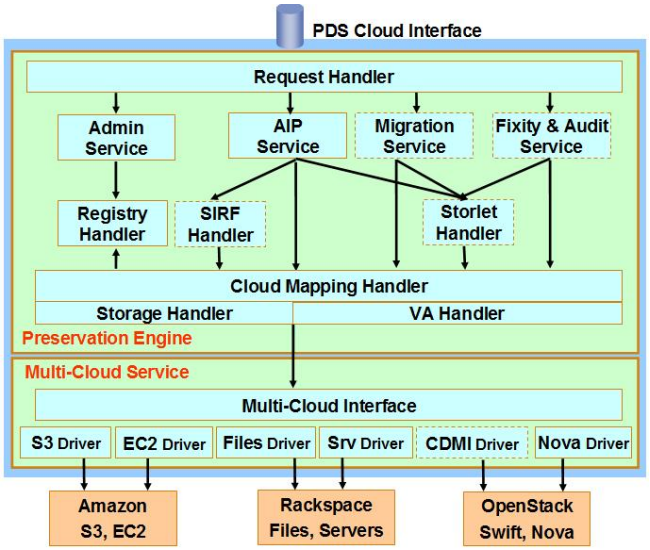

Figure 1. PDS Cloud high-level architecture

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, and to increase data survivability.

We assume user authentication and authorization to the preservation system are performed in higher levels of the runtime environment, prior to invoking 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.
- **Preservation Engine** – provides preservation functionality for AIPs. It accepts requests via the PDS Cloud external interface and services them, utilizing the multi-cloud service underneath.

A. Multi-Cloud Service

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.

PDS Cloud supports public and private clouds. 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 implementation, we are initially experimenting with Amazon [5] S3 storage and EC2 compute (public cloud), and Openstack [6] Swift storage and Nova compute (private cloud).

We leverage jclouds [7], 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.

B. 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 V). 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 [8] 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 IV).

Migration Service, which is intended for future implementation, supports logical preservation by coordinating transformations of AIPs from one format to another as requirements and data handling capabilities evolve.

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 II, 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 storage are called "storlets" [4]. 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 later will execute periodically or when explicitly triggered.

IV. Hierarchical Data Model

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. Also, as cloud technology advances, it should be possible to easily migrate data to another cloud platform and leverage new cloud services. Ideally, any changes in data management should be completely transparent to the user of the storage service.

The goal 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.

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.

The top-down hierarchy consists of: tenants, aggregations, dockets, and objects. This can be seen in Figure 2.

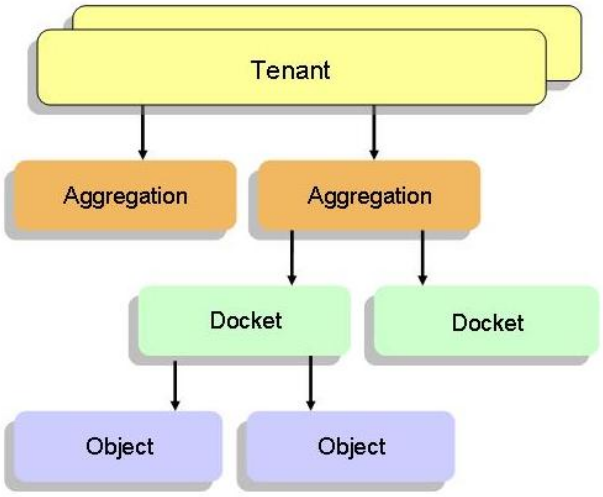

Figure 2. Data model in PDS Cloud

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, 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 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.

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.

V. 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 IV. On the storage side, it leverages multiple cloud platforms that typically use a hierarchical model consisting 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.

A. AIP Logical Structure

The logical structure of the AIP as defined in OAIS is illustrated in Figure 3. The AIP contains zero or one Content Information compartments and one or more Preservation Description Information (PDI) compartments.

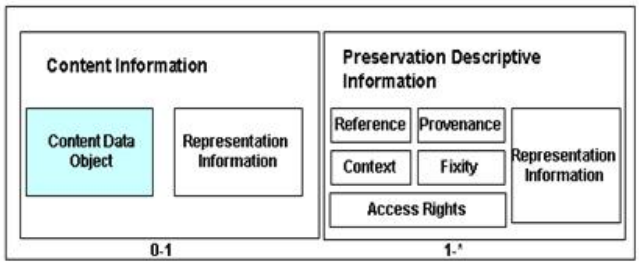

Figure 3. OAIS AIP logical structure

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).

B. 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, RepInfo reference, 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 need 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.

C. 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. This is used by PDS Cloud to provide access to the data by means of a VA (see Section VI).

D. 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.

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 various 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.

VI. 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.

A. 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, 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 does some operation on the data and presents the results to the user.

PDS Cloud automates the process of utilizing a VA in a compute cloud to render data stored in the storage cloud. Instead of the user/application executing multiple steps: (1) access the storage cloud; (2) retrieve data from storage cloud; (3) access the VA on compute cloud; (4) copy the data to the VA and possibly transform the data, the system automates it into a single action. This simplifies the application, yields better performance and improves robustness.

B. VA Provisioning Automation

A VA is ingested into the system as an OAIS AIP designated with the role of "Representation Information" (RepInfo). Data AIPs can be associated with a RepInfo VA AIP. When users request access to the data, the system transparently provisions the data 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 the cloud. The data may be tagged with a preferred VA, or the user may specify a particular VA.

Two data copy methods are provided: *PDS Cloud Copy*, and *VA Copy*.

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 storage, 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 may need to be anonymized.

The *VA Copy* method is more efficient. The VA copies the data directly from the storage cloud to the VA on the compute cloud. Further optimization can be achieved by collocating the VA and the subject data to reduce copy distance.

C. 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 EC2.

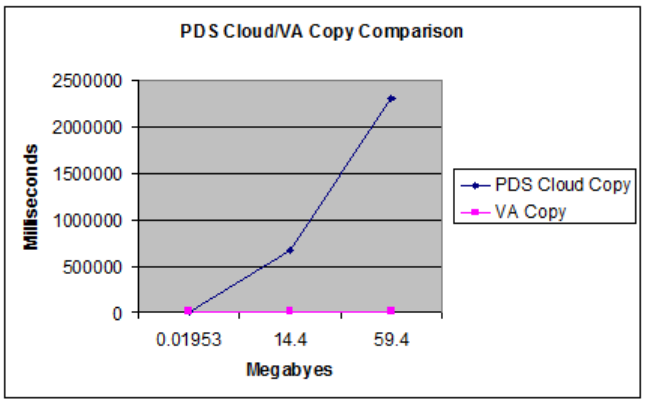

Figure 4. Comparison of VA Copy vs PDS Cloud Copy

In Figure 4 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 to PDS Cloud storage and then to the VA, and the data must traverse the distances between the locations. We also observe that as data size increases, *PDS Cloud Copy* exhibits an exponential increase in the time it takes to transfer the data.

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

When a VA is first launched it must be copied to a cloud compute node and brought to running state. Then it 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 whose server daemons are 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.

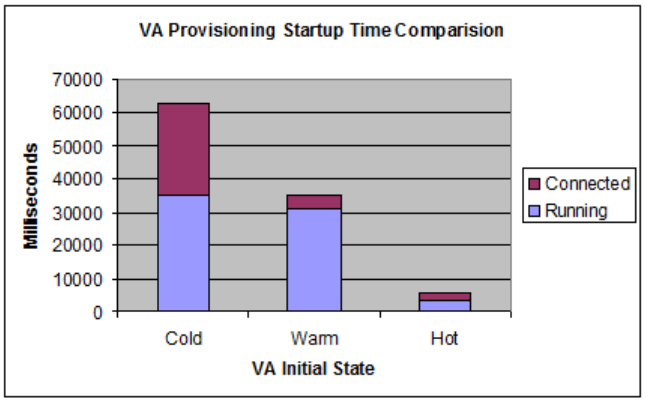

Figure 5. Comparison of VA provisioning startup time

Figure 5 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 to transition a Cold or Warm VA to running state is similar. We reason that the similarity indicates that EC2 does not actually copy the VA Image to the compute node, rather it uses Copy-On-Write. On the other hand, discovering a Hot VA already in running state takes on average only one tenth of this time. 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 to complete a client ssh connection after transitioning to or discovering a VA in running state varies, depending on the VA's initial state. We suggest that the reason 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 the Cold VA starts with no state.

More details of the performance analysis can be found in the expanded version of this work [3].

VII. 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 discusses the architecture of PDS Cloud and its novel features, in particular the data model, the mapping of AIP to the cloud, and the deployment and data provisioning of virtual appliances. 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 will continue the vision of offloading advanced preservation functions to the storage. In particular, we intend to implement a storlet engine for Openstack Swift to perform data intensive tasks locally within the storage, or as close as possible to it.

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.

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

[3] S. Rabinovici-Cohen, J. Marberg, and K. Nagin. Preservation DataStores in the Cloud (PDS Cloud): Long term digital preservation in the cloud. Technical Report H–0318, IBM Research – Haifa, January 2013.

[4] 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.

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

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

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

[8] 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.