Abstract: This paper sets out a model of these elements that describes a logical view of their functions and capabilities using a descriptive taxonomy. The purpose of this model is to form a basis upon which industry efforts can be organized, needed standards identified and vendor products can be described by vendor independent terminology.

This document has been released and approved by the SNIA. The SNIA believes that the ideas, methodologies, and technologies described in this document accurately represent the SNIA goals and are appropriate for widespread distribution. Suggestion for revision should be directed to http://www.snia.org/feedback/.
Publication of this SNIA Whitepaper has been approved by the SNIA. This document represents a stable proposal for use as agreed upon by the SNIA Technical Council. The SNIA does not endorse this proposal for any other purpose than the use described. This proposal may not represent the preferred mode, and the SNIA may update, replace, or release competing proposals at any time. If the intended audience for this release is a liaison standards body, the future support and revision of this proposal may be outside the control of the SNIA. Suggestion for revision should be directed to snia-tc@snia.org

USAGE

Copyright © 2022 SNIA. All rights reserved. All other trademarks or registered trademarks are the property of their respective owners.

The SNIA hereby grants permission for individuals to use this document for personal use only, and for corporations and other business entities to use this document for internal use only (including internal copying, distribution, and display) provided that:

1. Any text, diagram, chart, table or definition reproduced shall be reproduced in its entirety with no alteration, and,
2. Any document, printed or electronic, in which material from this document (or any portion hereof) is reproduced shall acknowledge the SNIA copyright on that material, and shall credit the SNIA for granting permission for its reuse.

Other than as explicitly provided above, you may not make any commercial use of this document, sell any excerpt or this entire document, or distribute this document to third parties. All rights not explicitly granted are expressly reserved to SNIA.

Permission to use this document for purposes other than those enumerated above may be requested by emailing tcmd@snia.org. Please include the identity of the requesting individual or company and a brief description of the purpose, nature, and scope of the requested use.

All code fragments, scripts, data tables, and sample code in this SNIA document are made available under the following license:

DISCLAIMER

The information contained in this publication is subject to change without notice. The SNIA makes no warranty of any kind with regard to this specification, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The SNIA shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this specification.

Suggestions for revisions should be directed to https://www.snia.org/feedback/.

Copyright © 2022 SNIA. All rights reserved. All other trademarks or registered trademarks are the property of their respective owners.
Abstract

Today's IT environment incorporates various products that are intended to store, protect, secure and make available the information used by businesses and business processes. These products encompass elements used in both the data and control path between the user and the locations of that information. Standards exist and are emerging for interoperability between these elements, however, what is missing is a comprehensive description of where interoperability is needed and where standards can best be applied.

Why use multiple clouds?

Risk reduction

Avoid lock-in

Applications that are built on technology or features only provided by a single cloud are tightly coupled to that cloud provider. Moving to another cloud requires development work that may be expensive or take a long time to complete. If features are not present on other clouds, that technology may need to be re-implemented within the application, or functionality that depends on that technology may need to be removed before moving to another cloud.

Supporting a storage abstraction layer that insulates the application from the underlying cloud provider’s interfaces (such as using a CDMI proxy server that communicates with multiple clouds, or by using CDMI proxies on each cloud) allows an application to be used with multiple clouds, thus reducing or eliminating the cost to exit any given cloud. This also ensures that applications are not tightly coupled to cloud-specific technologies.

Avoid single points of failure

Applications that are built on a single cloud are dependent on the reliability and availability of that cloud. By spanning more than one cloud, reliability, availability and performance can be increased.

Supporting a storage abstraction layer allows an application to store and access data across multiple clouds, and the placement of data can be transparently changed without having to change the application. This allows greater control over the cost vs. reliability, availability, and performance tradeoffs.
Political, regulatory and compliance restrictions

Regulatory, compliance and political constraints can change over time, which can result changes in which clouds are viable options for use.

![Figure 3: Avoid Regulatory Compliance Violations](image)

Supporting a storage abstraction layer allows data to be evacuated from, or moved from cloud to cloud transparently without requiring changes to the application. For example, data governed by different regulations can be placed in different clouds that are compliant with those regulations, and for data where cloud storage is not acceptable, can store data on-premises.

Cost reduction

Operational cost reduction

Applications that are built on a single cloud can only use service offerings made available by that cloud provider. By selecting services offered by multiple cloud providers, a lower cost service offering that meets the needs of the application may be available on one cloud that is not available on another cloud.

![Figure 4: Select lowest cost services](image)

Supporting a storage abstraction layer allows data to be placed on storage services that most closely meet the desired SLO, thus minimizing paying for SLO levels that are not required. Data can also be transparently moved from cloud service to cloud service if pricing and SLOs change over time (when egress costs are less than the potential savings).
Exit cost reduction

Applications that are built on a single cloud are unable to stop using that cloud. By supporting multiple clouds, an application can choose to migrate services to an alternate cloud, or choose to stop using services on a given cloud, without disrupting application operation.

Container orchestration solutions, such as Kubernetes, enable workload packaging and mobility between clouds. Multicloud storage abstraction layer will make such workload migrations even more transparent.

Supporting a storage abstraction layer allows data movement to evacuate (or archive) data stored on a given cloud to be transparent to the application. It also allows data movement to be sourced from the most cost-effective location.

Running work in parallel across clouds

Applications that are built on a single cloud have to process data within the same cloud where the data is stored. By supporting multiple clouds, data can be packaged and spread across multiple clouds to take advantage of computing costs, available computing resources and types of computing resources, including running processing in multiple clouds in parallel. However, this approach does not work well for large datasets where the cloud-resident application must access the full dataset.

Supporting a storage abstraction layer allows collections of data to be stored together on the same cloud, so that computing performed on that cloud can be performed against locally stored data. An abstraction layer can also provide transparent access to remotely stored data for computing that is running in any given cloud, and can transparently migrate data when computing moves from cloud to cloud.
Hybrid cloud / Burst to cloud enablement

Supporting a storage abstraction layer allows uniformal data access on-premises and in the cloud. This may ease burst to cloud when on-premises capacity is exhausted, or to allow remote batch execution using cloud resources.

![Diagram showing cloud A and cloud B with CDMI access or CDMI multi-cloud proxy](image)

Figure 7: Run workloads across on-prem as well as public clouds

Access to features/performance

Access to cloud-specific features

Applications that are built on a single cloud are limited to the specific features provided by that cloud. By supporting multiple clouds, features that are present only a given cloud can still be taken advantage of, without having to use that cloud exclusively.

![Diagram showing cloud A and cloud B with CDMI access or CDMI multi-cloud proxy](image)

Figure 8: Access features not supported by all clouds

Supporting a storage abstraction layer allows storage features specific to a cloud to be exposed in a standardized manner and allows data to be transparently accessed and migrated as needed in order to take advantage of cloud-specific features without the application being aware of the underlying mechanics.

Latency reduction

When storing the same file in multiple clouds, a simultaneous request can be made for the file from each of those clouds. The response will vary for each cloud, but the first response can be used to satisfy the request. The other requests can be aborted to save egress charges. This also can be hidden behind a proxy CDMI server such that applications need not do anything different.

![Diagram showing latency reduction](image)

Figure 8: Selectively access clouds based on latency
Throughput increase (parallelism)

Storage objects on-prem are accessible to multiple clouds. There’s hard limits to how much of that on-prem object can be processed by any given cloud at any given type. By processing objects on-prem in a parallel multi-cloud fashion those hard-limits can be exceeded and increase the aggregate total throughput.

![Diagram of multi-cloud setup](image)

Figure 9: Increase throughput across multiple clouds.

Storage objects can be sharded and erasure coded such that each shard is stored on a different cloud. Parallel requests to each cloud return the shards. When a sufficient number of shards are returned, the object can be reconstituted. This gets around any individual cloud provider’s bandwidth limits. Sharding also increases the security since no one cloud has all the data.

Egress cost reduction

Cloud providers have varying charges for egress. Some may not charge at all for egress. By using multiple clouds to store the data, the cloud with the cheapest egress charges can be used to fetch the object.

![Diagram of egress cost reduction](image)

Figure 10: Reduce cost by retrieving from lower cost cloud