Introducing the new Cloud Data Management Interface

Standardizing the Cloud for Interoperability

Mark A. Carlson
Sun and SNIA Technical Council
Chair, SNIA Cloud Storage TWG

Abstract

- The use of the term cloud in describing new models for storage and computing arose from architecture drawings that typically used a cloud as the dominant networking icon. The cloud conceptually represented any to any connectivity in a network, but also an abstraction of concerns such the actual connectivity and the services running in the network that accomplish that connectivity with little manual intervention.

Agenda

- Some background on cloud storage
- The Cloud Storage Reference Model
- CDMI – the interface
  - Data Objects
  - Containers
  - Accounts
  - Capabilities
- Cloud Storage TWG progress and roadmap

SNIA Cloud Storage TWG

- Launched April 2009
  - 116 Technical Work Group members (43 active)
  - Google group for broader community (198 members): http://groups.google.com/group/snia-cloud
- Published first documents June 2009
  - Use Cases/Requirements, Reference Model
  - Public web page http://snia.org/cloud
- Working on Cloud Data Management Interface
  - Targeted at ANSI and ISO certification
- Starting up a Cloud Storage Initiative
  - Promote Cloud Storage and Standards

Participating* Companies & Academia

- ActiFio
- Brocade
- Bycast
- Calsoft
- Cisco
- Data Mobility Group
- Dell
- Dimension Data
- EMC
- Emulex
- ETRI
- FourColour IT
- Fujitsu
- Hamburg University of Technology
- HCL Technologies
- HDS
- Hitachi
- HP
- Huawei
- IBM
- Intel
- Intellium
- Int'l Microsystems
- Kerstor
- Knowledge Transfer
- LB-Systems Messgaerate GmbH
- LSI
- Microsoft
- Mindtree
- NEC
- NetApp
- Olocity
- Oracle
- Patni
- PMC-Sierra
- Qlogic
- Quantum
- Royer Systems
- UCSD
- UCSC
- Seagate
- Sun
- Symantec
- Tata Consultancy
- VMWare
- Wipro
- Xyratex

*** Based on TWG Roster**

Some Customer Scenarios

- Escalating Amount of Data, becoming more expensive to manage
  - Is that old data valuable still?
- Meet governance needs
  - Who is in charge of deleting old data, finding it again?
- New projects with a desire not to have isolated local storage
  - Do they already go around existing storage practices?
- Storage of important data
  - How do you make sure it's requirements are being met?
- Storage for new Cloud Computing projects
  - Where does the data live long term?

Overview

- When discussing cloud storage and standards, it is important to distinguish the various ***resources*** that are being offered as services.
- These resources are exposed to clients as Functional interfaces (Data Path) and are managed by Management interfaces (Control Path).
- We explore the various types of interfaces that are part of offerings today and show how they are related.
- We propose a model for the interfaces that can be mapped to the various offerings as well as form the basis for rich cloud storage interfaces into the future.

Introduction

- Another important concept explored in this paper is that of *MetaData*.
  - When managing large amounts of data with differing requirements. Metadata is a convenient mechanism to express those requirements in such a way that underlying data services can differentiate their treatment of the data to meet those requirements.
- The appeal of cloud storage is due to some of the same attributes that define other cloud services: pay as you go, the illusion of infinite capacity (elasticity), and the simplicity of use/management.
  - It is therefore important that any interface for cloud storage support these attributes, while allowing for a multitude of business cases and offerings, long into the future.

What is Cloud Storage?

- The use of the term ***cloud*** in describing these new models arose from architecture drawings that typically used a cloud as the dominant networking icon.
- The cloud conceptually represented any to any connectivity in a network, but also an abstraction of concerns such the actual connectivity and the services running in the network that accomplish that connectivity with little manual intervention.

Cloud Storage Defined

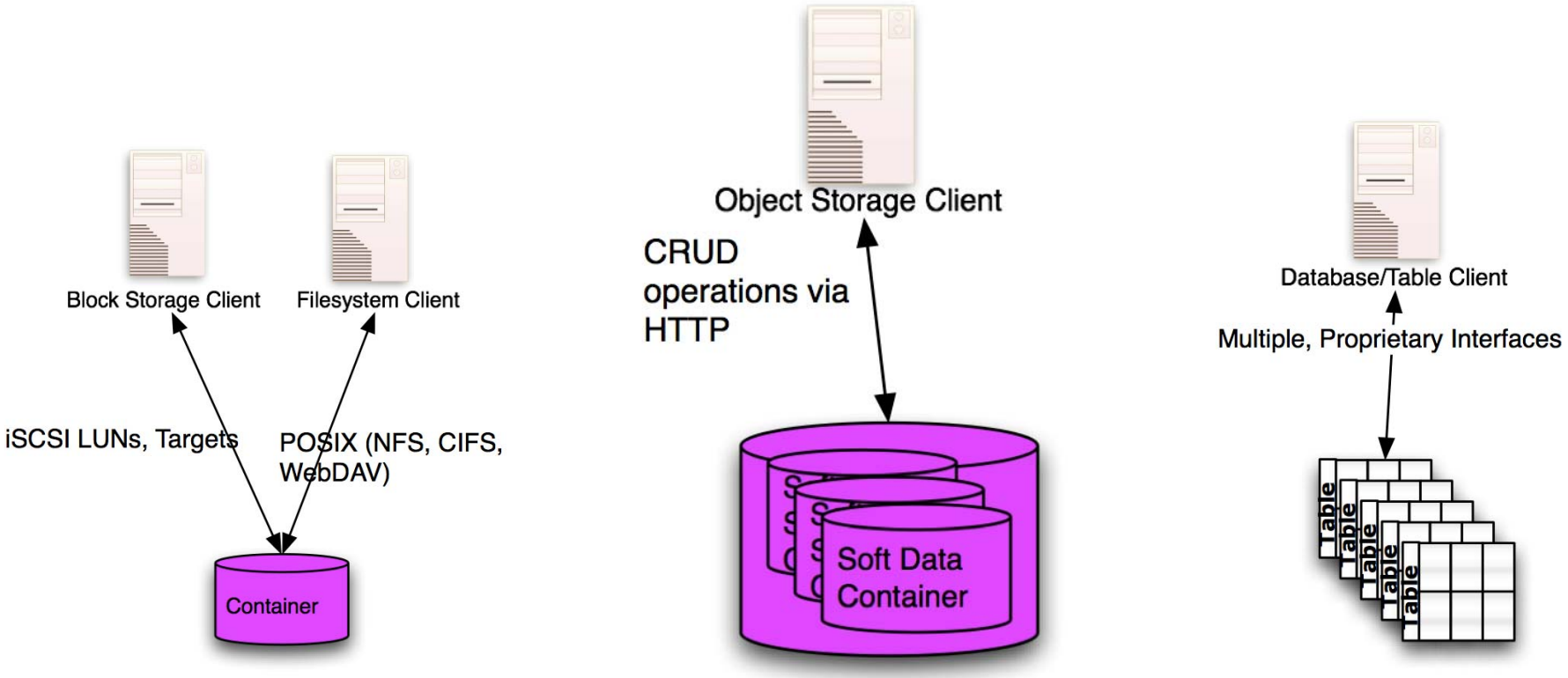
- This abstraction of complexity and promotion of simplicity is what primarily constitutes a cloud of resources, regardless of type.
  - An important part of the cloud model in general is the concept of a pool of resources that is drawn from upon demand in small increments (smaller than what you would typically purchase by buying equipment).
  - The recent innovation that has made this possible is virtualization.
- Thus cloud storage is simply the delivery of virtualized storage on demand. The formal term we proposed for this is *Data Storage as a Service* (**DaaS**).
- ***Data Storage as a Service***
  - *Delivery over a network of appropriately configured virtual storage and related data services, based on a request for a given service level. .*

Some use cases

- Offerings in the Data Storage as a Service space are increasing in capabilities
  - Additional service levels beyond "Best Effort" storage
  - Local appliances that "wrap" cloud storage for legacy applications
  - Cloud Storage offerings that layer onto best effort services
- These offerings go beyond merely storing the data, and start to account for data with differing requirements
  - There is a danger that the resulting complexity may cause the simplicity of cloud storage to be lost
  - There is an increasing "exit cost" to move from one vendor to another

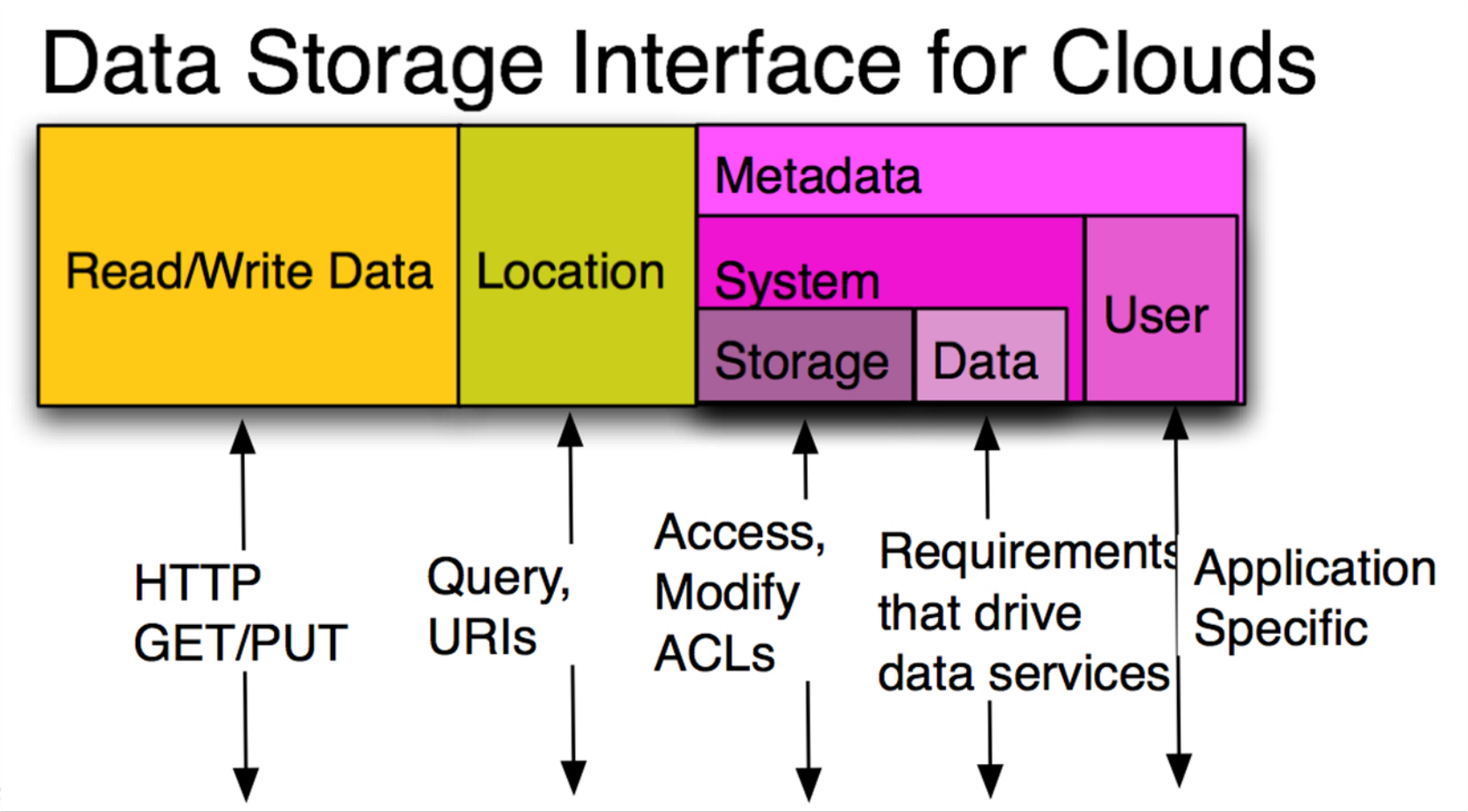
A look at some existing Cloud APIs

- What are some of the offerings and their Data Storage Interfaces?



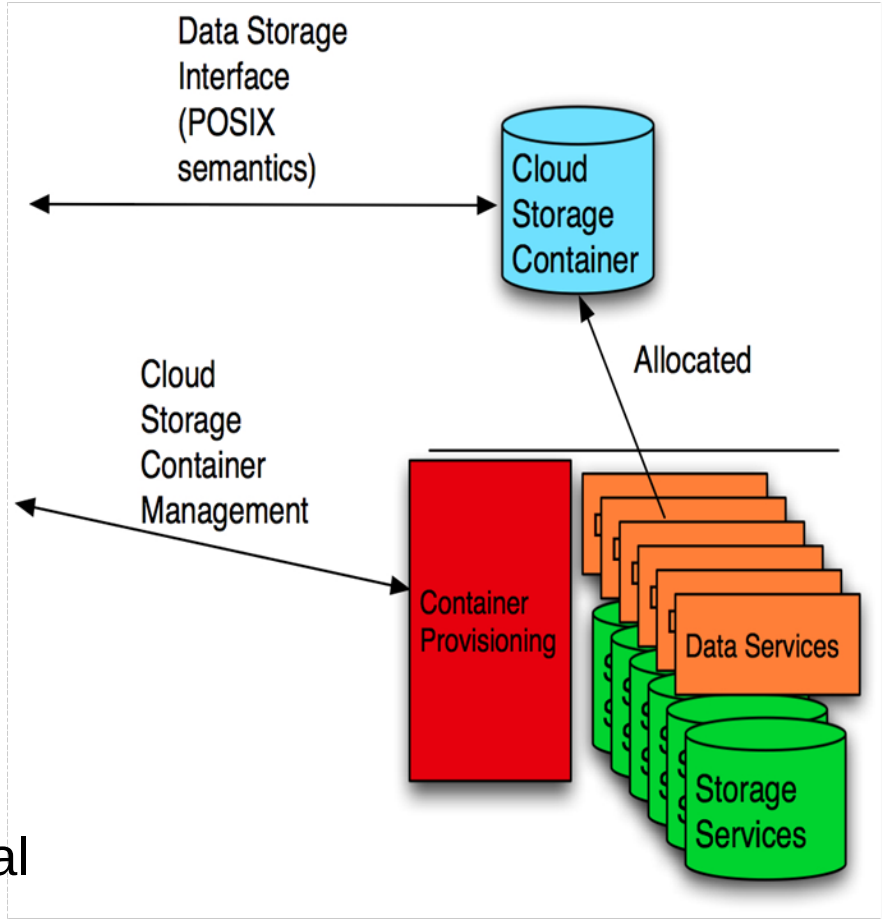
Leveraging the Storage Industry Resource Domain Model

All of these interfaces support some or all of this model. The key to retaining the simplicity of the cloud, however, is in the use of metadata to drive the underlying services so that users need not manage the services themselves.



Cloud Storage Container

- Cloud Storage is used as a volume/filesystem
- DSI Protocols include: WebDAV, NFS, CIFS, iSCSI, OSD
- Management interfaces: proprietary, Web UI, SMI-S
- Billing based on allocated space, Data Requirement (DR) parameters
- Resource guarantee (desired and required), consumption
- Configuration of DR is an object oriented hierarchy from containers on down to individual data elements



Container DSI

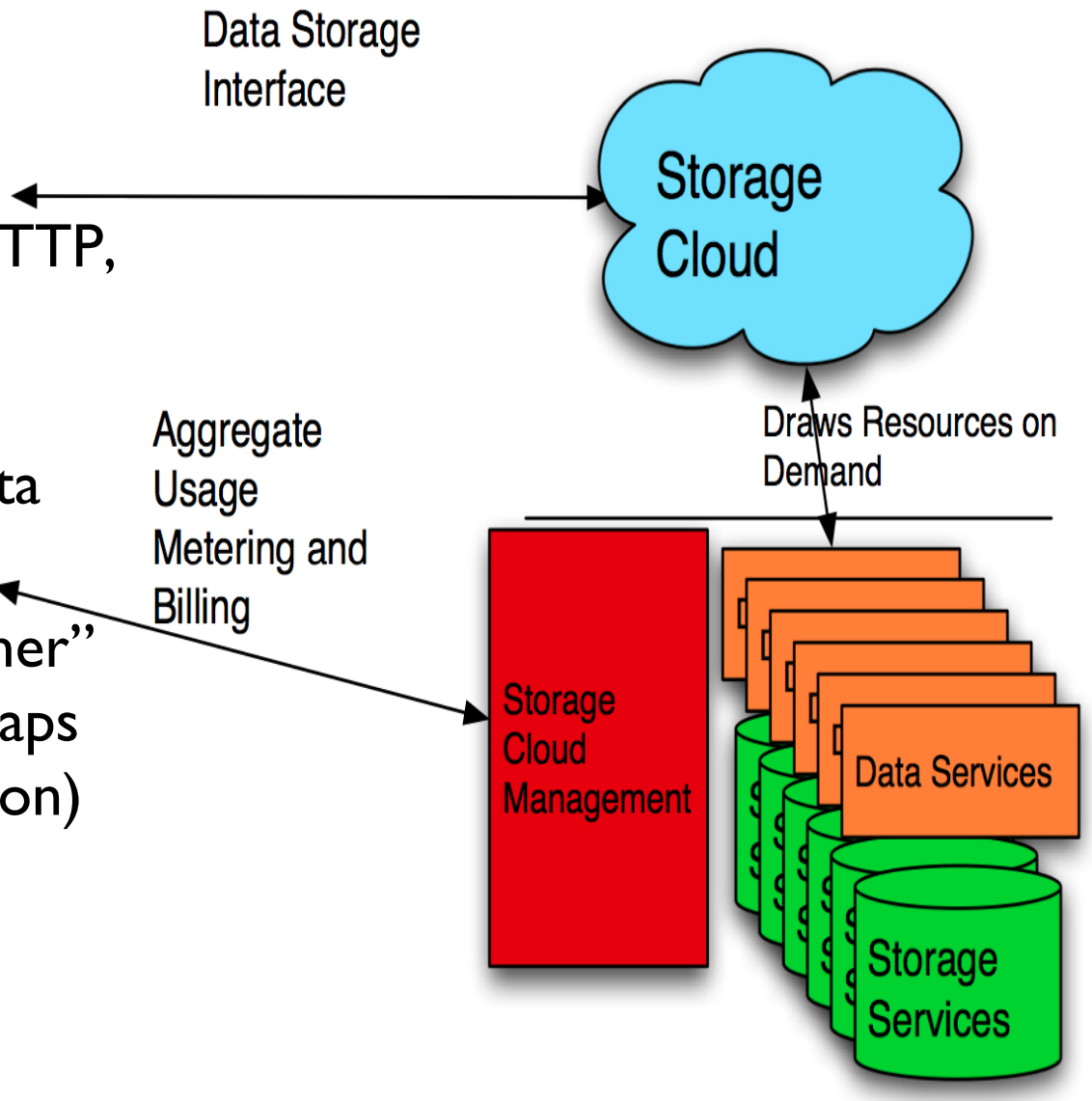
- Typically already standardized
- Needs support for metadata
- Need for standardization:
  - Standard Data System Metadata to be interpreted by the cloud as Data Requirements
- Broad categories of Data Requirements
  - Retention
  - Initial Security, Performance and Availability Requirements
    - Lifecycle – Defined Epochs with Requirements for each, Defined state transitions
    - Security Classification
    - Requirement may be expressed as minimum and desired
  - Location
    - Geography (political boundaries) – specific local requirements
    - Network Topology (bandwidth, latency to specific clients - Affinity to specific resources)
  - Budget
- History and Versioning metadata

Container Management

- Management Aspects
  - For iSCSI (for example) the "Container" could be equivalent to a Target, the LUNs would inherit (support a hierarchy of Containers) the Target's requirements, but could be overridden
- Allocation of Storage
  - Size (Initial. May be thin provisioned. May be grow-able)
  - Default Data Requirements (DR) expressed in the same schema as the file metadata
    - ACLs for the container as a whole as well
  - Requirements may be drawn from an existing set of templates
  - Price per GB (hint) of capacity allocated at this requirement
  - Price per GB (hint) of access (R vs. W) at this requirement
  - Timing aspects of the allocation – perhaps a "job" submittal
  - Characteristics and capabilities on each requirement (knob "stops")
    - Maximums (i.e. number of objects)
- Monitoring of Storage
  - Aggregate Consumed
  - Bandwidth Used – per Type
  - Reporting of "performance" against requirements
  - Based on connections to other elements
  - Container capability ranges for each requirement
  - Aggregate across containers

Storage Cloud

- Cloud storage is un-provisioned
- DSI interface: RESTful, HTTP, S3, XAM
- Management interfaces: proprietary, Web UI, Data System Metadata
- Soft definition of "container" and sub-containers (perhaps through a query expression)

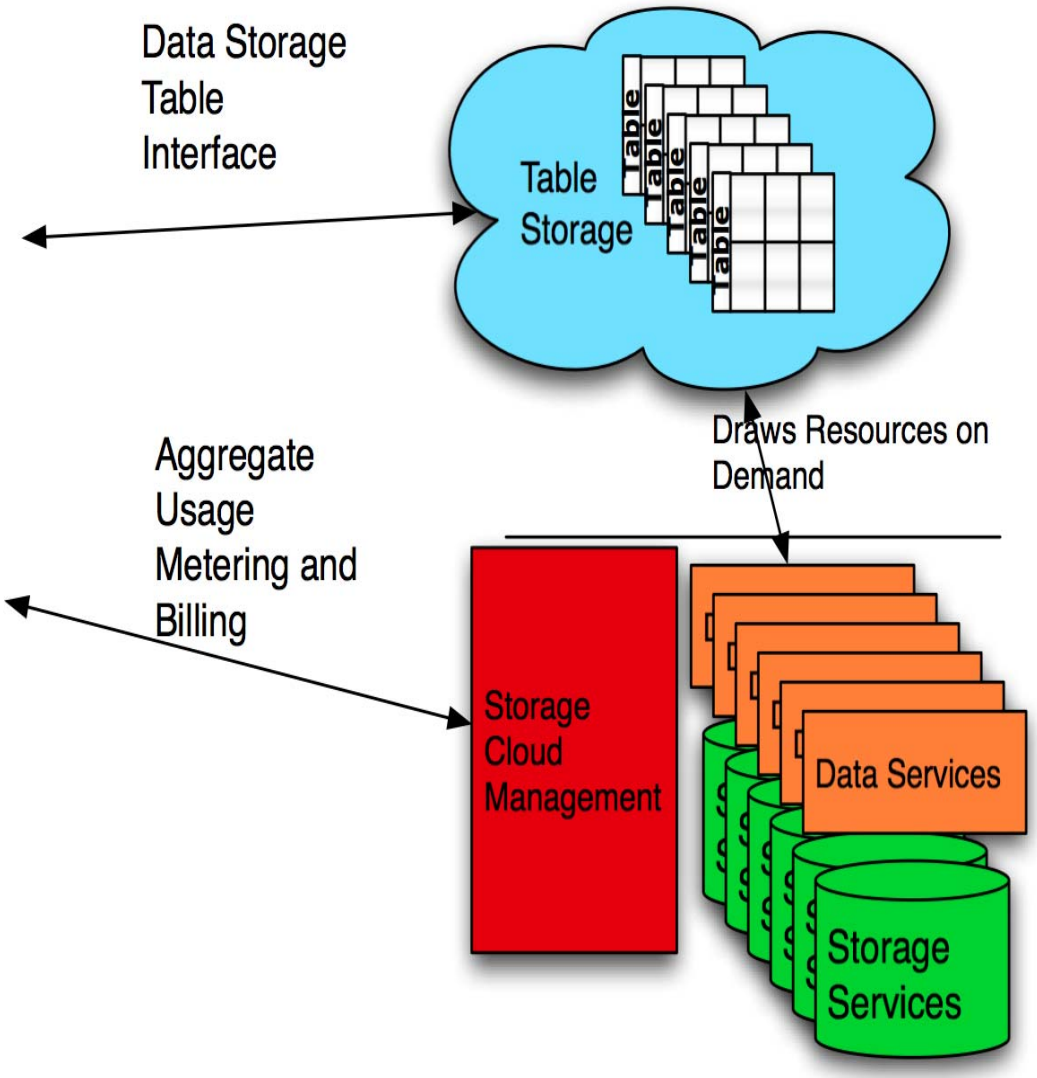


Cloud DSI

- Need to standardize existing proprietary interfaces
- Needs support for metadata
- Need for standardization:
  - Standard Data System Metadata to be interpreted by the cloud as Data Requirements

Cloud Table Storage

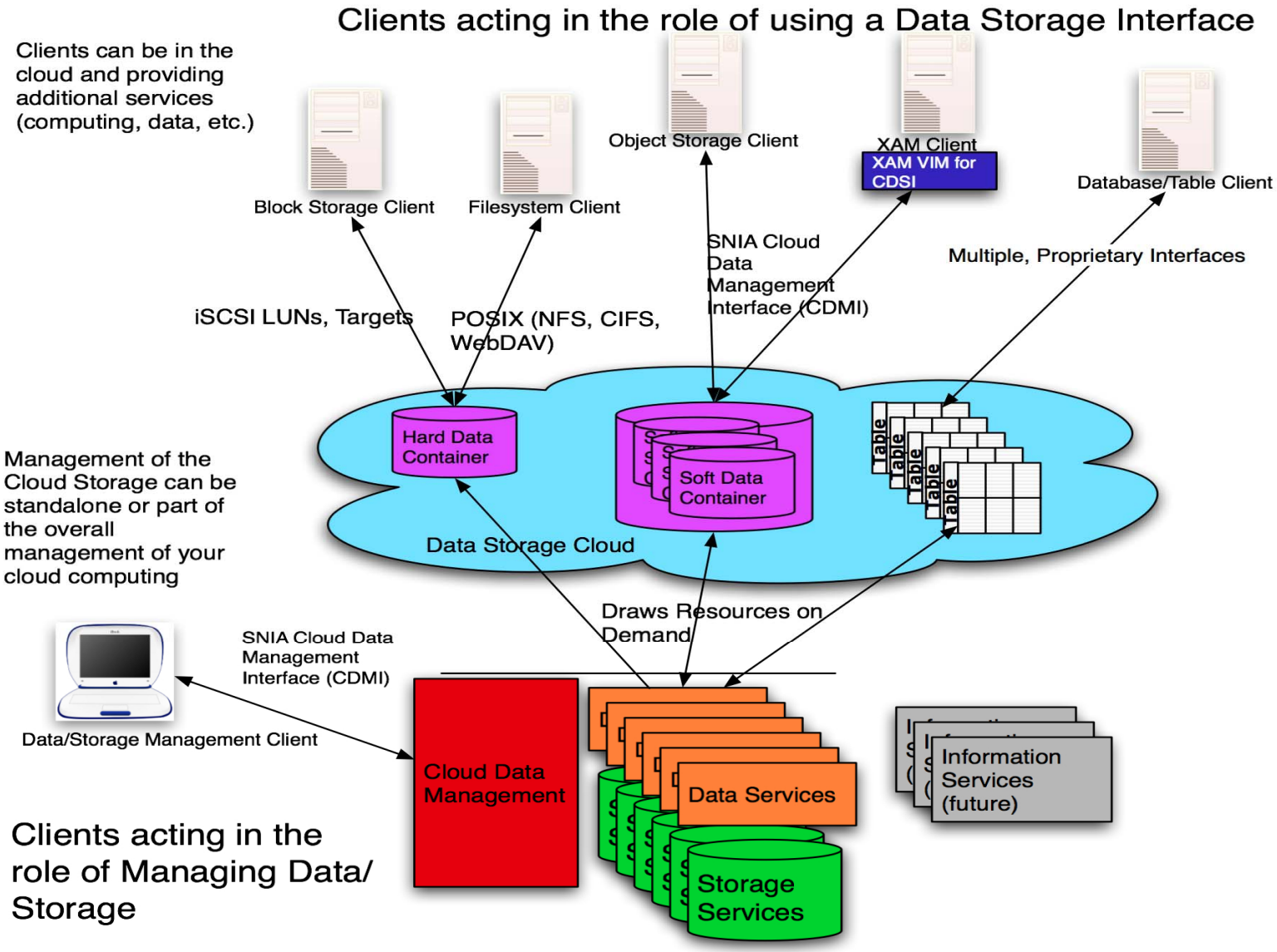
- Tables are created in the cloud
- Horizontally scalable "database"
- Query capability
- DSI protocols: proprietary
- Management protocols: Proprietary, Web UI
- Data Requirements may need to be expressed on a column or row basis



Cloud Table DSI

- Need to standardize existing proprietary interfaces?
  - This interface is rapidly evolving, so the actual standard I/F is low priority, but encouraging support for metadata can be done separately
- Needs support for metadata
  - Container, Table, Row/Column
- Need for standardization:
  - Standard Data System Metadata to be interpreted by the cloud as Data Requirements

The Complete Picture

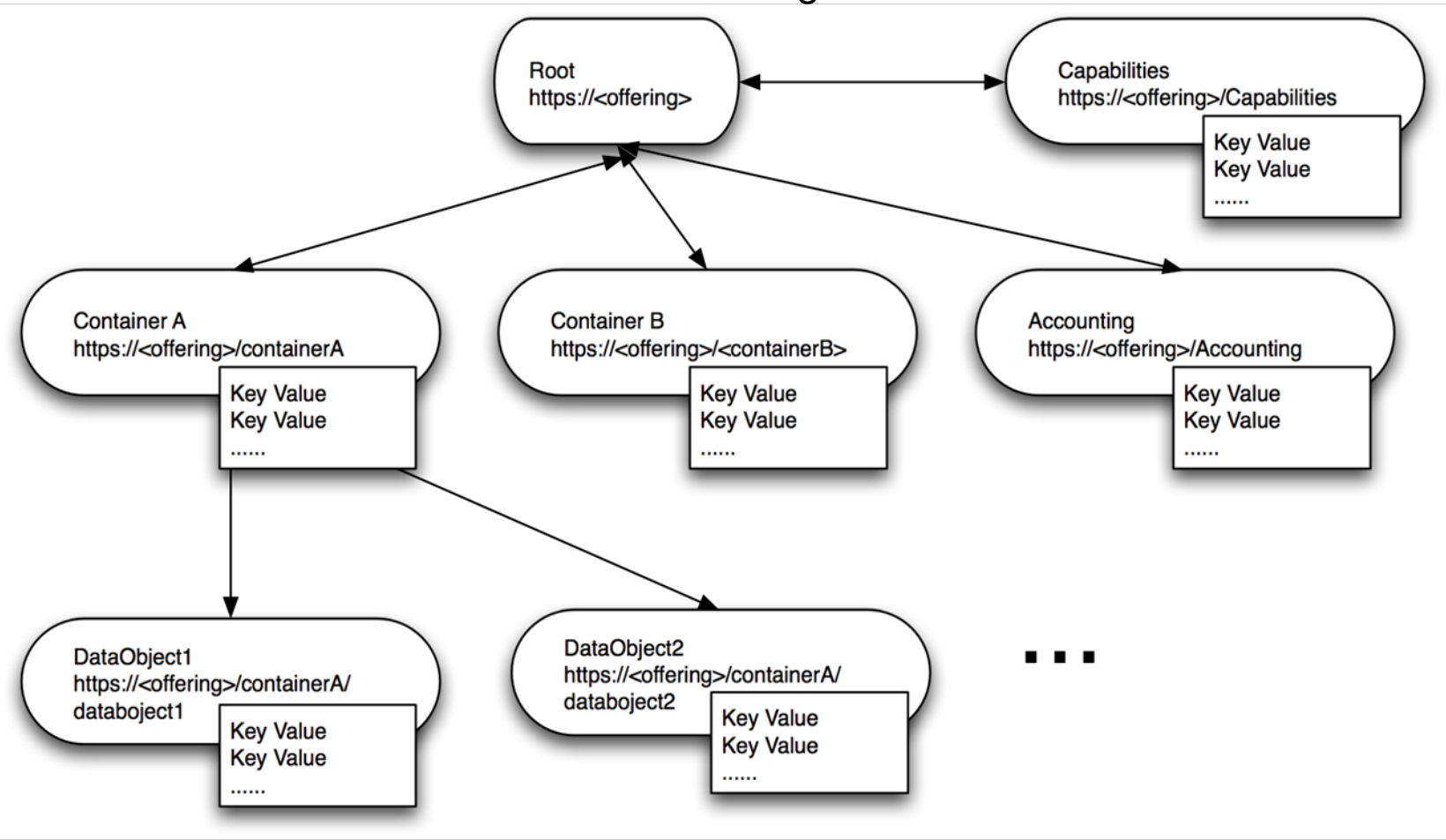


Cloud Data Management Interface

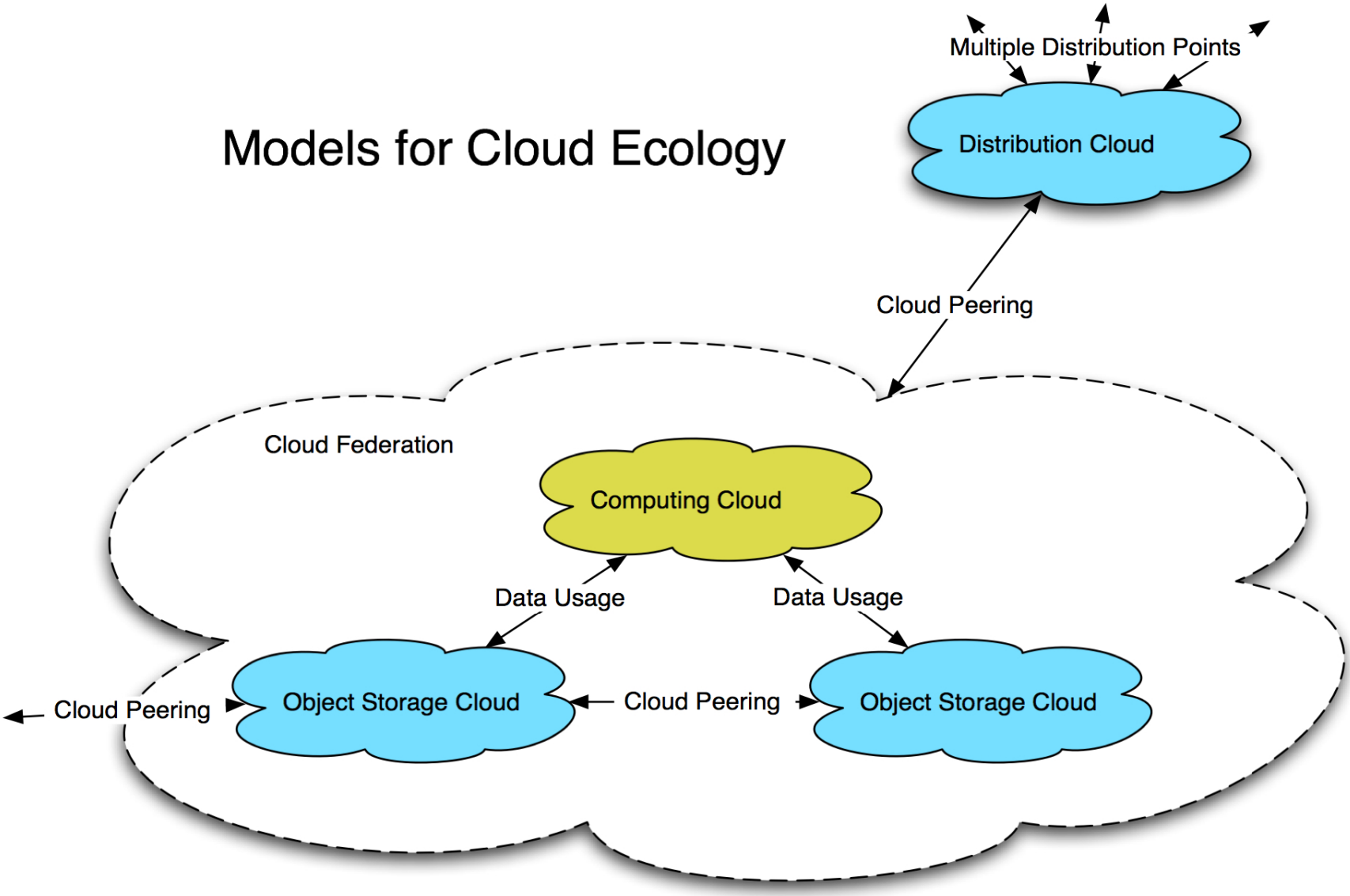
- Apply these to the "management interface" as well as the Data Interface"
- Semantics
  - Look to XAM, SMI-S for semantics – subset, extend where needed
  - Map-able where possible to other interfaces such as XAM, SMI-S, POSIX
- Protocols
  - RESTful HTTP as "core" interface style (violate where needed)
  - JSON vs. XML – format of the representations need to be extensible
  - We will define these representations for interoperability
- Language Bindings, Client Libraries
  - Standardize protocol, provide a few libraries, then only standardize the APIs and language bindings if needed
- Reference Implementation, Example Clients

Model for the Interface

The resources which are accessed through the RESTful interface



The Big Picture



CDMI Data Objects

- CDMI provides a RESTful HTTP API to allow management access for a wide range of stored data in cloud environments, including:
  - Files
    - http://cloud.example.com/files/notes.txt
  - Block Devices
    - http://cloud.example.com/luns/iscsi/bootimage
  - Object Stores
    - http://cloud.example.com/0x239F0930D3294FA8
  - Database Tables
    - http://cloud.example.com/database/table

CDMI Data Objects

- Stored data can be accessed using native protocols:
  - HTTP, CIFS, NFS, iSCSI, SQL, etc.
- Stored data can also be accessed using CDMI in a standardized manner. This facilitates:
  - Cloud-to-cloud migration
  - Cloud federation
  - Cloud backup
  - Cloud virus scanning
  - Cloud search
  - And more.
- Desired cloud storage characteristics can be associated with stored data:
  - Replication, Compression, Placement, Retention, QoS, etc.

CDMI Data Objects

- Get the contents of a stored file via normal HTTP:

```
HTTP connection established to http://cloud.example.com/ port 80

>> GET /files/notes.txt HTTP/1.1<CR><LF>
>> host: cloud.example.com<CR><LF>
>> <CR><LF>
<< HTTP/1.1 200 OK<CF><LF>
<< Date: Mon, 20 Jul 2009 05:48:19 GMT<CF><LF>
<< Content-Type: text/plain
<< Content-Length: 55
<< I am stored in a CDMI compatible cloud storage system.

HTTP Connection closed
```

CDMI Data Objects

- Get the information about a stored file via CDMI:

```
HTTP connection established to http://cloud.example.com/ port 80

>> GET /files/notes.txt HTTP/1.1<CR><LF>
>> host: cloud.example.com<CR><LF>
>> Accept: application/vnd.org.snia.cdmi.dataobject+json
>> Content-Type: application/vnd.org.snia.cdmi.dataobject+json
>> X-CDMI-Specification-Version: 1.0
>> <CR><LF>
<< HTTP/1.1 200 OK<CF><LF>
<< Date: Mon, 20 Jul 2009 05:48:19 GMT<CF><LF>
<< Content-Type: application/vnd.org.snia.cdmi.dataobject+json
<< Content-Size: 242
<< X-CDMI-Specification-Version: 1.0
<< {
<<    “name” : “notes.txt”,
<<    “URI” : “/files/notes.txt”,
<<    “MimeType” : “text/plain”,
<<    “metadata” : [ “max_latency” : “12 hours”,
<<                   “min_replicas” : “3”, ],
<<    “cdmi_objectid” = “2fbf3630-7f9f-4551-8dd6-ae8de7aa30d3”,
<<    “capabilitiesURI” : “/capabilities/dataobject”,
<<    “value” : “I am stored in a CDMI compatible cloud storage system.”,
<< }

HTTP Connection closed
```

CDMI Data Objects

- Content-Type indicates management access:

```
HTTP connection established to http://cloud.example.com/ port 80

>> GET /files/notes.txt HTTP/1.1<CR><LF>
>> host: cloud.example.com<CR><LF>
>> Accept: application/vnd.org.snia.cdmi.dataobject+json
>> Content-Type: application/vnd.org.snia.cdmi.dataobject+json
>> X-CDMI-Specification-Version: 1.0
>> <CR><LF>
<< HTTP/1.1 200 OK<CF><LF>
<< Date: Mon, 20 Jul 2009 05:48:19 GMT<CF><LF>
<< Content-Type: application/vnd.org.snia.cdmi.dataobject+json
<< Content-Size: 242
<< X-CDMI-Specification-Version: 1.0
<< {
<<    "name" : "notes.txt",
<<    "URI" : "/files/notes.txt",
<<    "MimeType" : "text/plain",
<<    "metadata" : [ "max_latency" : "12 hours",
<<                   "min_replicas" : "3", ],
<<    "cdmi_objectid" = "2fbf3630-7f9f-4551-8dd6-ae8de7aa30d3",
<<    "capabilitiesURI" : "/capabilities/dataobject",
<<    "value" : "I am stored in a CDMI compatible cloud storage system.",
<< }

HTTP Connection closed
```

CDMI Data Objects

- CDMI Data Object information:

```
HTTP connection established to http://cloud.example.com/ port 80

>> GET /files/notes.txt HTTP/1.1<CR><LF>
>> host: cloud.example.com<CR><LF>
>> Accept: application/vnd.org.snia.cdmi.dataobject+json
>> Content-Type: application/vnd.org.snia.cdmi.dataobject+json
>> X-CDMI-Specification-Version: 1.0
>> <CR><LF>
<< HTTP/1.1 200 OK<CF><LF>
<< Date: Mon, 20 Jul 2009 05:48:19 GMT<CF><LF>
<< Content-Type: application/vnd.org.snia.cdmi.dataobject+json
<< Content-Size: 242
<< X-CDMI-Specification-Version: 1.0
<< {
<<     “name” : “notes.txt”,
<<     “URI” : “/files/notes.txt”,
<<     “MimeType” : “text/plain”,
<<     “metadata” : [ “max_latency” : “12 hours”,
<<                    “min_replicas” : “3”, ],
<<     “cdmi_objectid” = “2fbf3630-7f9f-4551-8dd6-ae8de7aa30d3”,
<<     “capabilitiesURI” : “/capabilities/dataobject”,
<<     “value” : “I am stored in a CDMI compatible cloud storage system.”,
<< }

HTTP Connection closed
```

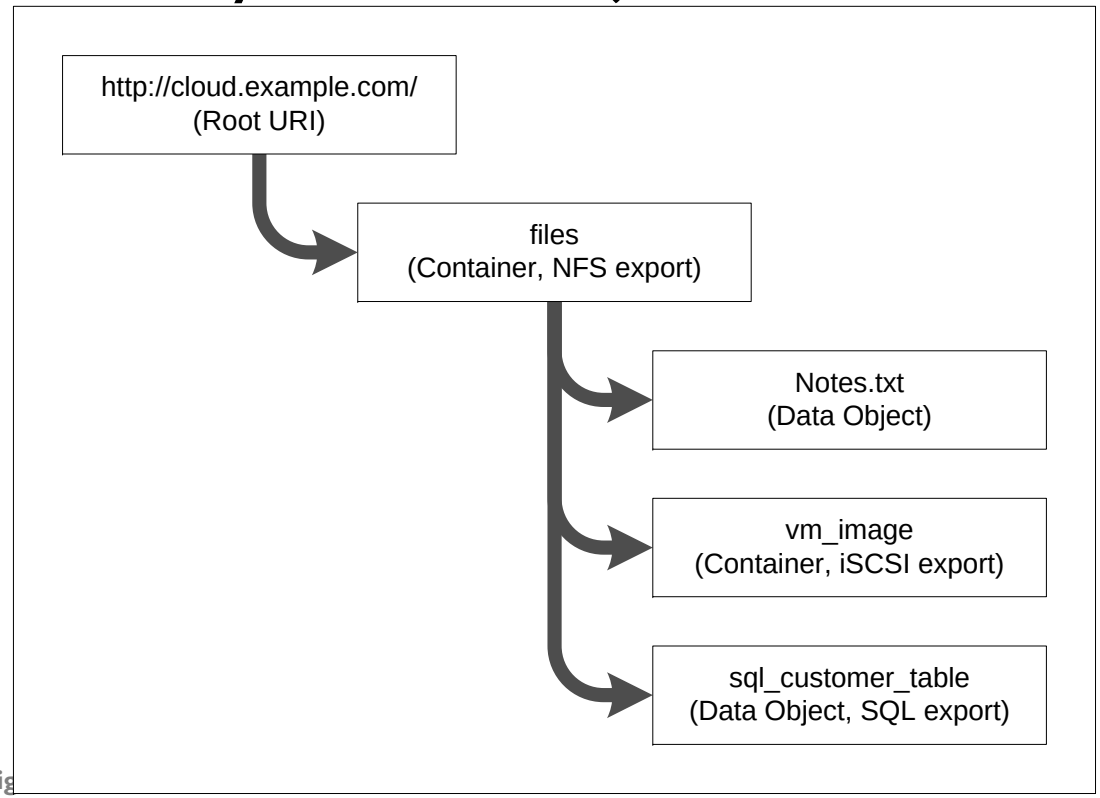
CDMI Data Objects

- Examples of Data System Metadata:

```
HTTP connection established to http://cloud.example.com/ port 80

>> GET /files/notes.txt HTTP/1.1<CR><LF>
>> host: cloud.example.com<CR><LF>
>> Accept: application/vnd.org.snia.cdmi.dataobject+json
>> Content-Type: application/vnd.org.snia.cdmi.dataobject+json
>> X-CDMI-Specification-Version: 1.0
>> <CR><LF>
<< HTTP/1.1 200 OK<CF><LF>
<< Date: Mon, 20 Jul 2009 05:48:19 GMT<CF><LF>
<< Content-Type: application/vnd.org.snia.cdmi.dataobject+json
<< Content-Size: 317
<< X-CDMI-Specification-Version: 1.0
<< {
<<    “name” : “notes.txt”,
<<    “URI” : “/files/notes.txt”,
<<    “MimeType” : “text/plain”
<<    “metadata” : [ “max_latency” : “12 hours”,
<<                   “min_replicas” : “3”, ],
<<    “cdmi_objectid” = “2fbf3630-7f0f-455f-8dd6-ee8de7ee30d2”
<<    “capabilitiesURI” : “/capabilities/dataobject”,
<<    “value” : “I am stored in a CDMI compatible cloud storage system.”,
<< }

HTTP Connection closed
```

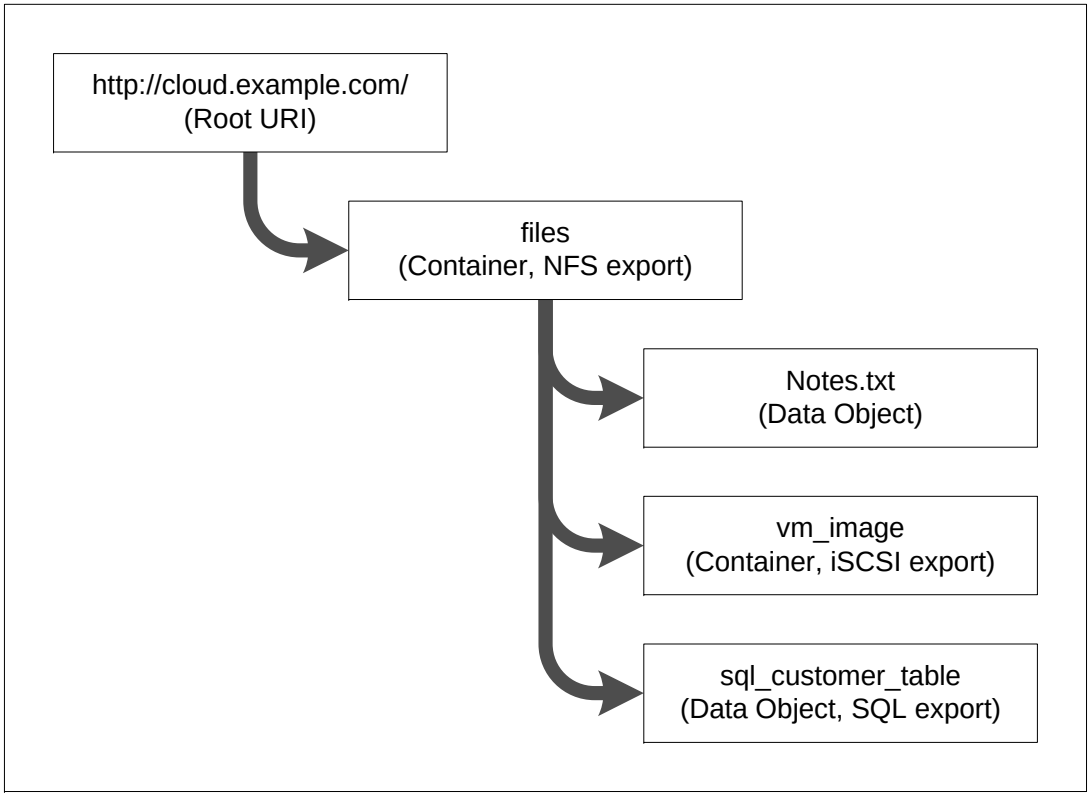
CDMI Containers

- CDMI uses hierarchies to enable simplified management
  - Data System Metadata can be specified on a container, and inherited by all child objects.



CDMI Containers

- Objects and containers can be exported using other native protocols.



CDMI Containers

- Get the metadata and children of a CDMI container:

```
HTTP connection established to http://cloud.example.com/ port 80

>> GET /files HTTP/1.1<CR><LF>
>> host: cloud.example.com<CR><LF>
>> Accept: application/vnd.org.snia.cdmi.container+json
>> Content-Type: application/vnd.org.snia.cdmi.container+json
>> X-CDMI-Specification-Version: 1.0
>> <CR><LF>
<< HTTP/1.1 200 OK<CF><LF>
<< Date: Mon, 20 Jul 2009 05:48:19 GMT<CF><LF>
<< Content-Type: application/vnd.org.snia.cdmi.container+json
<< Content-Size: 215
<< X-CDMI-Specification-Version: 1.0
<< {
<<    “name” : “files”,
<<    “URI” : “/files”,
<<    “metadata” : [ “min_replicas” : “2”,
<<                   “compression” : “true”, ],
<<    “cdmi_objectid” = “2fbf3630-7f9f-4551-8dd6-ae8de7aa30d3”,
<<    “capabilitiesURI” : “/capabilities/container”,
<<    “children” : [ “notes.txt”, “vm_image”, “sql_customer_table”, ],
<< }

HTTP Connection closed
```

CDMI Containers

- Data system metadata in CDMI is inherited from parent containers to child containers and data objects.

- In the previous example, the minimum number of replicas was set to "2" for the container. This sets the default value for children of this container.

- The child "notes.txt" has data system metadata that defines the minimum number of replicas to "3". This overrides the default value inherited from the parent container.

CDMI Accounts

- Accounts provide a place to manage who can access cloud content, discover what operations have been performed on cloud content, and obtain billing and reporting information related to cloud content.

- Account Summaries provide summary information about the account
- Account Membership provides access to credential mapping and permissions
- Account Notifications provide details on events occurring in the account
- Account Query provides mechanisms for discovering content within an account

CDMI Accounts - Summary

- Get the daily account summary object:

```
HTTP connection established to http://cloud.example.com/ port 80

>> GET /myaccount/cdmi_account_summary/daily/2009-07-20 HTTP/1.1<CR><LF>
>> host: cloud.example.com<CR><LF>
>> <CR><LF>
<< HTTP/1.1 200 OK<CF><LF>
<< Date: Mon, 20 Jul 2009 05:58:29 GMT<CF><LF>
<< Content-Type: text/plain
<< <CR><LF>
<< {
<<    “cdmi_account_name” : “myaccount”,
<<    “cdmi_summary_start” : “2009-07-20T05:49:19”,
<<    “cdmi_summary_end” : “2009-07-20T05:58:29”,
<<    “cdmi_summary_objects” : “1”,
<<    “cdmi_summary_puts” : “1”,
<<    “cdmi_summary_gets” : “4”,
<<    “cdmi_summary_bytes” : “34”,
<<    “cdmi_summary_writes” : “34”,
<<    “cdmi_summary_reads” : “136”,
<<    “cdmi_summary_rawbytes” : “2422”
<<    “cdmi_summary_power” : “1”,
<<    “cdmi_summary_cost” : “0.05”,
<< }

HTTP Connection closed
```

CDMI Accounts - Membership

- Account Membership allows access to user/account/privilege information for the account
  - Add, disable and remove users (enrollment)
  - Specify user credentials
  - Allow users to change their credentials
  - Define delegation of credentials (e.g., external LDAP, AD, etc.)

- Allows users to define multi-party sharing relationships
  - Essential for peering/federation relationships
  - User can create an account for a third-party virus scanner, for example, and grant it only the privileges needed for this role

CDMI Accounts - Membership

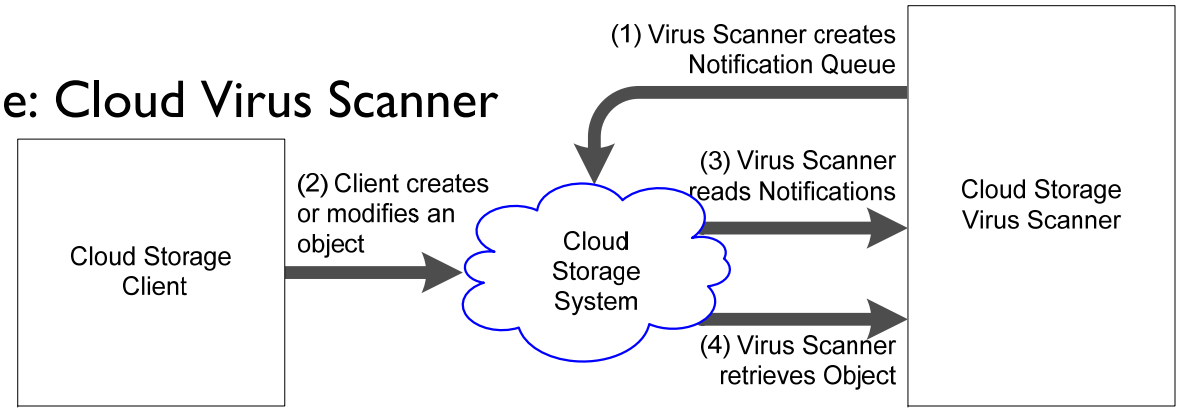
- Get an account member object:

```
HTTP connection established to http://cloud.example.com/ port 80

>> GET /myaccount/cdmi_account_members/jdoe HTTP/1.1<CR><LF>
>> host: cloud.example.com<CR><LF>
>> <CR><LF>
<< HTTP/1.1 200 OK<CF><LF>
<< Date: Mon, 20 Jul 2009 05:58:29 GMT<CF><LF>
<< Content-Type: text/plain
<< Content-Length: 522
<< <CR><LF>
<< {
<<    “cdmi_member_name” : “jdoe”,
<<    “cdmi_credentials_type” : “certificate”,
<<    “cdmi_credentials” : “-----BEGIN CERTIFICATE-----
          MIIC2DCCAkGgAwIBAgIDEL90MA0GCSqGSIb3DQEBBAUAMGwxCzAJBgNVBAYTAlVT ...
          -----END CERTIFICATE-----”,
<<    “cdmi_acl_name” : “jdoe”,
<<    “cdmi_groups” : [ “”, ],
<<    “cdmi_privilages” : [ “administrator”, ],
<<    “cdmi_quota” : “1000000000”,
<< }

HTTP Connection closed
```

CDMI Accounts - Notification

- Notification queues allow a cloud storage system to tell a client about changes of state
  - Object created, modified, deleted, etc…
  - Each change results in an event that is enqueued into a queue
  - Clients can create queues, and specify what events they are interested in, and for each event, what information should be included

- Example: Cloud Virus Scanner



CDMI Capabilities

- Capabilities define what storage operations a CDMI provider is capable of providing.
- Contrast with permissions, which define what storage operations a CDMI provider will permit a user to perform.
- Capabilities are static for a given cloud storage system, but different sets of capabilities may be present for different URIs.

CDMI Capabilities

- CDMI Clients typically use capabilities for the following purposes:
  - Initial introspection of the capabilities of a cloud storage provider
  - Introspection of the capabilities of a CDMI object at a given URI
- In order to facilitate the first use case, every CDMI system shall have a tree structure of capabilities objects that is referenced from the root.
- In order to facilitate the second use case, every CDMI object will have a capabilitiesURI field that points to the capabilities object corresponding to the capabilities of the specific CDMI object.

CDMI Capabilities

- List the contents of the root URI:

```
HTTP connection established to http://cloud.example.com/ port 80

>> GET / HTTP/1.1<CR><LF>
>> host: cloud.example.com<CR><LF>
>> Accept: application/vnd.org.snia.cdmi.container+json<CR><LF>
>> Content-Type: application/vnd.org.snia.cdmi.container+json<CR><LF>
>> X-CDMI-Specification-Version: 1.0<CR><LF>
>> <CR><LF>
<< HTTP/1.1 200 OK<CF><LF>
<< Date: Mon, 20 Jul 2009 05:48:19 GMT<CF><LF>
<< Content-Type: application/vnd.org.snia.cdmi.container+json<CR><LF>
<< X-CDMI-Specification-Version: 1.0<CR><LF>
<< Content-Length: 123<CR><LF>
<< {
<<    “name” : “”,
<<    “URI” : “/”,
<<    “objectID” : “54619e21-baa2-45cd-9f53-9a8a27750bc8”,
<<    “capabilitiesURI” : “/capabilities/root”,
<<    “metadata” : [ , ],
<<    “children” : [ “files”, ],
<< }

HTTP Connection closed
```

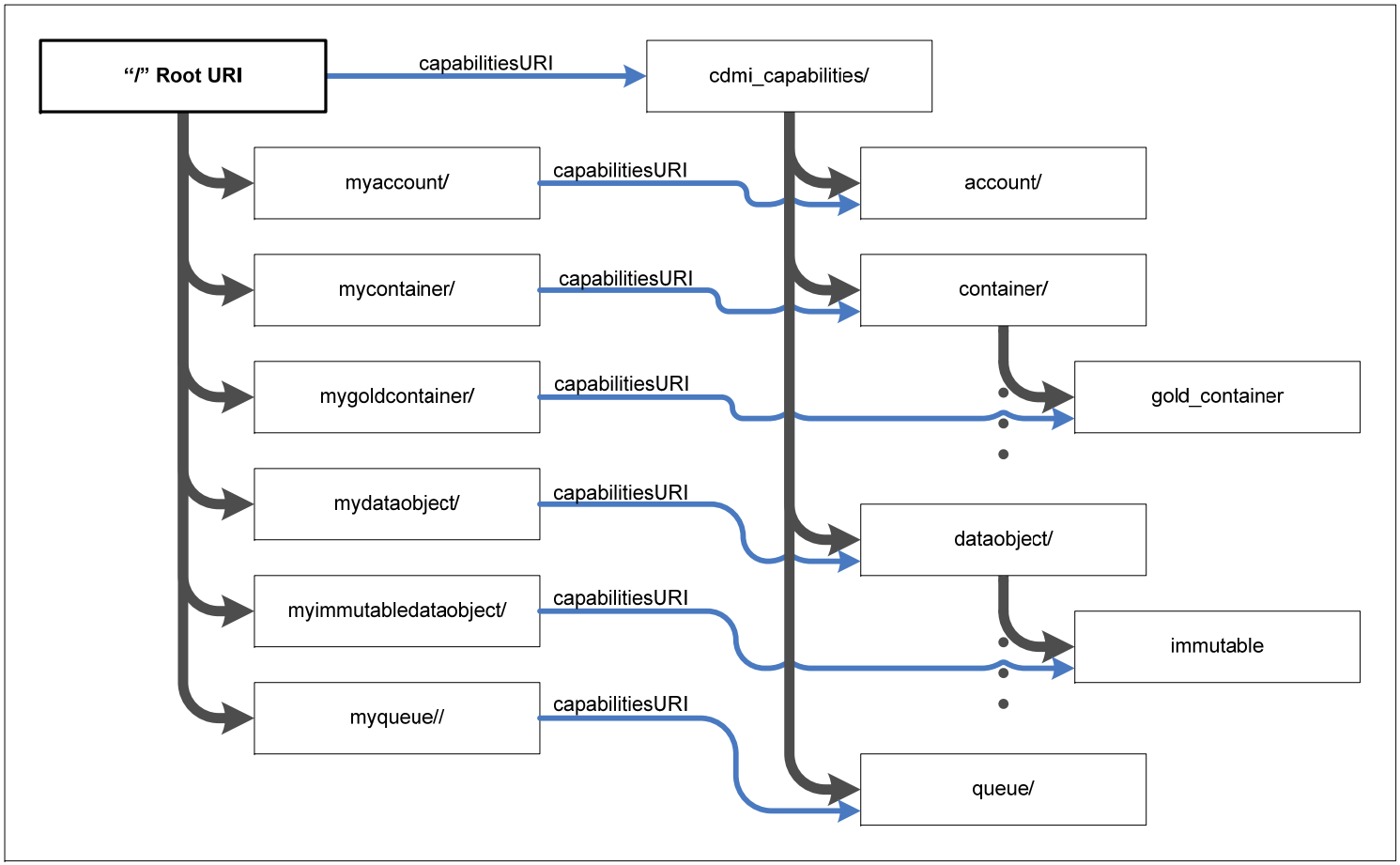
CDMI Capabilities

- Get the capabilities object:

```
HTTP connection established to http://cloud.example.com/ port 80

>> GET /capabilities/root HTTP/1.1<CR><LF>
>> host: cloud.example.com<CR><LF>
>> Accept: application/vnd.org.snia.cdmi.capabilities+json<CR><LF>
>> Content-Type: application/vnd.org.snia.cdmi.capabilities+json<CR><LF>
>> X-CDMI-Specification-Version: 1.0<CR><LF>
>> <CR><LF>
<< HTTP/1.1 200 OK<CF><LF>
<< Date: Mon, 20 Jul 2009 05:48:19 GMT<CF><LF>
<< Content-Type: application/vnd.org.snia.cdmi.capabilities+json<CR><LF>
<< X-CDMI-Specification-Version: 1.0<CR><LF>
<< Content-Length: 136<CR><LF>
<< {
<<    “name” : “root”,
<<    “URI” : “/capabilities/root”,
<<    “objectID” : “e949c80d-aba5-42f0-921a-903ddc95f359”,
<<    “capabilities” : [ “cdmi_list”, “cdmi_serialize”, “cdmi_create_object”,
<<                       “cdmi_create_container”, ],
<<    “children” : [ “account”, “container”, “dataobject”, “queue”, ],
<< }

HTTP Connection closed
```

CDMI Capabilities

- The below diagram illustrates how CDMI objects relate to capabilities:



CDMI Capabilities

- The following aspects of capabilities are up to the cloud storage implementer:
  - The layout of the capabilities tree
  - The capabilities in each capabilities object
  - The mapping between created CDMI objects and capabilities objects
- The following aspects of capabilities are mandatory:
  - A root capabilities object that describes system-wide capabilities
  - A capabilities URI link from every object to a corresponding capabilities object
  - A capabilities URI link from the root URI to the root capabilities object

High Level Cloud TWG Roadmap

- Taxonomy – submit terms for SNIA Dictionary (done)
- Use cases – first draft June 2009
- Reference Model (RM) – first draft June 2009
- SNIA Cloud Data Management Interface (CDMI) – first draft SDC 09
- Reference Implementation (RI) – first work in progress release December '09
- Q1/2010
  - Update on CDMI, RI, Use Cases and RM

Questions

- Thank you!