its@PennState
information technology services

We save everything!

This model can't be sustained !

Repository (BAD Word)

- **Ways to collect data or objects**
  - Fedora, DSpace, Wiki's, Blogs (Software)
  - Databases
  - Directory on a disk
  - Cloud
  - Etc….

Silos of information continue to grow

its@PennState
information technology services

Silo Examples

its@PennState
information technology services

Oh…. And did I mention

- **Petabytes**
  - or more
- **Compliance issues**
  - we need to keep some data 72 years
- **Hardware & software life cycle issues**
  - life cycling around the data
- **Data migration issues**
  - how do we know it got moved

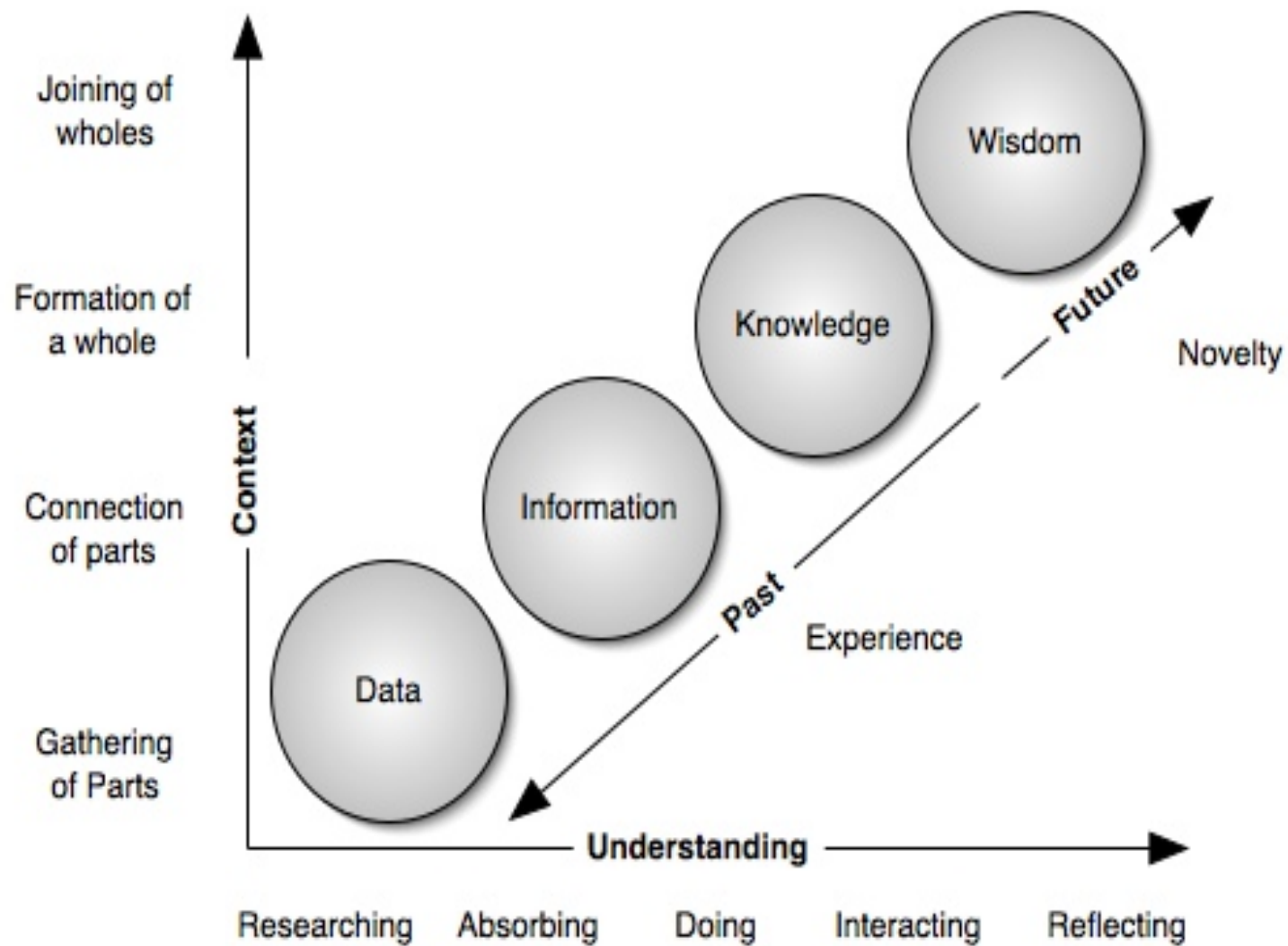# Data Value Proposition

(Can we afford it)

- **Tiered Storage Models**
  - Customer chooses value (with realistic expectations)
    - Expensive redundant disks all the way to tape
  - **XAM e<u>X</u>tensible <u>A</u>ccess <u>M</u>ethod**
    - PSU driving industry to adopt
    - It's all about the metadata
    - e-discovery, provenance, searchability
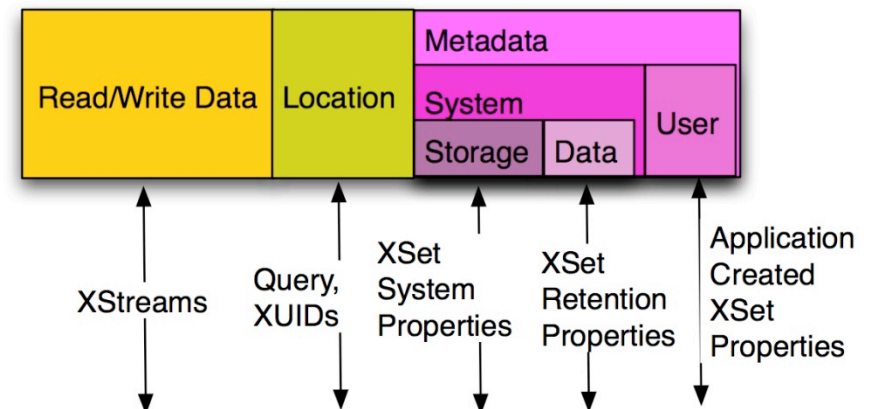
Some basic cloud storage attributes

- Pay as you go
- Self service provisioning
- Rich application interfaces
- No need for consumers to manage their own storage

How does an interface like XAM address these needs?

XAM API: an example Data Storage Interface

- XAM is the first interface to standardize system metadata for retention of data

- ❖ XAM implements the basic capability to Read and Write Data (through Xstreams)
- ❖ XAM has the ability to locate any XSet with a query or by supplying the XUID
- ❖ XAM allows Metadata to be added to the data and keeps both in an XSet object
- ❖ XAM uses and produces system metadata for each XSet
- ❖ For example Access and Commit times (Storage System Metadata)

- ❖ But it also uniquely specifies Data System Metadata for Retention Data Services

- ❖ XAM User metadata is un-interpretable by the system, but stored with the other data and is available for use in queries

- ❖ Given this we can see that XAM is a data storage interface that is used by both Storage and Data Services (functions)

XSet Interface for XAM

Read/Write Data	Location	Metadata			
		System			User
		Storage	Data		

XStreams — Query, XUIDs — XSet System Properties — XSet Retention Properties — Application Created XSet Properties

Metadata for Data Services

•Metadata available through the data storage interface may also be managed by data services

•This data service metadata can be used by data services to provide differentiated value to individual data elements

•The model or schema for data service metadata may be defined by each data service and may be standardized
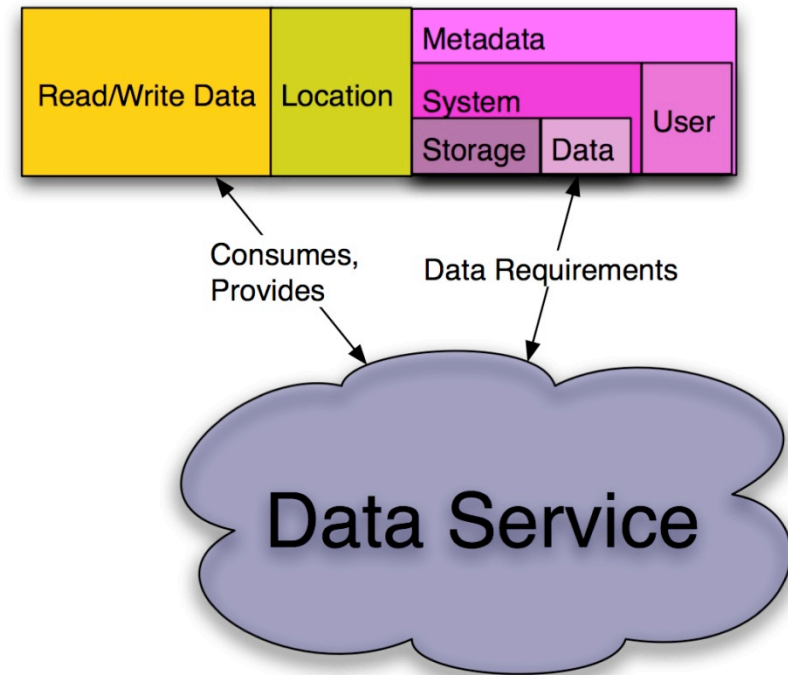
Metadata and Information Services

➢The role of metadata in information services is as a communication mechanism with the underlying storage services and data services.

➢Information services are primarily concerned with the data service system metadata as a means to convey the data's requirements to the underlying data services.

➢An information service may also interpret user metadata for purposes of data classification.

➢An information service can create its own user metadata that is un-interpreted by the underlying services for its own use.
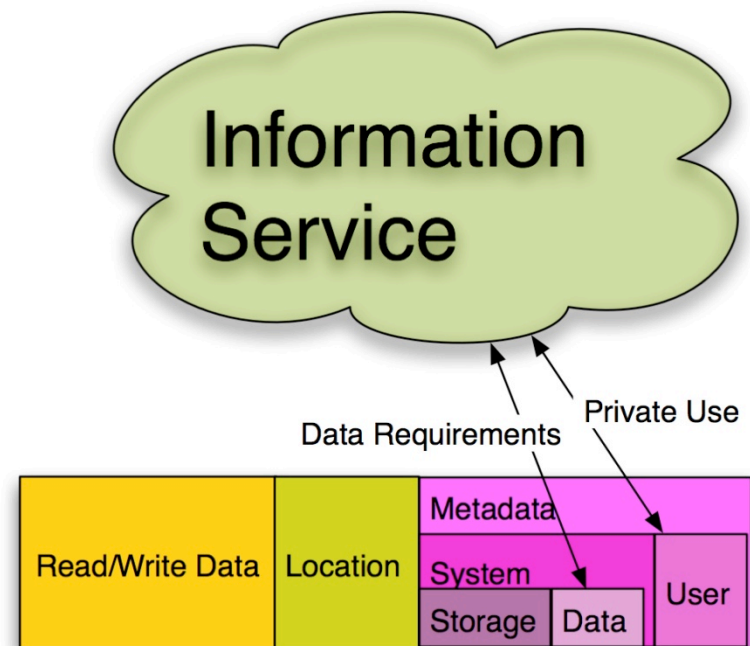
Lining it up

- Pay as you go
  - Billing would be orthogonal to XAM API, but as XSets are created and destroyed, you would be billed accordingly

- Self service provisioning
  - Just create and destroy XSets

- Rich application interfaces
  - Metadata, query, import/export, etc.

- No need for consumers to manage their own storage
  - Focus instead on managing the data (setting its requirements)

Cloud API to the Resource Domain Model

Nothing further is needed to be standardized in order to use XAM for a Cloud Storage API today.

Applications are written to a local (Java, C, etc.) API and the Cloud vendor's VIM handles the rest.

User

Metadata | **System** | **Data**

Storage

Location

Read/Write Data

Business Context Determines Requirements

Interprets System Metadata

Consumes to add value to the data

Provides storage and retrieval for data

Information Resource Domain
Services understand the semantics of the content in context

Data Requirements, Classification

Data Resource Domain
The content is opaque to the Services and without context

Added Value Behind Virtualized Interface

Storage Resource Domain
The bits are contained by these Services

Information Policies
Data classified according to importance to organization

Data Policies
Data treated according to requirements, lifecycle

Storage Policies
Ensure correct and reliable operation