Towards Transparent Integration of Heterogeneous Cloud Storage Platforms

Ilja Livenson*, Erwin Laure

KTH PDC

livenson@kth.se

* Presenter

Outline

- Motivation and problem
- Our approach
  - CDMI-Proxy
- Status and roadmap

Background

- Work done within EU FP7 VENUS-C Project
  - creating a platform that enables user applications to leverage on cloud computing principles;
  - creating a sustainable infrastructure with a valid business model.
- Resource providers are MS Azure, Engineering, BSC and KTH
- User scenarios from biomedicine, civil engineering, civil protection and emergencies, marine biodiversity and more.

Problem

- Lacking component – common storage access mechanism
- Clouds typically expose RESTful interfaces for file access
  - AWS S3 or MS Azure Blob
- DCI and local infrastructures (including laptops) tend to provide POSIX interface
  - FS or shared FS
- Need to offer a compatibility layer

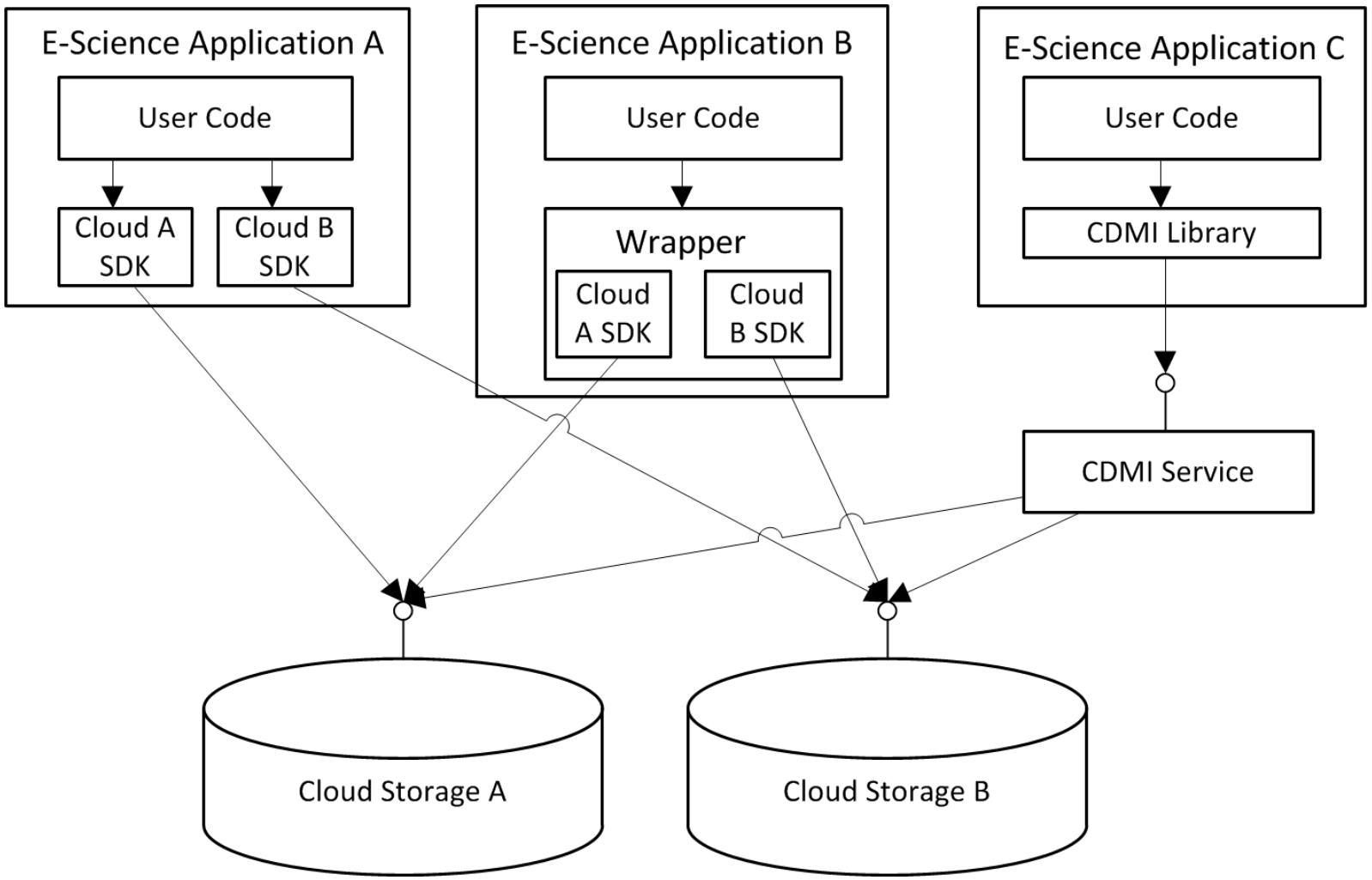
Storage Objects

- There are three objects with generally close semantics
  - Container
  - Blob
  - Message Queue
- Each resource provider offers its own flavour of APIs
  - AWS S3 vs MS Azure Blob vs POSIX
  - AWS SQS vs MS Azure Queue vs AMQP

VENUS-C Applications Requirements

- Blob
  - generic data item + metadata
- Message Queue
  - FIFO queue
- Key-value database
  - Aka NoSQL databases
  - Semantics depend on implementation

Data Access Strategies



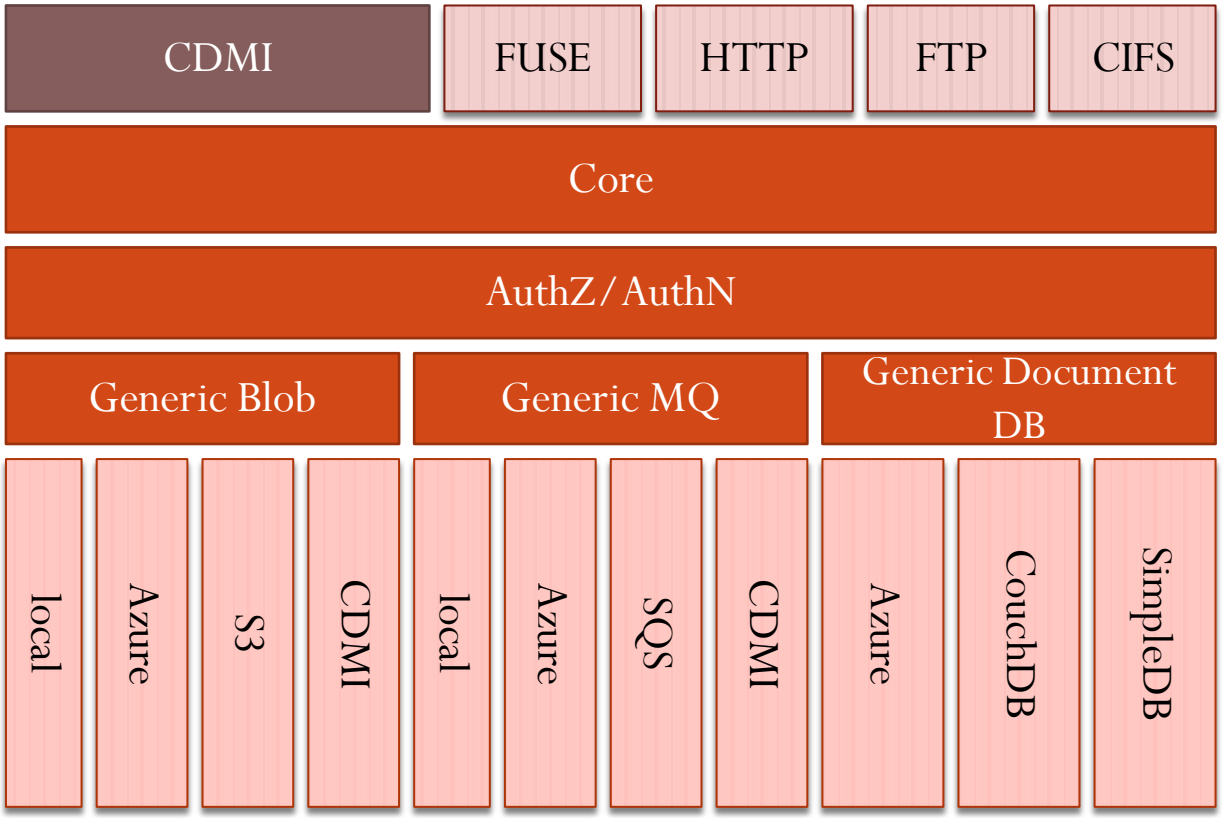
Motivation for a Proxy Approach

- Easier exposure of local storage through RESTful API
- Centralized control over resources
- Easier access to resources
  - Integration point with existing identity providers
- Easier release cycle. It is much easier to update a central CDMI-proxy service than a set of deployed libraries
- Optimization effect from optimizing data of multiple users can be higher than if optimized individually

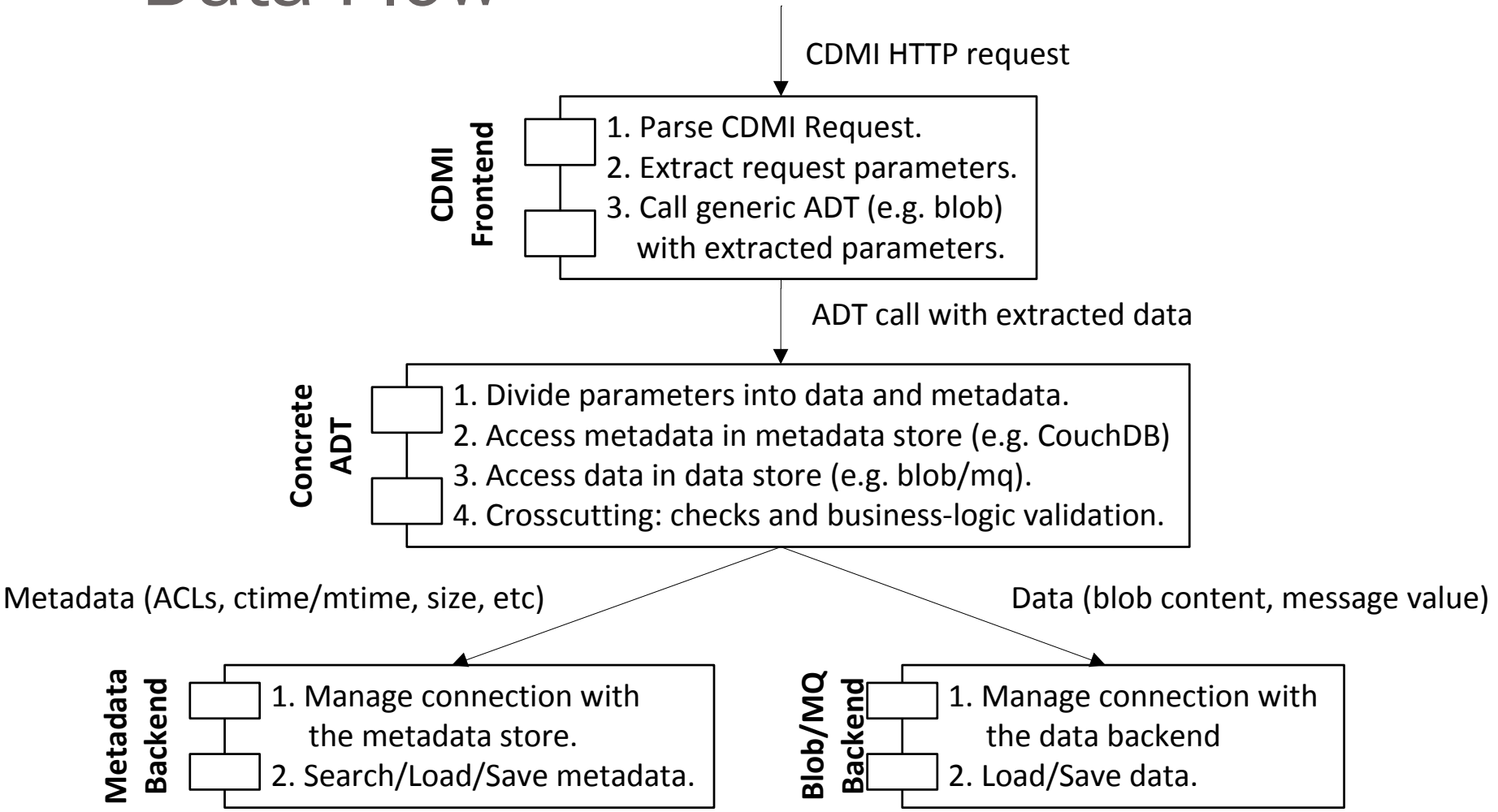
CDMI

- SNIA's Cloud Data Management Interface
  - http://www.snia.org/cloud
  - Standard (1.0.1h) + rising adoption by vendors
- CDMI provides an interface description for performing a set of operations on the data elements from the cloud
- CDMI objects:
  - Data
  - Queue
  - Container
  - Domain
  - Capability

CDMI-Proxy Structure



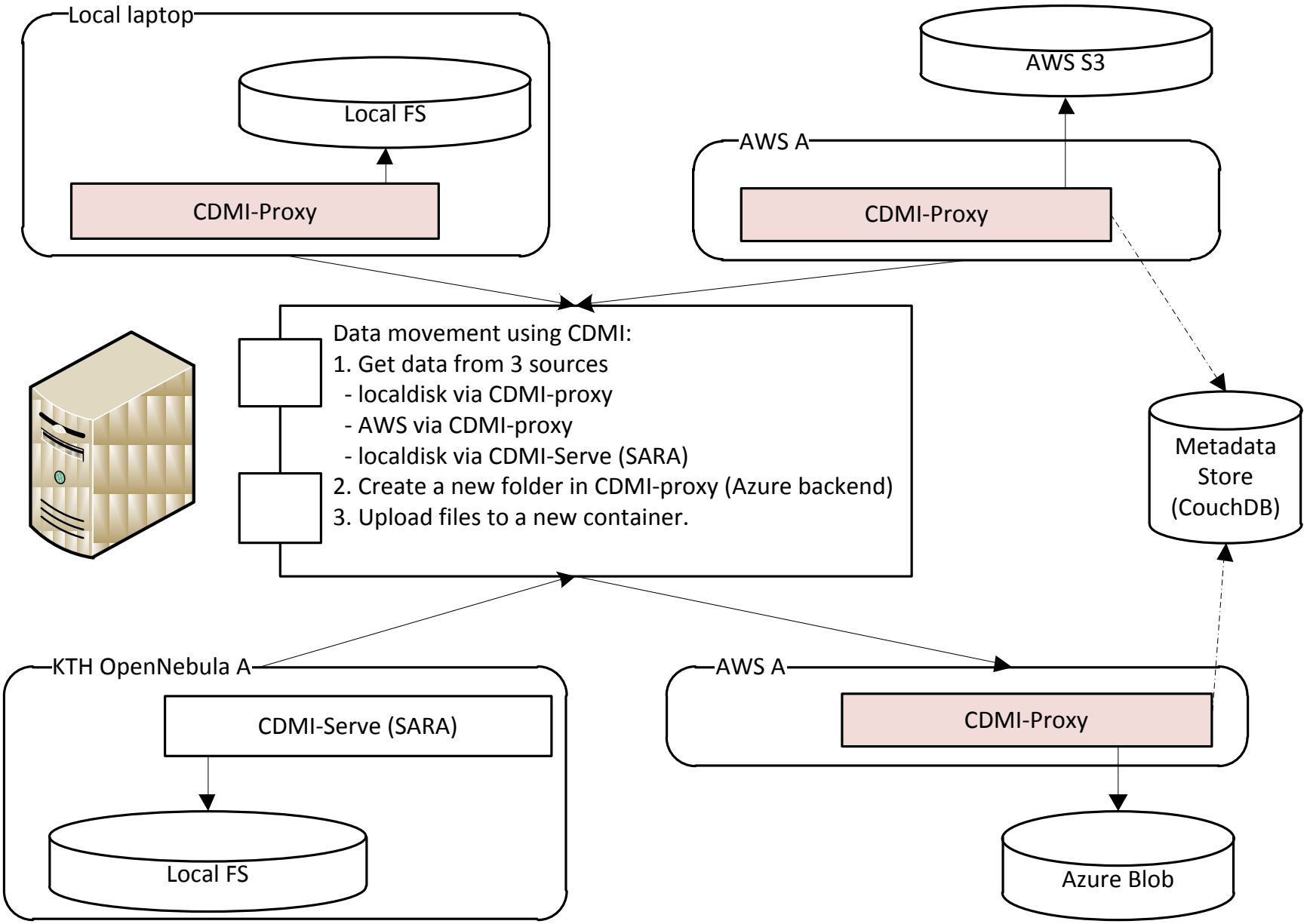
Data Flow


CDMI HTTP request
CDMI Frontend
1. Parse CDMI Request.
2. Extract request parameters.
3. Call generic ADT (e.g. blob) with extracted parameters.
ADT call with extracted data
Concrete ADT
1. Divide parameters into data and metadata.
2. Access metadata in metadata store (e.g. CouchDB)
3. Access data in data store (e.g. blob/mq).
4. Crosscutting: checks and business-logic validation.
Metadata (ACLs, ctime/mtime, size, etc)
Data (blob content, message value)
Metadata Backend
1. Manage connection with the metadata store.
2. Search/Load/Save metadata.
Blob/MQ Backend
1. Manage connection with the data backend
2. Load/Save data.

VENUS-C Deployment Models

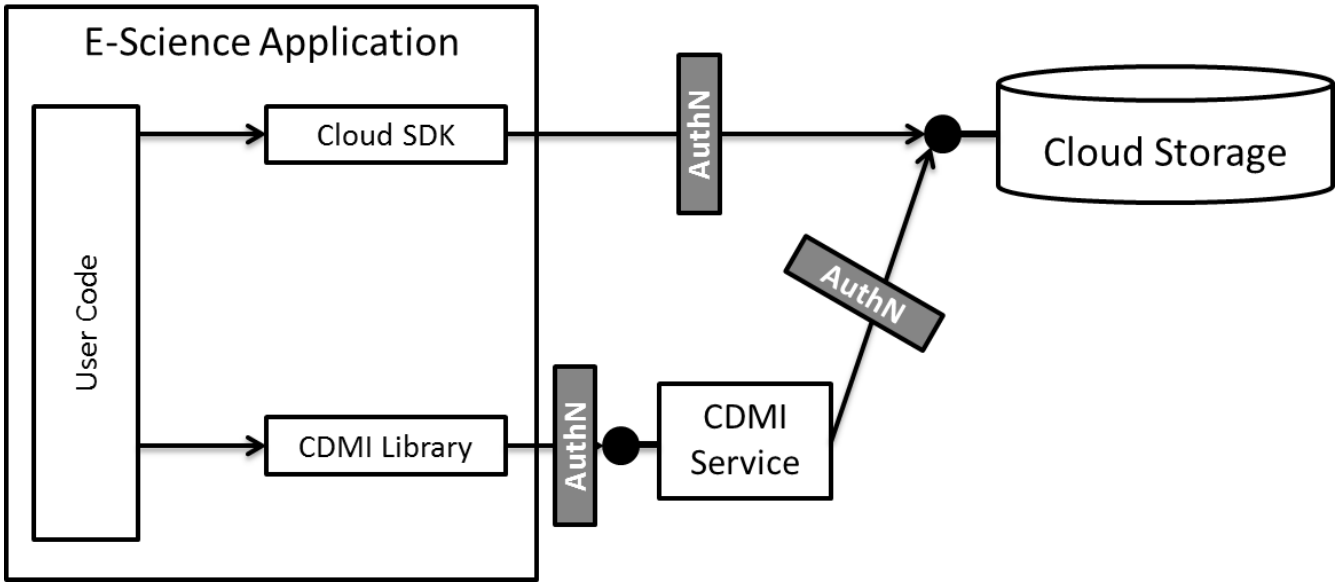
- Everything from the laptop
  - Client would need to have a business relationship with a cloud provider
- VENUS-C on-premises
  - E.g. VENUS-C services deployed at a research group
- VENUS-C in the cloud
  - E.g. a commercial offer to a company

Demo deployment



Security

- Crossing of trust domain
- Integration point with in-house
  - Identity providers
  - AuthZ systems



Client Side

- Developing CDMI SDKs in .Net, Java and Python, also exporting as CLIs
- Integration with EMIC's Generic Worker and BSC COMP Superscalar
- Community efforts
  - SARA
  - OCCI/CDMI demo from NetApp
  - (More are coming)
- Commercial offerings
  - Mezeo Cloud

Status and plans

- Core functionality is getting more mature
  - Supported ADTs: Blobs and Message Queues
  - Extended namespace for 1-level cloud storages (AWS S3, Azure Blob)
- Delivery of the first prototype is due in Autumn 2011
  - Prerelease earlier
- Will not expose document store via CDMI
  - Custom installations at DCIs with a shared security system
  - Will wait for CDMI specification

Roadmap

- Integration into application's workflows
  - Ongoing: bioinf, rendering, medical imaging
- Performance and stability testing
- 3rd party transfers with encryption of the content
- Enrichment of data items with (approximate) costs
- Basic accounting + interface to VENUS-C accounting and billing engine
- Dynamic credential passing to allow reuse of personal accounts

Technical Details

- CDMI-Proxy core
  - Twisted networking engine (Python)
  - Python 2.5+
- Backends
  - Metadata store: CouchDB (Azure Table, AWS SimpleDB)
  - Blobs: POSIX, Azure Blob, AWS S3, CDMI
  - MQ: AMQP, Azure Queue, AWS SQS, CDMI

Thank you!

http://github.com/livenson/vcdm
http://github.com/livenson/libcdmi-java
http://github.com/livenson/libcdmi-python