Enterprise NeuroSystem Group Meeting

Feb 7, 2022

Topics to Discuss

Approach to develop the catalog

- develop a light-weight catalog vrs modify existing catalogs (decision – develop a light weight catalog)
- Name of the current repo for coding (decision defer to activity 3 team)
- Dividing into teams for different activities (decision – teams in later part of the deck)
- Rescheduling of time for weekly calls. (decision -- change to 12 noon every Monday).

Augmenting Existing Catalogs

There is open-source software available for AI related catalogs

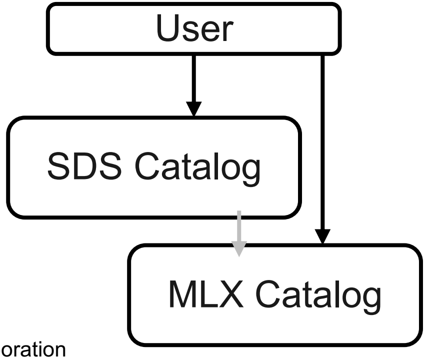
- For example, MLX Catalog
  - https://github.com/machine-learning-exchange/mlx

Does not provide explicit metadata, but supports four different types of digital assets

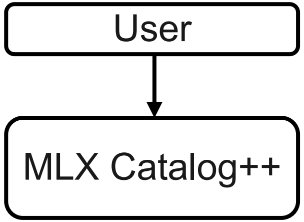
Two possible approaches:

- (a) A new SDS catalog stores links to existing catalog, provides access to the user
- (b) Modify and update code for MLX catalog to include metadata

(a)



(b)



Pros & Cons

Discuss Topic: Go with Option (a), Option (b) or defer decision as output of the first Sprint within working group.

Decision: Will adopt (a) but have a team study (b) and existing catalogs.

Option (a)

Pros:

- Easier independent software development
- Keep all ENG utilities in a single location
- More flexibility in development
- Can work with other catalogs
  - e.g. https://www.acumos.org/platform/
  - Existing products with catalogs

Cons:

- Proliferation of catalogs

Option (b)

Pros:

- Reuse of existing assets

Cons:

- Complexity of modifying existing code
- Inability to work with other catalogs easily

One activity to do should be to examine gaps and study other catalogs and understand architectural decisions.

Sprint 1 Teams

Activity 1:

- Flesh out use-cases
- Volunteers: **Tong Zhang (Intel),** Bill Wright (RH), Ravi Sinha (Jio), Dinesh Verma (IBM)

Activity 2:

- Study other open source catalogs
- Volunteers: Tong Zhang (Intel), Sanjay Aiyagari(RH), **Sathya Santhara (IBM)**, Theo Thomas (IBM)

Activity 3:

- Code design and development
- Volunteers: Austin Eovito, **Josh Purcell**, Rahul Batra, Shirley Han, Janki Vora, Amir Khanof, Dinesh Verma (all from IBM)

Name is bold is leader/coordinator of the activity.

Capability

Central Intelligence Group should develop the capability of a catalog of ***self-describing assets***

The catalog provides

- A REST interface for CRUD to a database of SDE (self-describing entries) and a database of opaque blobs
- Each SDE is a JSON record with two entries
  - “asset”: An URI for a digital asset (could be a URI for a blob in the catalog)
  - “metadata”: A metadata description
- Metadata description has following json structure “{“scope”: org-domain, “type”: string, “link”: blob}”
  - Link can have additional metadata details as desired by the publisher/context
  - <scope + type> defines the convention for the link and metadata description
  - Examples: {“owner”: slac.edu, “type”: “dataset”, “link”: ”None”}

Catalog Features

A Simple REST interface for a catalog

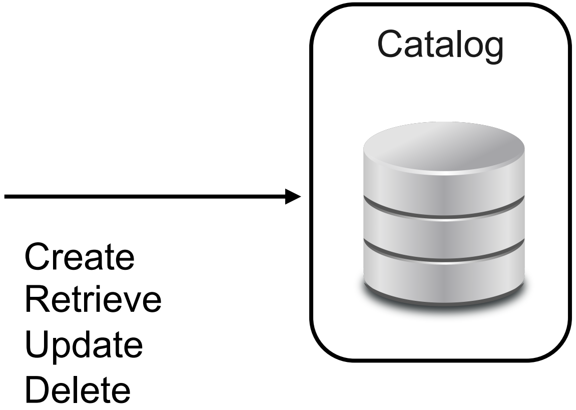
Component would support standard access control mechanisms

Component would be implemented in accordance with ‘Operate First’ principles

– Group will provide a container that can be deployed readily to obtain an instance

Guiding Principle

– Least number of additional specifications to support largest number of use-cases

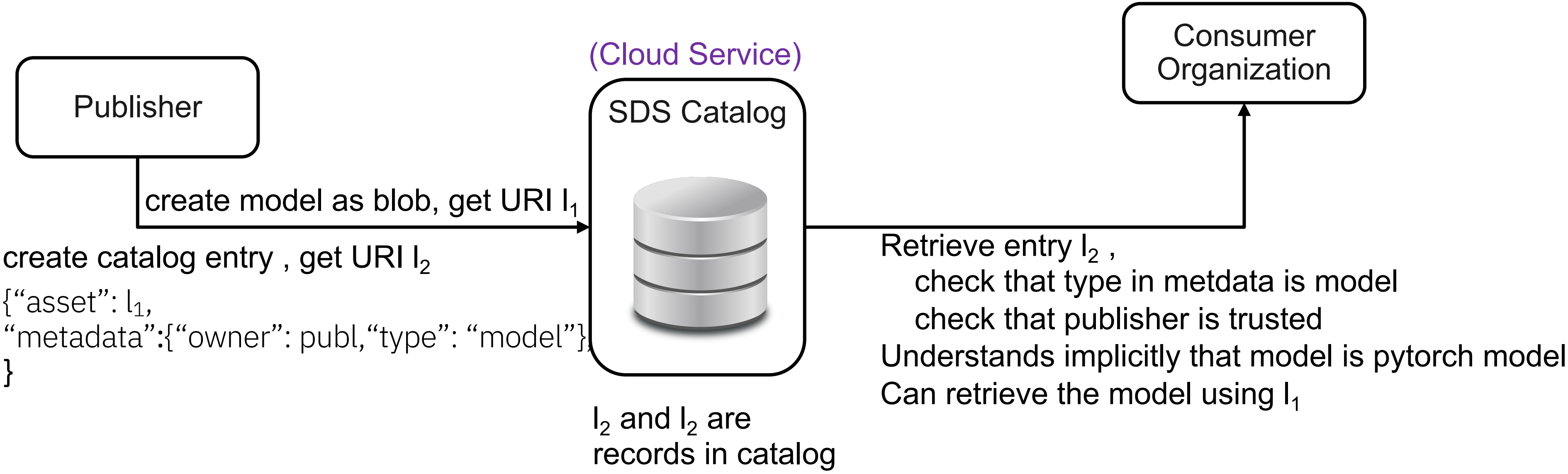


Usage Scenario 1

An organization creates an AI model and wants to share it with another organization

– Organizations have agreed upon their own acceptable vocabulary of types of model and type of acceptance

- An AI model is a Neural Network exported in pytorch



create model as blob, get URI I_1

create catalog entry , get URI I_2

{“asset”: l_1,
“metadata”:{“owner”: publ,“type”: “model”},
}

I_2 and I_2 are
records in catalog

Retrieve entry I_2 ,
check that type in metdata is model
check that publisher is trusted
Understands implicitly that model is pytorch model
Can retrieve the model using I_1

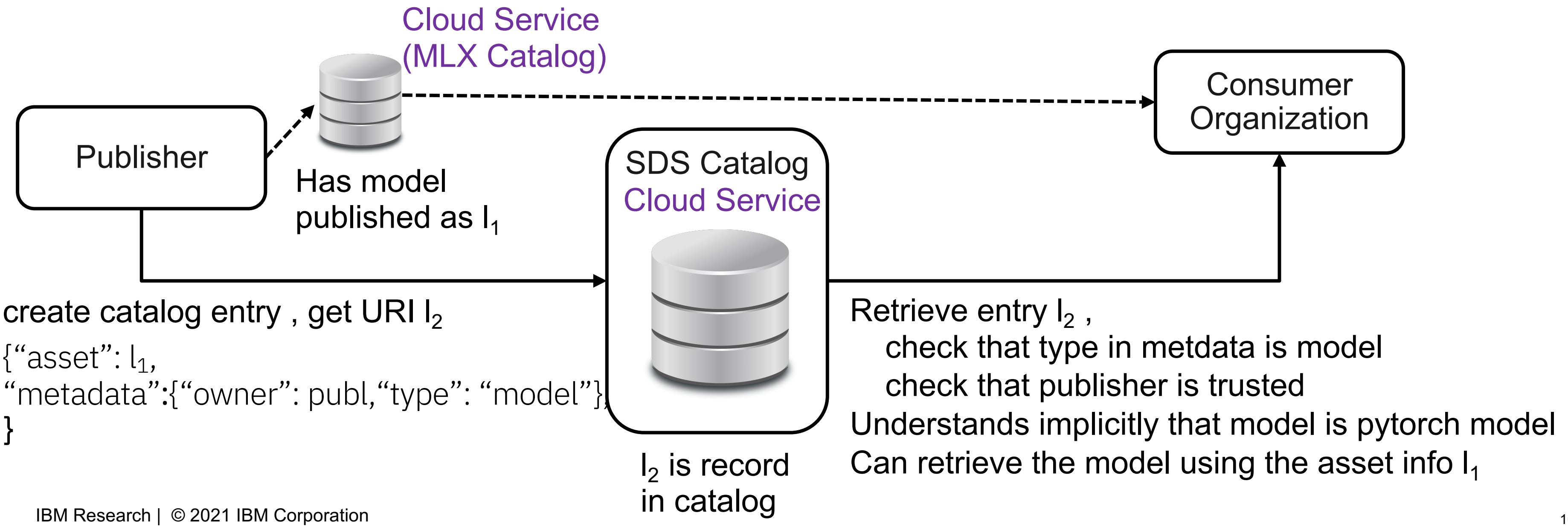
Usage Scenario 1

An organization creates an AI model and wants to share it with another organization

– Organizations have agreed upon their own acceptable vocabulary of types of model and type of acceptance

  - An AI model is a Neural Network exported in pytorch

– Organization has its own repository of assets



Cloud Service (MLX Catalog)

Publisher

Has model published as I_1

Consumer Organization

SDS Catalog Cloud Service

create catalog entry , get URI I_2

{“asset”: l_1,
“metadata”:{“owner”: publ,“type”: “model”},
}

Retrieve entry I_2 ,
check that type in metdata is model
check that publisher is trusted
Understands implicitly that model is pytorch model
Can retrieve the model using the asset info I_1

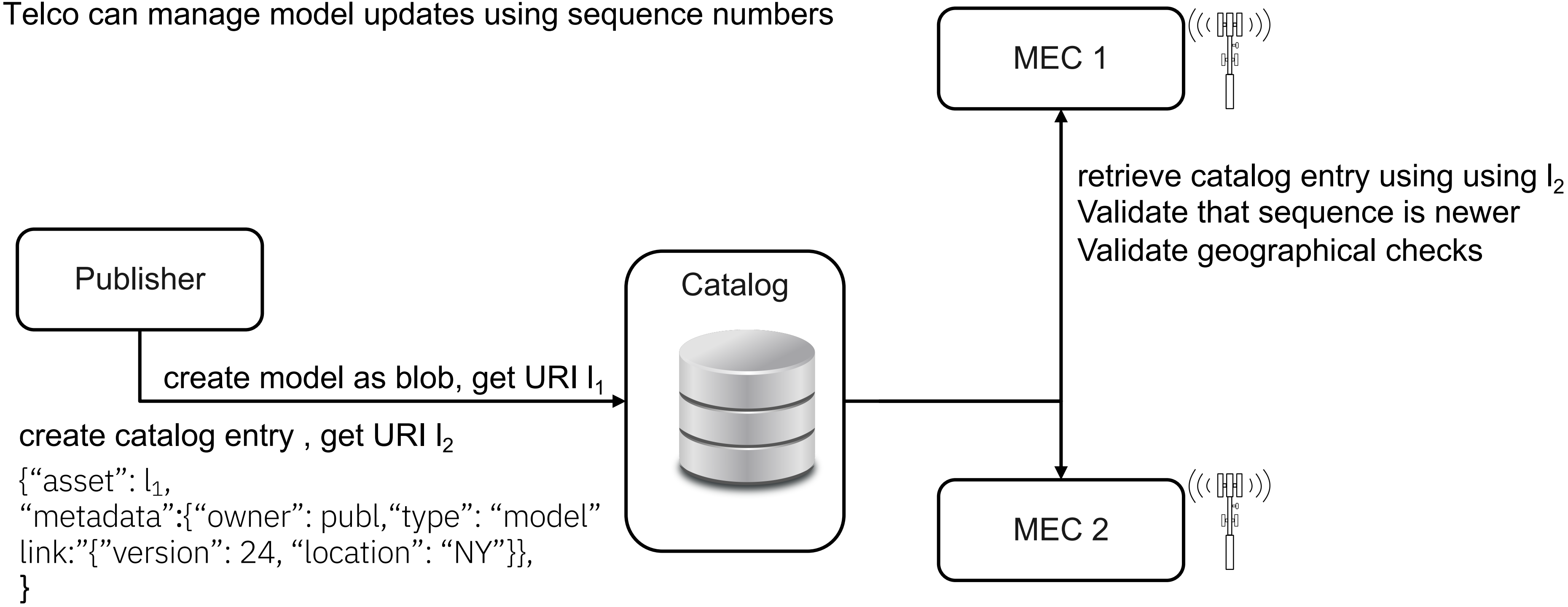
I_2 is record in catalog

Usage Scenario 3

A Telco publishes a model for Visual Analytics, it is to be used by MEC servers at base of its towers

– Telco adds additional information in metadata link (e.g. a sequence no, or a geographic boundary check}

Telco can manage model updates using sequence numbers



create model as blob, get URI I_1

create catalog entry , get URI I_2

{“asset”: l_1,
“metadata”:{“owner”: publ,“type”: “model”
link:”{”version”: 24, “location”: “NY”}},
}

retrieve catalog entry using using I_2
Validate that sequence is newer
Validate geographical checks

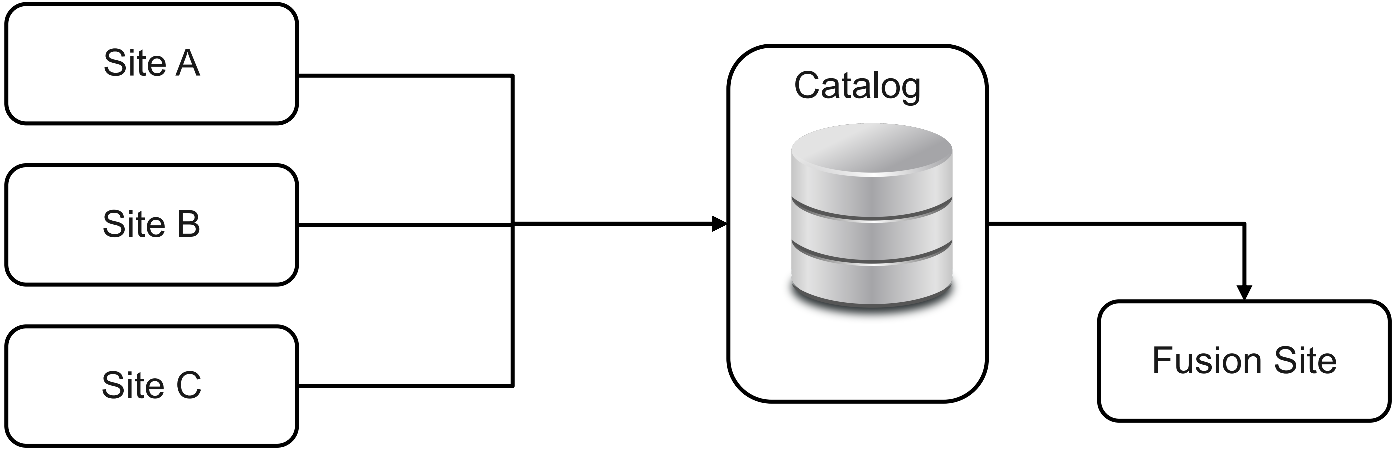
Usage Scenario 4

Federated Learning with several organizations and a Model Fusion Service

Each site publishes their models with round number in the catalog, along with environment for training

Fusion server picks up the entries with different rounds and merges them, republishing with catalog

- Fusion server could call transformations on retrieved model if it is in a different environment/architecture
- Metadata link would contain round no, link to merged model and version of models used from each
- Each site will remove their model once the version is merged


Site A
Site B
Site C
Catalog
Fusion Site