LF AI & DATA

The EGERIA Metadata Show

Egeria Webinar Program

How to use a repository proxy connector

Ljupcho Palashevski, ING
Egeria Maintainer

David Radley, IBM
Egeria Maintainer

Webinar program

3rd October 2022	15:00 UTC	**How to build a native repository connector**	Ever wanted to build an OMRS native repository connector? This session will take you though what the considerations are and you need to do. A native repository is a repository that contains native Egeria content (Entities and relationships and Classifications) and participates in Egeria cohorts. It will show how to create the simplest "Hello World" connector using XTDB as the main example. Zoom Conference https://zoom.us/j/523629111	**Chris Grote**
7th November 2022	15:00 UTC	**How to use a repository proxy connector**	Ever wanted to use an OMRS repository proxy connector A repository proxy connector is a wrapper around an existing 3rd party metadata store, that allows that 3rd party metadata store to participate in Egeria cohorts. This session takes you through how to use a repository proxy connector, so existing 3rd party metadata stores can benefit from being in the Egeria eco system. Zoom Conference https://zoom.us/j/523629111	**Ljupcho Palashevski** **David Radley**
December 2022	N/A	N/A	Northern hemisphere winter break	

Agenda

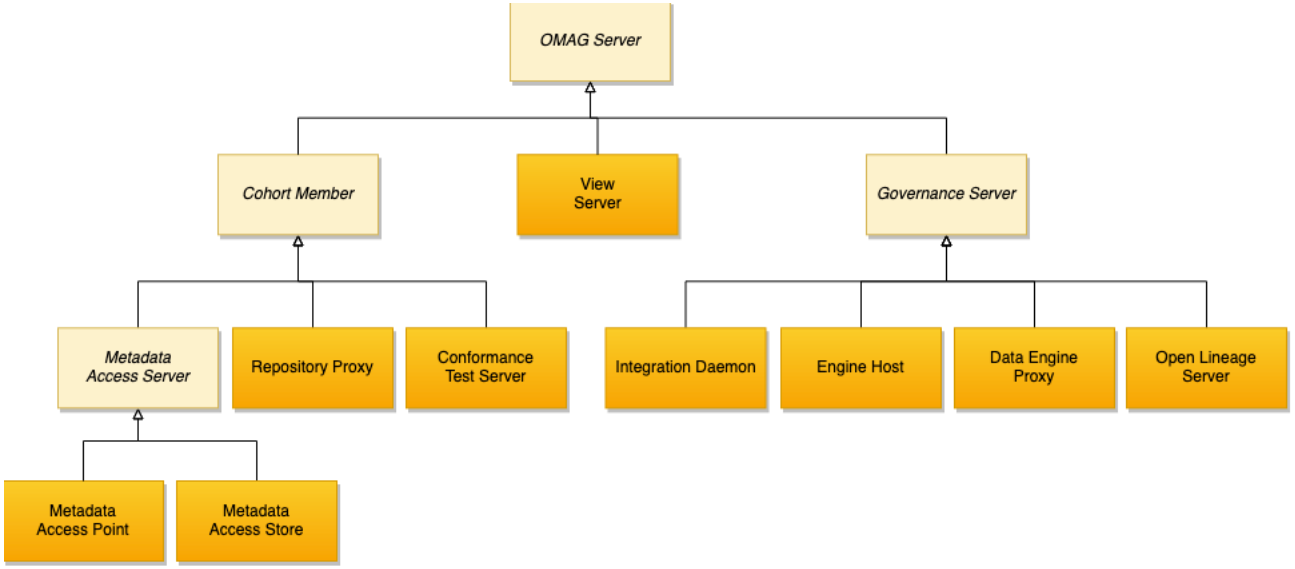
What is a Repository Proxy
Function, components, integration choices

Practical implementation – Using IBM IGC Repository Proxy connector
Technology capabilities and how those affect the cohort integration

Practical implementation – Using Caching Repository proxy connector
The need, how it works, usage pros and cons, current experience

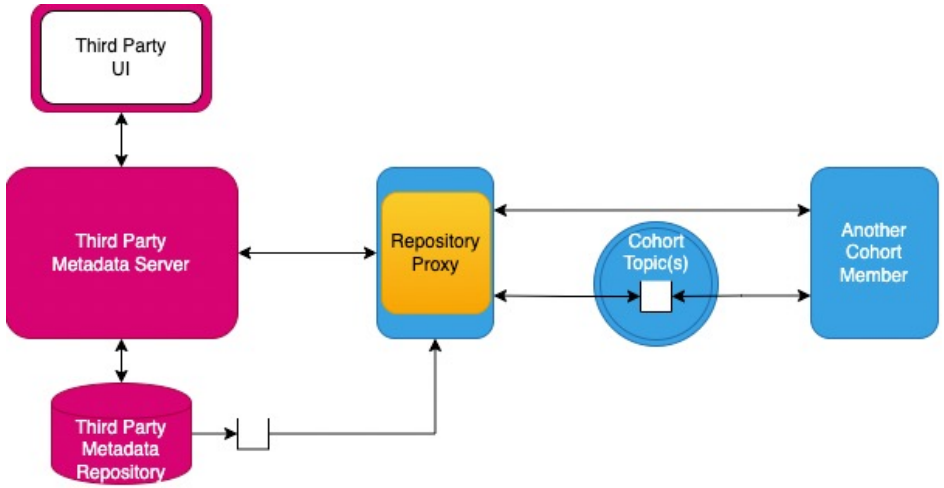
Other known implementations and their capabilities

What is a Repository Proxy



What is Repository Proxy (function)

- Metadata is present in different third party technologies and their respective repositories
- Repository Proxy acts as an adapter to the third party technology
- Brings third party metadata into Egeria Ecosystem



Repository Proxy components (connectors)

Repository connector

- Runs under the local repository service
- Provides standard access to the third party exposing it as egeria metadata collection
- Implements translation via type mappings
- Can implement limited caching logic

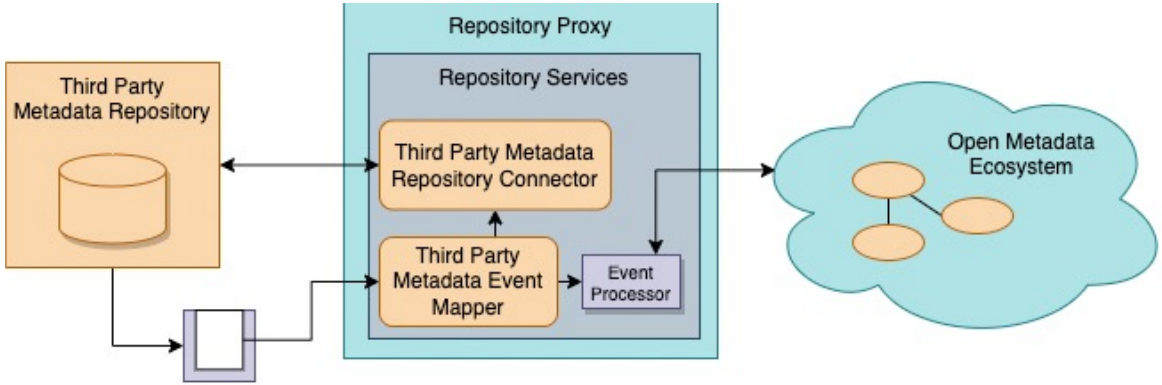
Event mapper connector

- Runs next to repository connector
- Processing inbound and outbound events to keep metadata in sync and consistent (enabling active integration)
- Maps proprietary events types to egeria omrs events

Repository Proxy components (connectors)

	Standard repository proxy style	Caching repository proxy style
Repository connector	Runs under the local repository service Provides standard access to the third party exposing it as egeria metadata collection Implements translation via type mappings Can implement limited caching logic Accesses 3rd party technology	Runs under the local repository service Provides standard access to the third party exposing it as egeria metadata collection ~~Implements translation via type mappings~~ Implements caching using an embedded native repository connector
Event Mapper	Runs next to repository connector Processing inbound and outbound events to keep metadata in sync and consistent (enabling active integration) Maps proprietary events types to egeria omrs events	Runs next to repository connector Implements translation via type mappings Polls 3rd party technology and then sends out batch events

Optimal integration style



- For optimal integration both connectors should be implemented as they are complementary
- Repository connector is required, **event mapper is optional** since it depends on third party technology capabilities
- Caching connector can compensate to some extent (discussed further in more details)

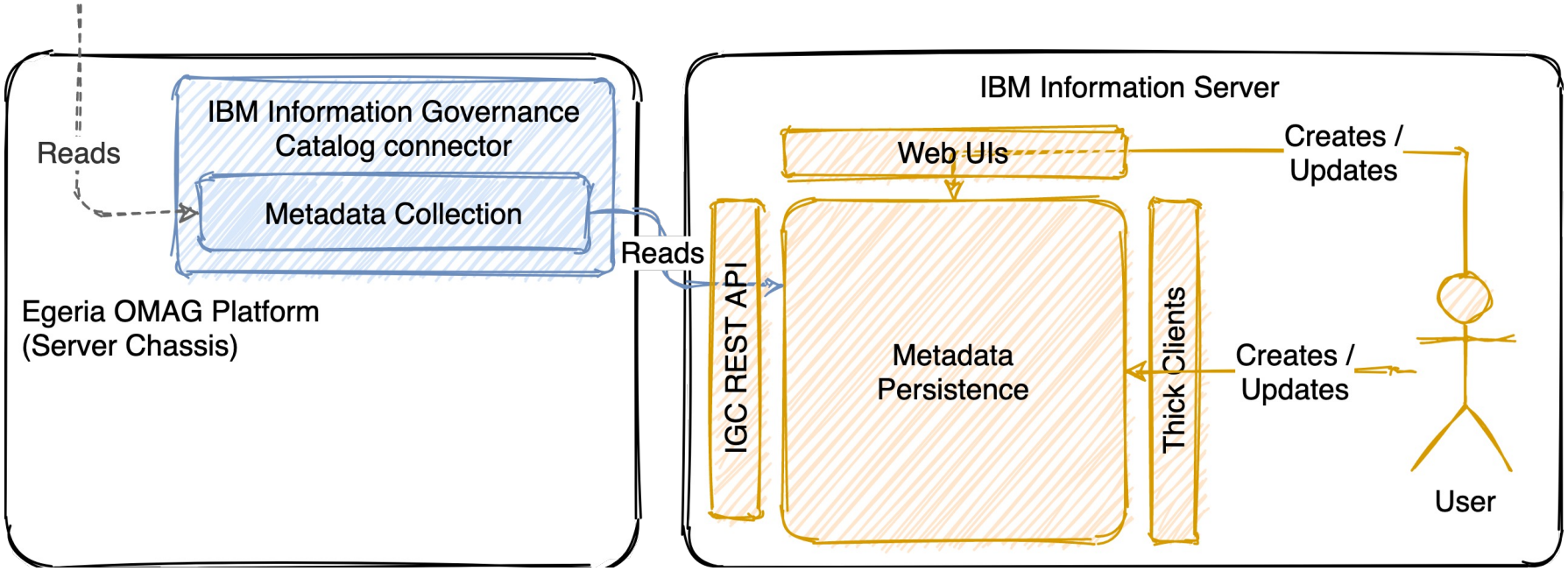
Using IGC Repository proxy connector

IGC Repository proxy

- https://odpi.github.io/egeria-connector-ibm-information-server/getting-started/igc/ - IGC Adapter - Repository connector for IBM IGC (Information Governance Catalog)

- Few technical characheristics
  - Java multi-tier application (service, engine, micro-service)
  - REST API for CRUD and advanced search operations
  - Limited event notification interface

How it works


Reads
IBM Information Governance
Catalog connector
Metadata Collection
Egeria OMAG Platform
(Server Chassis)
Reads
IBM Information Server
Web UIs
Creates /
Updates
IGC REST API
Metadata
Persistence
Thick Clients
Creates /
Updates
User

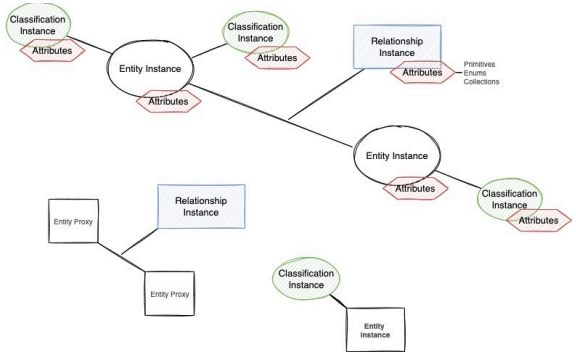
Limitations and choices

- No event mapper implemented
- No reference copies maintained

- What does it mean for the cohort
  - No notification to the rest of the cohort members
  - Notifications from the cohort members are ignored by the proxy (no support for immutable reference copies)
  - Metadata instances are visible only via federated queries retrieval through the metadata collection interface
  - Impact on performance

How do we compensate

- Metadata instances from IGC repository proxy are not stored as reference copies any more (because they cannot be keept up to date)
- Egeria can still maintain related metadata instances to guarantiee consistent and secure retrieval (anchors and last changed classifications)
- Classifications are stored separately now (for EntityProxies)
- Metadata can be still augmented elsewhere in the cohort (i.e. in a different local repository member of the cohort)



Using Caching Repository proxy connector

Requirement was connectors to 2 HMS implementations

IBM Cloud Data Engine
(formerly IBM Cloud SQL Query).

Egeria
connector

Egeria
connector

Metastore

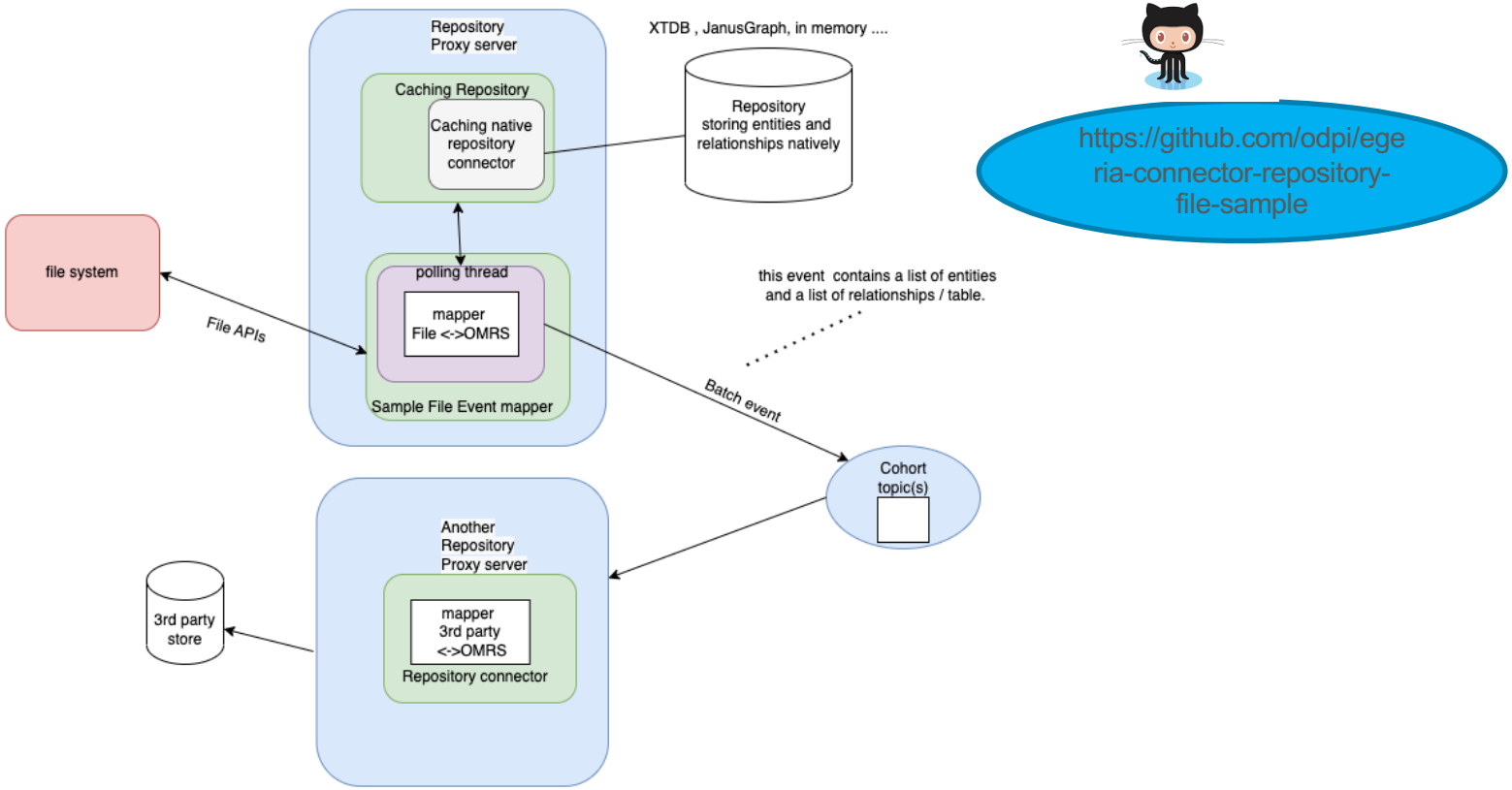
We decided to use an OMRS proxy connector pattern.

Restrictions:

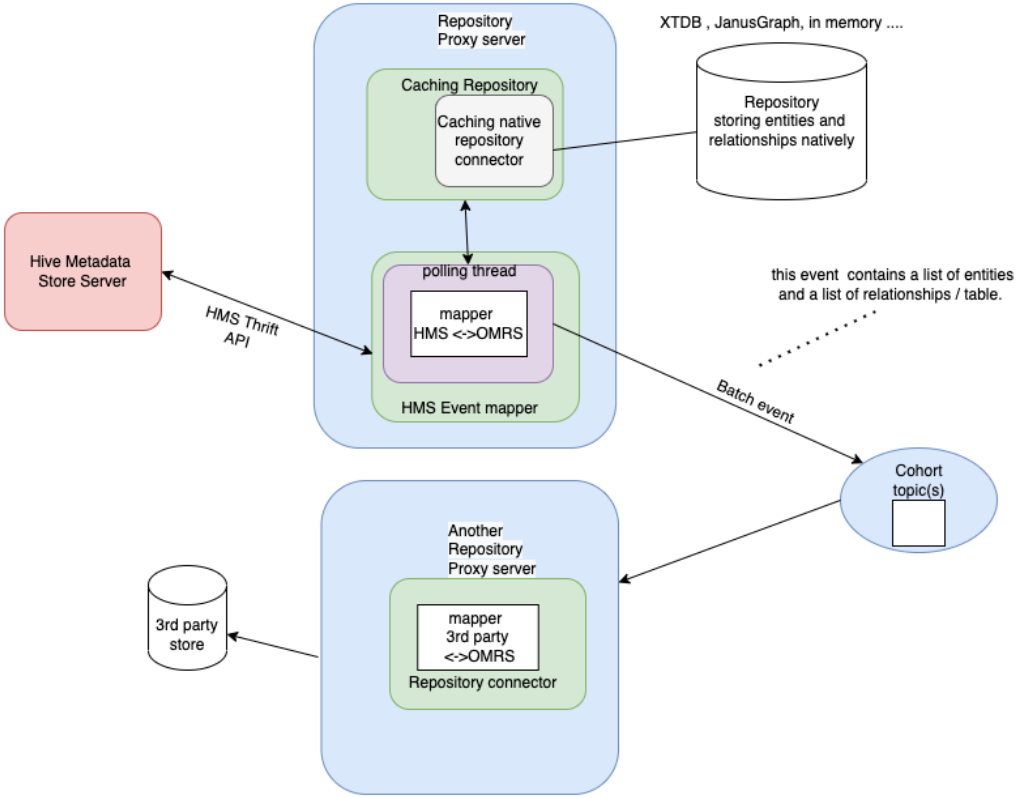
The recipient repository does not support Queries (only events)

The HMS repository is readonly

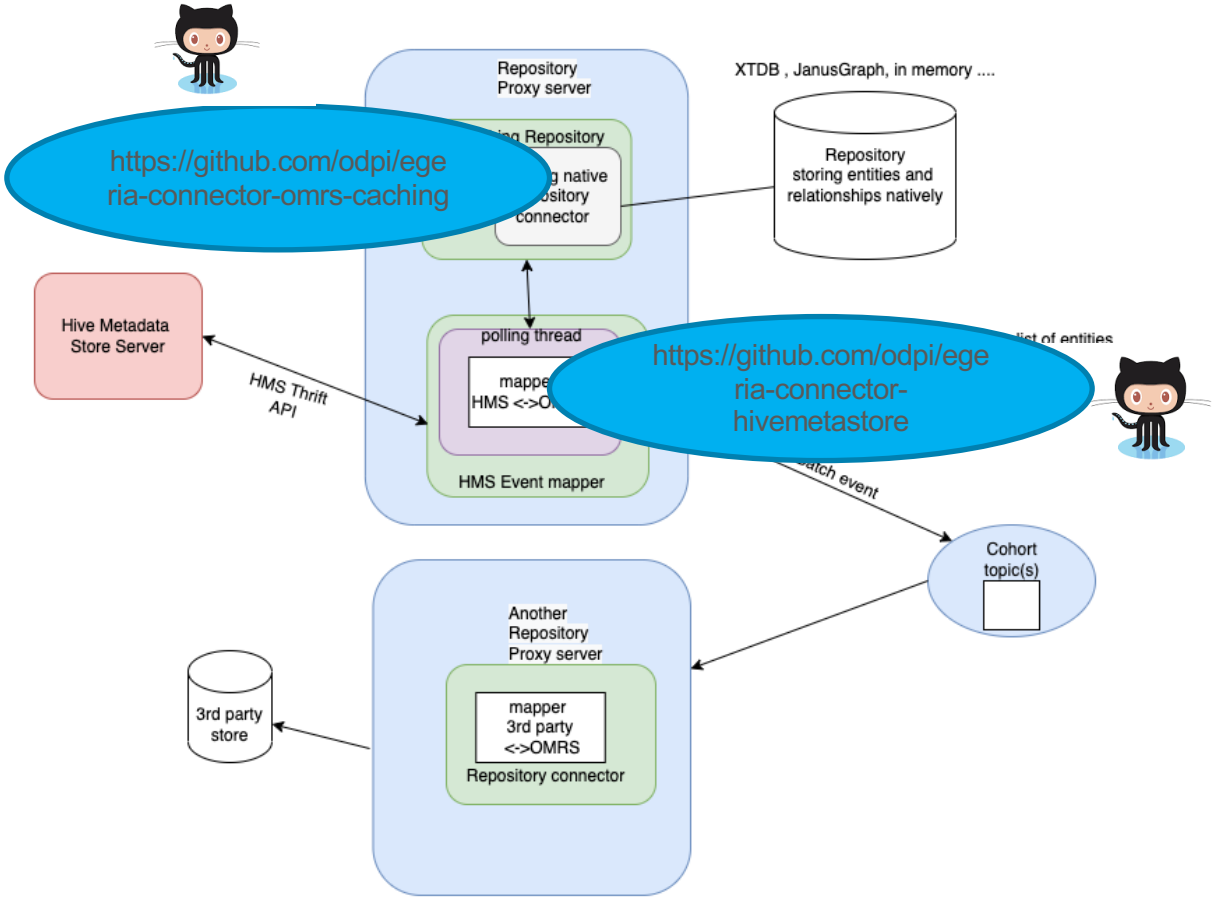
A new repository proxy pattern – tried first in a file sample



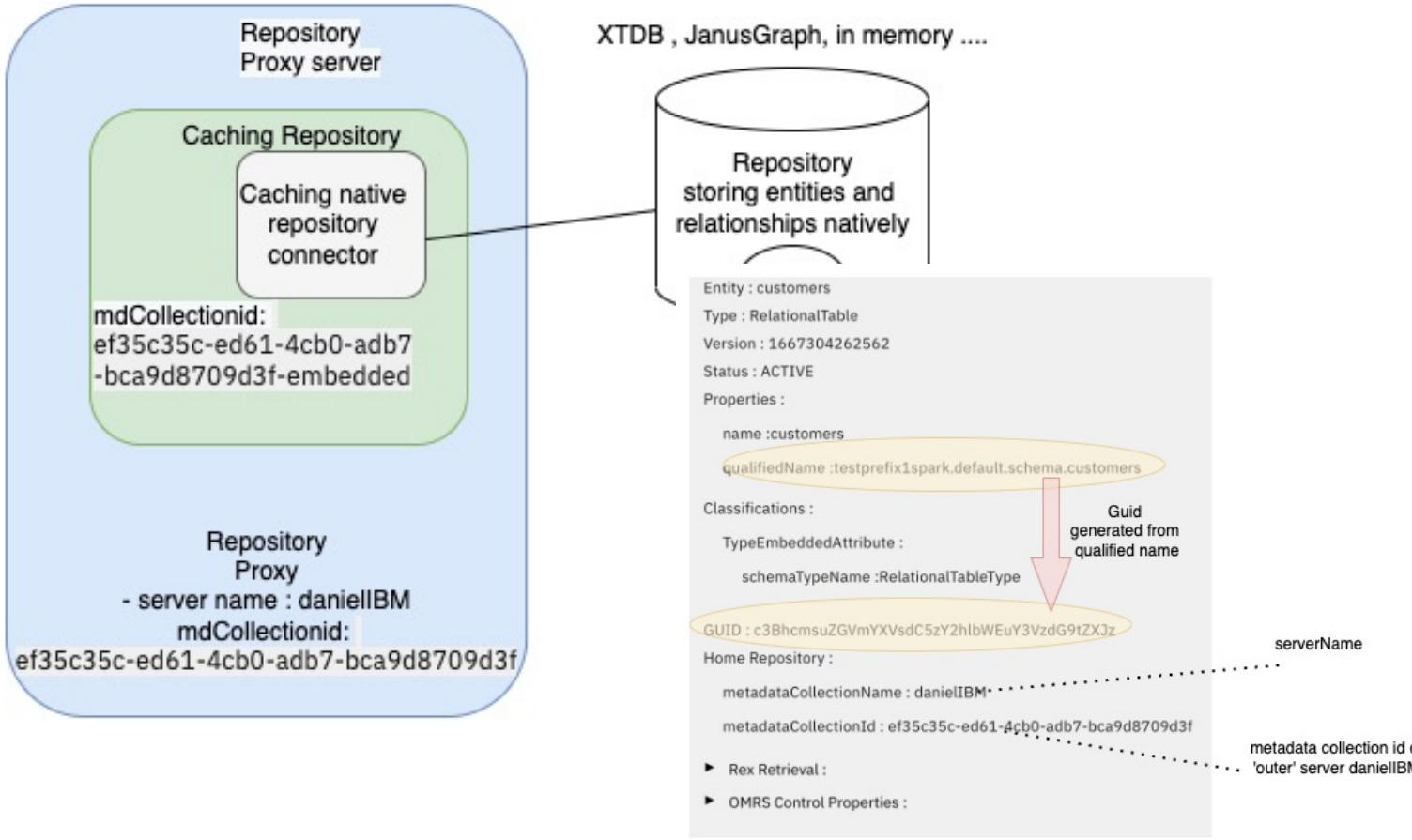
Connecting to HMS using a repository proxy by caching and polling



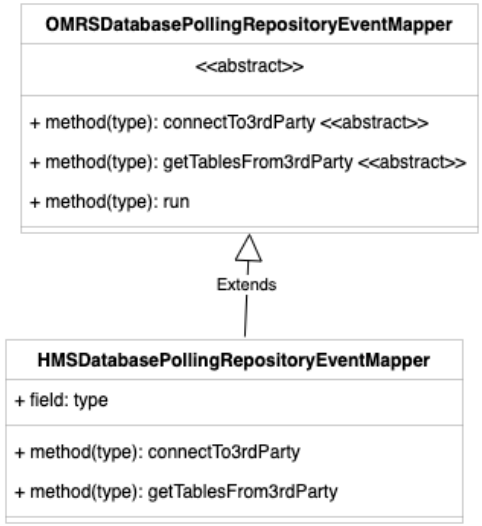
HMS Github repos



Caching


Repository
Proxy server
Caching Repository
Caching native repository connector
mdCollectionid:
ef35c35c-ed61-4cb0-adb7-bca9d8709d3f-embedded
Repository
Proxy
- server name : danielIBM
mdCollectionid:
ef35c35c-ed61-4cb0-adb7-bca9d8709d3f
XTDB , JanusGraph, in memory ....
Repository storing entities and relationships natively
Entity : customers
Type : RelationalTable
Version : 1667304262562
Status : ACTIVE
Properties :
name :customers
qualifiedName :testprefix1spark.default.schema.customers
Classifications :
TypeEmbeddedAttribute :
schemaTypeName :RelationalTableType
Guid generated from qualified name
GUID : c3BhcmsuZGVmYXVsdC5zY2hlbWEuY3VzdG9tZXJz
Home Repository :
metadataCollectionName : danielIBM
metadataCollectionId : ef35c35c-ed61-4cb0-adb7-bca9d8709d3f
serverName
metadata collection id o
'outer' server danielIBM
Rex Retrieval :
OMRS Control Properties :

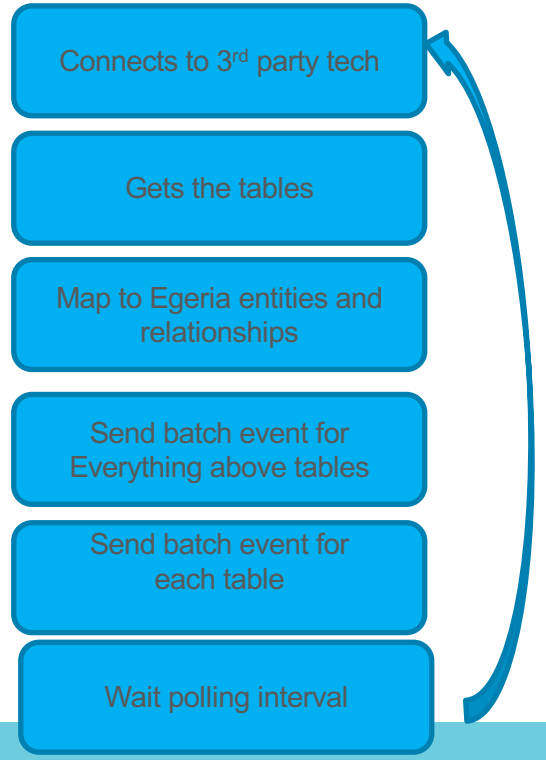
Polling



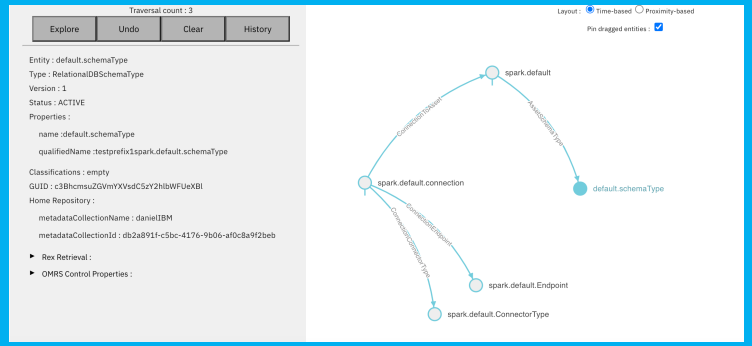
Technology independent code

Technology dependent code

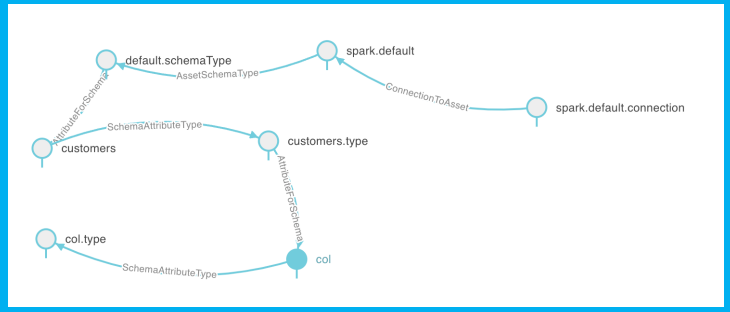
Polling thread loop



Batch events

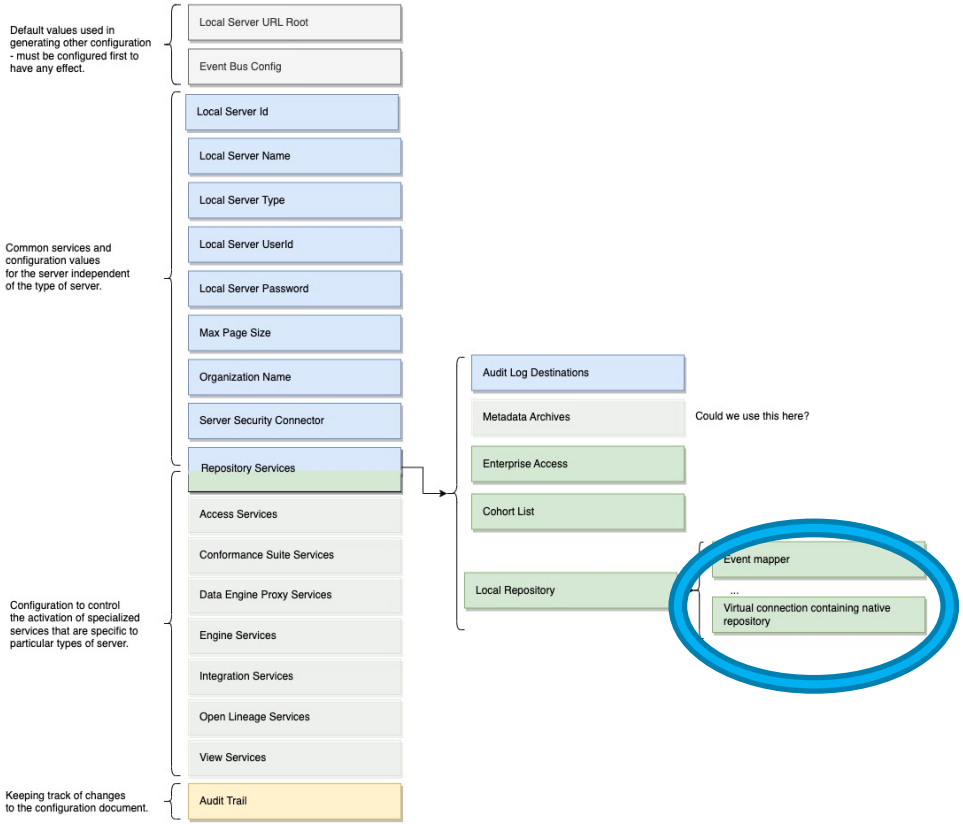


Above the table

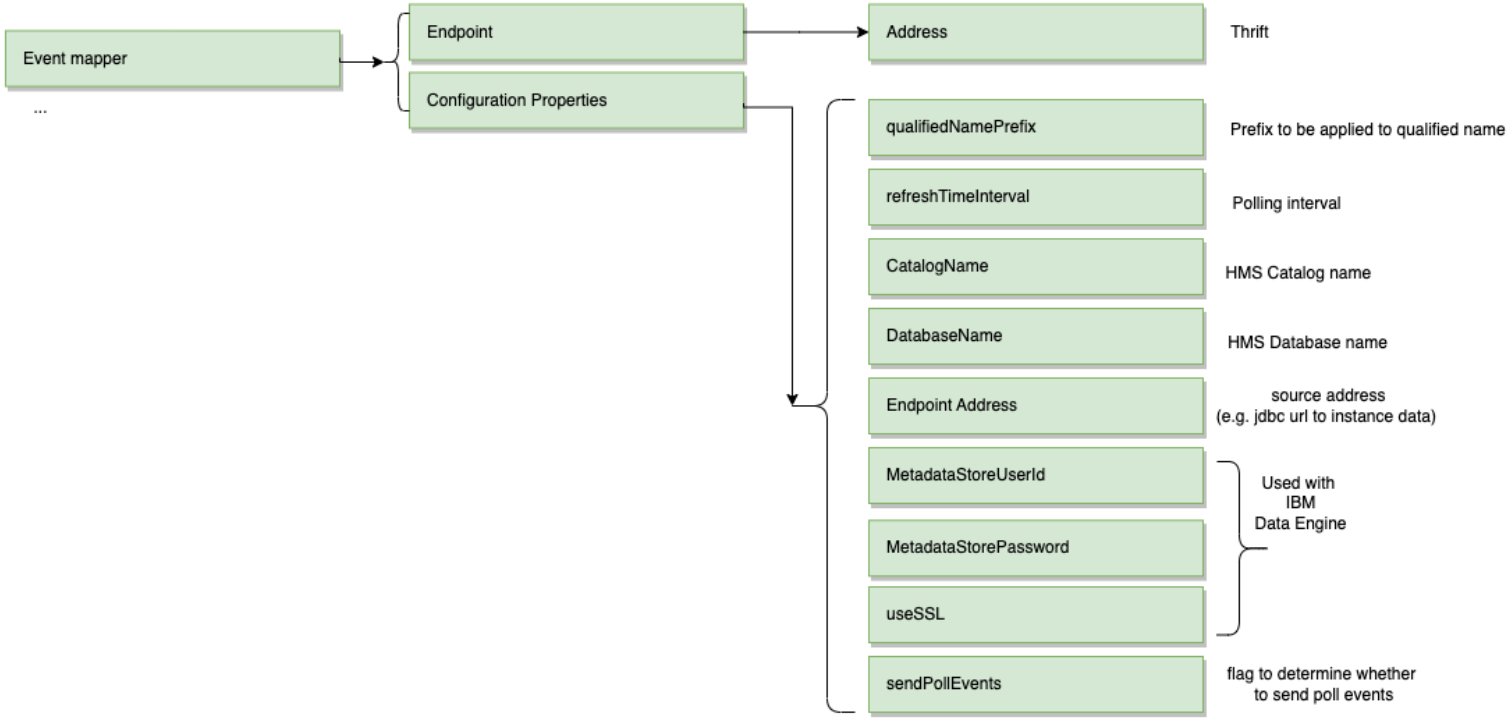


One for each table

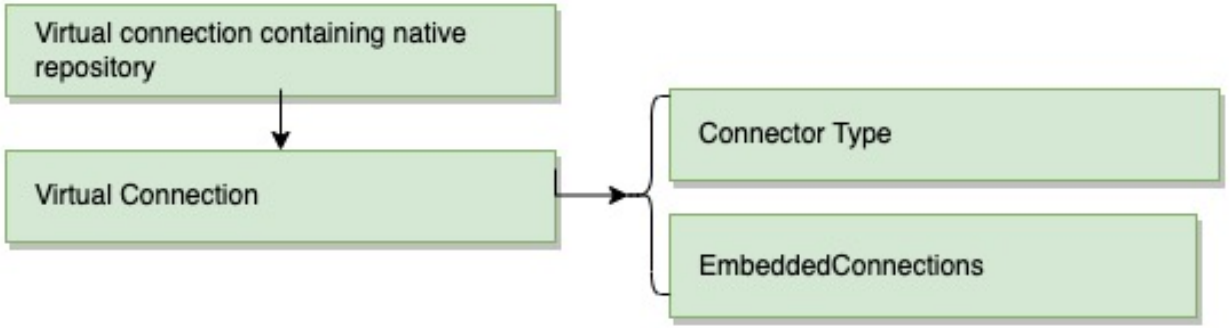
Configuration document for caching repository proxy



The HMS Event mapper configuration



The Repository caching connection configuration



This is where the native repo is

Pros and cons

+ simpler development

do not need to implement all the searches (40 often time consuming)

only need to populate the embedded repo at poll time. Simple 3rd party traversal and populate.

at query time, we do not need to query HMS and calculate identifiers of other connected elements.

+ Events will be well formed as they are from well tested repository connectors.

- Swamp the network every poll *
- Resolving the OMRS query to a Hive query real time could be more performant/ scalable in some larger HMS systems
- No delete support **

*The grabbing of all content could be done once then subsequent changes be made using incremental events

** polling logic could check what is now missing, and send delete events

Running with Data engine on IBM Cloud

- Data engine supplies a Hive compatible client. That allows a java program to connect into the Data Engine's HMS. The underlying data is stored in object storage not Hive.

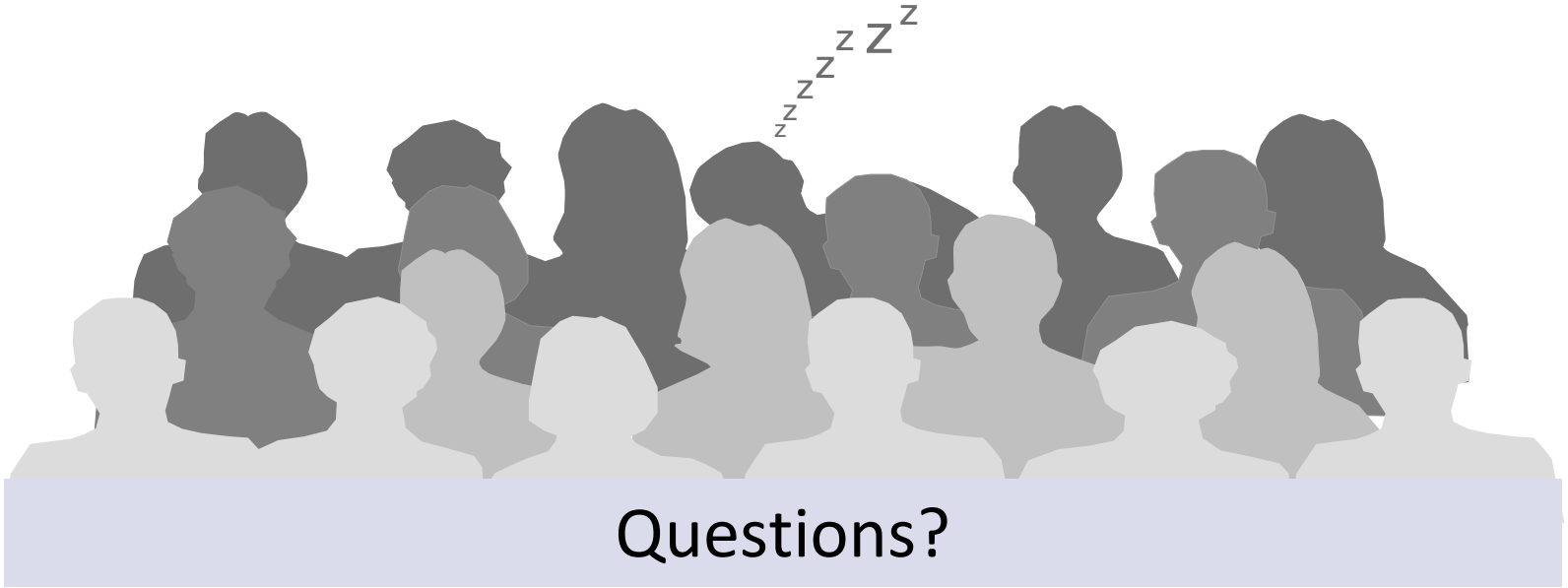
- We have a bash script that takes the vanilla Hive source amends it to download and incorporate the IBM client, to create a version of the connector that is compatible with IBM client.
- I hope to check this bash script into the open repository. This is an ongoing discussion.

Other known implementations and their capabilities

- Apache Atlas repository proxy connector
  - Open source connector with limited support
  - No Reference copies

- Microsoft Purview repository proxy connector
  - Technical preview based on Apache Atlas Repository connector
  - In early testing phase
  - Planned support as service part of Microsoft Azure Purview cloud solution

Open forum



THANK YOU!

https://egeria-project.org/concepts/repository-proxy/?h=repository+proxy#repository-proxy

https://odpi.github.io/egeria-connector-ibm-information-server/how-it-works/igc/

https://github.com/odpi/egeria-connector-omrs-caching