DataGEMS

Data Discovery Platform with Generalized Exploratory, Management, and Search Capabilities



OTHER

D5.3
D5.3a Dataset and Model Management Services

PU

28/11/2025

Disclaimer

This document contains description of the DataGEMS project findings, work and products. Certain parts of it might be under partner Intellectual Property Right (IPR) protection rules so, prior to using its content please consult carefully the licensing information of this document and/or contact the Project and the Editor for any further information.

In case you believe that this document harms in any way IPR held by you as a person or as a representative of an entity, please do notify the Project and the Editor immediately.

The editor and authors of this document have taken any available measure in order to ensure that its content is accurate, consistent and lawful. However, neither the project consortium as a whole nor the individual partners that implicitly or explicitly participated in the creation and publication of this document hold any sort of responsibility that might occur as a result of using its content. The information in this document is provided “as is” without guarantee or warranty of any kind, express or implied, including but not limited to the fitness of the information for a particular purpose. The user thereof uses the information at his/ her sole risk and liability.

This project has received funding from the European Union’s Horizon Europe framework programme under grant agreement No. 101188416.



Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Executive Agency, neither of which can be held responsible for those views and opinions and are not liable for any use that may be made of the information contained therein.

Document Information

Type	OTHER
Identifier	D5.3
Title	D5.3a Dataset and Model Management Services
Dissemination level	PU
Due Date	30/11/2025
Actual Delivery Date	28/11/2025
Lead Beneficiary	UVR
Other beneficiaries	CNRS, ZHAW, CITE, SCAYLE
Editor	Matteo Lissandrini (UVR)
Authors	Diamantis Tziotzios, Konstantinos Kakaletris, Georgios Kakaletris, Giorgos Papanikos (CITE), Bianca Benelli (UVR), Georgia Troullinou (CNRS)
Contributors	
Reviewer	Mike Xydas, Antonis Mandamadiotis (ARC)
Workpackage(s)	WP5 - Dataset and Model Management
Keywords	REST, API, Python, SQL, JSON, CSV, FastAPI
Version	1.0
Version information	Number of pages: 21 Last file save: 28/11/2025 9:46 PM File revision: 11 File name: D5.3_M11_D5.3a Dataset and Model Management Services

Copyright Information

License: Creative Commons Attribution (CC-BY)
This document is released under the Creative Commons Attribution (CC-BY) license, which allows for the free use, distribution, and adaptation of the work, provided proper attribution is given to the original author(s). By accessing or using this report, you acknowledge and agree to comply with the terms and conditions of the CC-BY license. For the full text of the license, please visit: https://creativecommons.org/licenses/by/4.0/legalcode

Document Version

Version	Date	Change Description	Editor
1.0	28/11/2025	Submitted to EU	Matteo Lissandrini

Document Tables

Table of Tables

Table 1 MoMa Task Orchestrator Component Location 9
Table 2 Cross-Dataset Discovery Component Location 15
Table 3 Dataset Workflow Orchestrator Component Location 17
Table 4 MoMa Object Catalog Component Location 18

Table of Figures

No table of figures entries found.

Table of Images

No table of figures entries found.

Table of Contents

Disclaimer 2
Document Information 3
Copyright Information 3
Document Version 4
Document Tables 5
Table of Contents 6
Abstract 7
1 Introduction 8
2 Software Components for Dataset and Model Management 9
2.1 MoMa Task Orchestrator 9
2.1.1 Component Overview and Location 9
2.1.2 MoMa Task Orchestrator in Action 9
2.2 PolyGlot Query Engine 15
2.2.1 Component Overview and Location 15
2.2.2 PolyGlot Query Engine Component in Action 15
2.3 Data Workflow Orchestrator 17
2.3.1 Component Overview and Location 17
2.4 MoMa Object Catalog 18
2.4.1 Component Overview and Location 18
3 Conclusions 19
Appendix I List of acronyms 20
Appendix II References 21

Abstract

This software release provides the implementation framework and initial functionality of the fundamental components implementing the services of the DataGEMS data-management infrastructure. This includes an initial set of core functionalities for the MoMa Task Orchestrator, the PolyGlot Query Engine, the Data Workflow Orchestrator, and the MoMa Object Catalog.

1 Introduction

This software release provides the implementation framework and initial functionality of the fundamental components implementing the services of the DataGEMS data-management infrastructure.

They are:

- The MoMa Task Orchestrator, which exposes the data and model management service API and handles all data-management operations, including accessing metadata, searching datasets registered in the system, managing user profiles, and providing access to metadata for other services and tasks.

- The PolyGlot Query Engine, which enables computation over heterogeneous data models through a REST API, integrates data via Virtual Knowledge Graphs, and executes queries required by analytical patterns to produce new MoMa datasets.

- The Data Workflow Orchestrator, which coordinates ingestion, maintenance, and operational workflows across the platform using Apache Airflow, and serves as the main execution and coordination engine for administrative processes.

- The MoMa Object Catalog, which stores and retrieves all metadata about datasets, models, and analytical patterns in a Neo4j database.

2 Software Components for Dataset and Model Management

2.1 MoMa Task Orchestrator

2.1.1 Component Overview and Location

The Data Management Platform will be accessible via the MoMa task orchestrator exposed by the data and model management service API. The service will accept and forward data management operations (e.g., read/write data profile metadata, discover and query datasets, obtain and update user profile, extract information from access logs to inform recommendation operations).

The Data Workflow Orchestrator has a REST API interface. It is accessible to internal services only via the DataGEMS Gateway (Deliverable 6.2 Platform Architecture Design Report). It is involved in the processes of accessing information residing in the MoMa Object Catalog, in the onboarding of new data by interacting with the Data Workflow Orchestrator, and in general in all the management operations of both data and metadata within the DataGEMS platform.

Table 1 MoMa Task Orchestrator Component Location

MoMa Task Orchestrator Component	
Service Name	data-model-management
Service Endpoint	https://datagems-dev.scayle.es/dmm/api/v1/dataset
Repository	https://github.com/datagems-eosc/data-model-management
Documentation	https://datagems-eosc.github.io/data-model-management
Visibility (currently)	PUBLIC
Visibility (planned)	PUBLIC
LICENSE	MIT License

2.1.2 MoMa Task Orchestrator in Action

Below are examples demonstrating how to interact with the dataset-related endpoints of the API. We describe MoMa-related cURL commands to onboard, update and then retrieve a dataset. The onboarding step is closely entangled to the 2.3 Data Workflow Orchestrator component, which ingests the dataset in s3.

NOTE: the cURL commands should be run after moving to the dmm-api folder.

2.1.2.1 Register a Dataset

Endpoint: POST /dataset/register

The first step of the onboarding process, performed by the Data Model Management API, is to register the dataset by storing its metadata in the MoMa Object Catalog. The endpoint validates the AP, ensuring it contains a single dataset node. In this initial phase, the dataset has status ‘staged’ to differentiate from other datasets that have already been profiled/post-processed/etc. and have status ‘ready’. Two datasets with the same UUID cannot be registered. Trying to do so would result in a 409 error with the message ‘*Dataset with ID … already exists in the MoMa Object Catalog’*.

Command:

```
curl -X POST -H "Content-Type: application/json" --
data @../tests/register/oasa.json https://datagems-
dev.scayle.es/dmm/api/v1/dataset/register | python -
m json.tool
```

The API returns:


```
{
   "code":201,
   "message":"Dataset with ID fa512835-7ec9-41cc-87db-a94b99e8b97
registered successfully in Neo4j",
   "ap":{
      "nodes":[
         {
            "@id":"4ecb7e5b-eb82-4ae6-8354-5a2943702fcd",
            "labels":[
               "Analytical_Pattern"
            ],
            "properties":{
               "Description":"Analytical Pattern to register a
dataset",
               ...
            }
         },
         {
            "@id":"69ce4693-e71e-4616-9320-037c90a88858",
            "labels":[
               "DataModelManagement_Operator"
            ],
            "properties":{
               "Description":"An operator to register a dataset
into DataGEMS",
               ...
            }
         },
         {
            "@id":"fa512835-7ec9-41cc-87db-a94b99e8b97",
            "labels":[
               "sc:Dataset"
            ],
            "properties":{
               "@type": "sc:Dataset",
               "archivedAt": "s3://scratchpad/fa512835-7ec9-41cc-
87db-a94b99e8b97"
            }
```

```
        },
        {
            "@id":"38b5aafb-184d-4b1e-9e9e-5541afca2c96",
            "labels":[
                "User"
            ]
        },
        {
            "@id":"dca293c0-e20c-47de-be58-acad8b8c423c",
            "labels":[
                "Task"
            ],
            "properties":{
                "Description":"Task to register a dataset",
                ...
            }
        }
    ],
    "edges":[
        {
            "from":"4ecb7e5b-eb82-4ae6-8354-5a2943702fcd",
            "to":"69ce4693-e71e-4616-9320-037c90a88858",
            "labels":[
                "consist_of"
            ]
        },
        {
            "from":"69ce4693-e71e-4616-9320-037c90a88858",
            "to":"fa512835-7ec9-41cc-87db-a94b99e8b97",
            "labels":[
                "input"
            ]
        },
        {
            "from":"38b5aafb-184d-4b1e-9e9e-5541afca2c96",
            "to":"69ce4693-e71e-4616-9320-037c90a88858",
            "labels":[
                "intervene"
            ]
        },
        {
            "from":"dca293c0-e20c-47de-be58-acad8b8c423c",
            "to":"4ecb7e5b-eb82-4ae6-8354-5a2943702fcd",
            "labels":[
                "is_accomplished"
            ]
        },
        {
            "from":"38b5aafb-184d-4b1e-9e9e-5541afca2c96",
```

```
            "to":"dca293c0-e20c-47de-be58-acad8b8c423c",
            "labels":[
               "request"
            ]
         }
      ]
   }
}
```


2.1.2.2 Load a Dataset

Endpoint: PUT /dataset/load

The second step of the onboarding process is moving the dataset from the scratchpad to its permanent storage location under s3://dataset/UUID. The endpoint then updates the dataset attribute “archivedAt” (which represents where the dataset is) both in the AP and MoMa Obejct Catalog, to store the location change in s3.

If the dataset does not exist in the scratchpad, the cURL will result in an error with message ‘*Source dataset not found at expected location*’, since the file does not exist in the scratchpad.

Command:

```
curl -X PUT -H "Content-Type: application/json" --
data @../tests/load/oasa.json https://datagems-
dev.scayle.es/dmm/api/v1/dataset/load | python -m
json.tool
```

The API returns:


```
{
   "code":200,
   "message":"Dataset moved from s3://scratchpad/fa512835-7ec9-
41cc-87db-a94b99e8b97 to s3://dataset/fa512835-7ec9-41cc-87db-
a94b99e8b97",
   "ap":{
      "nodes":[
         {
            "@id":"c401292a-4c7a-4e7c-9856-682f626ee1ef",
            "labels":[
               "Analytical_Pattern"
            ],
            "properties":{
               "Description":"Analytical Pattern to load a
dataset",
               ...
            }
         },
```

```
        ...
          }
}
```

2.1.2.3 Update a Dataset

Endpoint: PUT /dataset/update

When the dataset information is first stored in MoMa Object Catalog, it only has user-provided metadata attached to it. Afterwards, profiling (D2.3a Dataset Profiling Services) extract more information, like the file location and file content metadata. The updated profile(s) for the dataset(s) is added to MoMa Catalog.

The update works only if a dataset is already stored in the MoMa Object Catalog, otherwise it results in a 404 error with message '*Dataset with ID … does not exist in the MoMa Object Catalog'*.

Command:

```
curl -X PUT -H "Content-Type: application/json" -
data
@../tests/update/dataset_profile/oasa_light.json
https://datagems-
dev.scayle.es/dmm/api/v1/dataset/update | python -m
json.tool
```

The API returns:

```
{
   "code":200,
   "message":"Dataset with ID fa512835-7ec9-41cc-87db-
a94b99e8b97 updated successfully in Neo4j",
   "ap":{
      "nodes":[
         {
            "@id":"a8bbe300-c7f2-429c-83fd-ecafda705c90",
            "labels":[
               "Analytical_Pattern"
            ],
            "properties":{
               "Description":"Analytical Pattern to update
a dataset",
               ...
            }
         },
      ...
        }
}
```

2.1.2.4 Get a Dataset

Endpoint: GET /dataset/dataset_id

This endpoint retrieves a dataset from the MoMa Object Catalog by its ID. If the dataset does not exist, it returns a 404 error.

Command:

```
curl -X GET -H "Content-Type: application/json"
https://datagems-dev.scayle.es/dmm/api/v1/dataset/
fa512835-7ec9-41cc-87db-a94b99e8b97 | python -m
json.tool
```

The API returns:

```
{
{
    "code": 200,
    "message": "Dataset with ID
cd3a2f580e24ccd002b70696b5c9e4ca39042cf4b584a7d854a67fa7e055b201
retrieved successfully from Neo4j",
    "dataset": {
       ...
    }
}
```

2.1.2.5 Filter the Datasets

Endpoint: GET /dataset

The endpoint requests metadata of datasets from the MoMa Object Catalog that allows optional sorting and filtering by dataset IDs, properties to include, node types, published date ranges, and status. The request parameters may expand as the needs of the DataGEMS platform evolve.

Command:

```
curl -X GET -H "Content-Type: application/json"
https://datagems-
dev.scayle.es/dmm/api/v1/dataset?publishedDateFrom=2
024-01-01&publishedDateTo=2025-01-01 | python -m
json.tool
```

The API returns:

```
{
    "code": 200,
    "message": "Datasets retrieved successfully",
    "datasets": {
```



2.2 PolyGlot Query Engine

2.2.1 Component Overview and Location

The Polyglot Query Engine will be able to perform computations on multiple data models and data files, and it will be made available through a Restful API. The service will follow the MoMa model and use Virtual Knowledge Graphs to integrate data. This will support query executions for the tasks of in- and cross-dataset-discovery (D3.3a Data Discovery Services). The API endpoint will take a JSON file of an Analytical Pattern as input (with information regarding the user, the task to achieve, the MoMa operator to use and the datasets to query), decompose and execute the appropriate data processing steps, and save the results of the query as a new dataset, which will be accessible to the user.

The PolyGlot Query engine has a REST API interface. It is accessible by internal services only via the DataGEMS Gateway (Deliverable 6.2). It will access information residing in the MoMa Object Catalog and then use this information in the execution of queries in either SQL or SPARQL over the data stored in the platform.

Currently it only uses information stored in the MoMa Object Catalog modelled as a metadata knowledge graph and can query relational data stored either as CSV files or in a SQL DBMS (here Postgres). In a future release it will have access to more advanced information to allow cross-model queries.

Currently it uses DuckDB as query engine to allow querying multiple storage representations. In future releases it will use OnTop to allow querying across data models.

Table 2 Cross-Dataset Discovery Component Location

PolyGlot Query Engine Component	
Service Name	polyglot-query-engine
Service Endpoint	https://datagems-dev.scayle.es/dmm/api/v1/polyglot
Repository	https://github.com/datagems-eosc/data-model-management
Documentation	https://datagems-eosc.github.io/data-model-management
Visibility (currently)	PUBLIC
Visibility (planned)	PUBLIC
LICENSE	MIT License

2.2.2 PolyGlot Query Engine Component in Action

Endpoint: POST /polyglot/query

Executes the SQL query described in the Analytical Pattern provided as input. The SQL query can span over one or more datasets either in CSV files or stored in a relational DB. The query is

executed using DuckDB. The result of the query is stored in CSV form and then it is registered in the MoMa Object Catalog.

Command:

```
curl -X POST -H "Content-Type: application/json" --data @../tests/query/query_before.json https://datagems-dev.scayle.es/dmm/api/v1/polyglot/query | python -m json.tool
```

The API returns:


```
{
   "code":200,
   "message":"Query executed successfully, results stored at /s3/data-model-management/results/2c830b06-a1da-48ca-a982-15062002797c",
   "ap":{
      "nodes":[
         {
            "@id":"a51f3e82-ca74-4ef6-8d1e-2bb08f4df6cf",
            "labels":[
               "Analytical_Pattern"
            ],
            "properties":{
               "Description":"Analytical Pattern to query a dataset",
               ...
            }
         },
      ...
        }
}
```

2.3 Data Workflow Orchestrator

2.3.1 Component Overview and Location

The Data Workflow Orchestrator is a core service in the DataGEMS platform responsible for managing and executing various data-related workflows, including ingestion, maintenance, and system operations. As a platform's workflow engine, it coordinates complex, multi-step processes that are essential for the smooth functioning and upkeep of the data infrastructure.

It is built on top of the widely adopted Apache Airflow engine, leveraging a robust and extensible framework for defining, scheduling, and monitoring workflows. This service is the primary entry point for administrative operations, offering centralized control and visibility over data pipelines and operational tasks across the platform.

The Data Workflow Orchestrator exposes its capabilities through Airflow's native HTTP API. This API is not publicly accessible. Instead, all external interactions pass through the Gateway API, where requests are fully authenticated and authorized before reaching the orchestrator. When the Data Workflow Orchestrator communicates with other internal components through their exposed HTTP API endpoints, it does so using authorized requests established through OIDC flows provided by the DataGEMS AAI service, ensuring secure and consistent identity and access management across the platform.

Table 3 Dataset Workflow Orchestrator Component Location

Data Workflow Orchestrator Component	
Service Name	data-workflow
Repository	https://github.com/datagems-eosc/dg-data-workflow
Documentation	https://datagems-eosc.github.io/dg-data-workflow
Visibility (currently)	PUBLIC
Visibility (planned)	PUBLIC
LICENSE	Apache v2.0

2.4 MoMa Object Catalog

2.4.1 Component Overview and Location

The MoMa Management service is a middleware that provides access to the MoMa Object Catalog and is responsible for managing the MoMa graph in Neo4j. The service relies on Cypher queries to manage MoMa objects supporting insertion, updates, and deletion within the catalog. It exchanges information with other components using JSON-formatted data.

The MoMa Management component provides a REST API, implemented with FastAPI, for interacting with the catalog. Access is exclusively available to internal DataGEMS services. It stores and provides information residing in the MoMa Object Catalog.

Currently the service stores metadata for datasets and their contained resources and allows retrieving this information using various filtering parameters. It also provides functionality for updating metadata and deleting complete datasets. In future releases, the service will also manage metadata about Machine Learning models and Analytical Patterns, with support for retrieving them through flexible filtering options.

Table 4 MoMa Object Catalog Component Location

MoMa Object Catalog Component	
Service Name	moma-management
Repository	https://github.com/datagems-eosc/moma-management
Documentation	https://datagems-eosc.github.io/moma-management/
Visibility (currently)	PUBLIC
Visibility (planned)	PUBLIC
LICENSE	MIT License

3 Conclusions

This release provides an initial deployment of core infrastructure components.
These components are being integrated in the DataGEMS platform at different levels.
Further, more functionalities are going to be added to make it fully functional.
At this stage, the components cover the main functionalities needed to register datasets within the infrastructure along with their metadata, generate additional metadata through the profiling service, make this metadata available to other services, and enable SQL querying across datasets with a relational data-model, regardless of their underlying storage format.

Appendix I List of acronyms

Acronym	Description
API	Application Programming Interface
SQL	Structured Query Language
AP	Analytical Pattern

Appendix II References