



MARCLAIMED

Open interfaces

Deliverable 2.4

WP2

Author: Luc Gasser

Date: 22/11/2024



**Funded by
the European Union**

GRANT AGREEMENT NUMBER	101136799		
ACRONYM / FULL TITLE	MARCLAIMED: integrated Decision Support Tool for Reliable and Affordable Application of Managed Aquifer Recharge with Alternative Water Resources in River Basin and Drought Management Plans		
START DATE	01/02/2024	DURATION	36 months
END DATE	31/01/2027		
PROJECT URL	https://marclaimed.eu/		
DELIVERABLE	D2.4		
WORK PACKAGE	WP2		
CONTRACTUAL DATE OF DELIVERY	31/01/2025		
ACTUAL DATE OF DELIVERY	28/01/2025		
NATURE	OTHER	DISSEMINATION LEVEL	PU
LEAD BENEFICIARY	EGM		
RESPONSIBLE AUTHOR	Luc GASSER		
CONTRIBUTIONS FROM	Luc GASSER (EGM) Franck Le Gall (EGM)		
ABSTRACT	This document provides specifications on a key component of the MARCLAIMED IDST: the Orchestration API, that aims to facilitate communication between the different tools, the NGSI-LD context broker, and the graphical user interface (GUI) that end-users will interact with. This system allows for the seamless orchestration of complex workflows, making it easier for users to configure, execute, and analyze the outcomes of the various tools. It provides an in-depth overview of the workflows and conventions associated with the registration and execution of DataServiceProcessing instances within the MARCLAIMED system. These workflows are essential for ensuring the proper configuration of each tool, as well as their subsequent execution in a manner that aligns with the user’s goals. By detailing the processes involved, this document aims to give clear guidance on how the tools interact with the broader IDST infrastructure to enable effective water resource management and decision-making.		

Disclaimer

Any dissemination of results reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains.

© **MARCLAIMED Consortium, 2024**

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both. Reproduction is authorised provided the source is acknowledged.

Copyright message

TABLE OF CONTENTS

EXECUTIVE SUMMARY	5
1.0 Introduction	6
2.0 Overview of the orchestration interface	7
2.1 The MARCLAIMED tools	7
2.2 Purpose of the orchestration API	8
2.3 Definition of the conventions	9
3.0 DataServiceProcessing registration workflow	12
4.0 DataServiceProcessing execution workflow	15
4.1 Triggering of the processing	15
4.2 Retrieval of input data in the IDST data core	16
4.3 Conversion data and actual execution of the service	17
5.0 Interaction with the DataServiceOutput	20
5.1 The DataServiceOutput entity	20
5.2 Interaction with results	21
5.3 How to link output attributes to their source entities	21
5.4 How to link a DataServiceOutput to its input data	22
6.0 CONCLUSIONS	23
7.0 ANNEX: NGSI-LD payloads	24

EXECUTIVE SUMMARY

The MARCLAIMED project is an innovative initiative aimed at addressing the increasing challenges of water scarcity and quality in Europe, particularly in the context of climate change. The project focuses on enhancing the sustainability and resilience of water supply systems through the integration of Managed Aquifer Recharge (MAR) with Alternative Water Resources (AWR). By developing and implementing advanced tools for water management, MARCLAIMED seeks to ensure the availability of clean water for various uses, protect ecosystems, and promote economic viability in the face of environmental stressors.

This document is a deliverable within the scope of the MARCLAIMED project and is produced as part Work Package 2, which is dedicated to the development and implementation of the Integrated Decision Support Tool (IDST). The document describes some major workflows involved in the IDST, with a particular focus on two essential processes: DataServiceProcessing registration and the execution of DataServiceProcessing. These processes ensure that the tools developed within MARCLAIMED can be efficiently configured, executed, and analyzed by end-users for decision-making in water resource management.

The purpose of this document is to define the conventions and workflows associated with the registration and execution of DataServiceProcessing instances in the IDST, ensuring that users can easily interact with the system, configure the tools, and execute them to obtain actionable insights. By providing detailed specifications for these workflows, this deliverable supports the integration of DataService into the IDST, contributing to the broader goal of facilitating effective, data-driven decision-making for water management across Europe.

1.0 Introduction

The MARCLAIMED project represents an innovative approach to tackling the pressing issues of water scarcity and quality that are increasingly affecting regions across Europe, particularly in light of climate change. As part of this initiative, the project integrates advanced solutions for water resource management through the development of six specialized tools. These tools are designed to address various aspects of water management, including water quality monitoring, drought prediction, and the efficient allocation of alternative water resources (AWR). The overarching goal of the project is to enhance the sustainability and resilience of water supply systems, ensuring that clean water is available for all uses while safeguarding ecosystems and supporting economic stability.

A critical element of the MARCLAIMED project is its Integrated Decision Support Tool (IDST), which enables stakeholders to make informed, data-driven decisions regarding water resource management. Central to the IDST is the Orchestration API, a key component that facilitates communication between the different tools, the NGSI-LD context broker, and the graphical user interface (GUI) that end-users will interact with. This system allows for the seamless orchestration of complex workflows, making it easier for users to configure, execute, and analyze the outcomes of the various tools.

This document provides an in-depth overview of the workflows and conventions associated with the registration and execution of DataServiceProcessing instances within the MARCLAIMED system. These workflows are essential for ensuring the proper configuration of each tool, as well as their subsequent execution in a manner that aligns with the user's goals. By detailing the processes involved, this document aims to give clear guidance on how the tools interact with the broader IDST infrastructure to enable effective water resource management and decision-making.

2.0 Overview of the orchestration interface

2.1 *The MARCLAIMED tools*

The MARCLAIMED project is an innovative initiative aimed at addressing the critical challenges of water scarcity and quality in Europe, particularly in the context of climate change. By integrating Managed Aquifer Recharge (MAR) with Alternative Water Resources (AWR). The overarching goal is to develop and implement advanced tools and methodologies that facilitate effective water resource management, ensuring the availability of clean water for various uses while protecting ecosystems and promoting economic viability. The following summaries briefly describe each one of the 6 tools developed within the project :

- **MARVIS (RES1):** A virtual sensing tool for near real-time water quality monitoring. It predicts nitrate concentration in Alternative Water Resources (AWR) using virtual sensors that exploit correlations between monitored parameters with AI algorithms. It aims to provide early warning indicators and prevent contamination of aquifers.
- **PREDMAR (RES2):** An AI-based tool for forecasting water availability and demand. This tool has a different approach in each demo site. This includes: forecast of the volume stored in reservoirs, forecast of daily flows in rivers, forecast of water resources availability, forecast the volume of groundwater available to be recovered, as well as identification of the surface water surpluses at basin scale to assess the feasibility of new MAR schemes.
- **MARAISE (RES3):** A tool for optimizing MAR performance in both short and long-term perspectives. It provides risk indicators based on water quality and hydrodynamic data, improving operational efficiency. It integrates data from various sources to optimize recharge volumes based on water availability and demand forecasts.
- **OASIS FLOW (RES4):** A digital tool that assesses the impact of MAR with AWR at the municipal scale. It evaluates water availability, demand, and the economic and social implications of MAR strategies, providing indicators that support local water governance discussions.
- **MARINSURE (RES5):** An ensemble forecasting tool that assesses the socioeconomic performance of AWR strategies under various future scenarios. It uses a modular approach to measure the economic value of water security provided by MAR, helping to address supply-demand imbalances.
- **RECOVER (RES6):** A tool focused on tariff setting for cost recovery. It analyzes current economic approaches to water service and compares them with successful global examples. It aims to design effective tariff components based on water availability, recovery efficiency, and economic value, facilitating implementation in local contexts.

2.2 Purpose of the orchestration API

The orchestration API serves as an extension of the NGSI-LD context broker (referred to as IDST Data Core in the MARCLAIMED project), which is the main component of the MARCLAIMED Integrated Decision Support Tool (IDST). While the NGSI-LD API facilitates data interaction and management at a technical level, end users will primarily interact with a graphical user interface (GUI) designed for ease of use. This document outlines the conventions and specifications for the orchestration API, detailing how the GUI will leverage the API to create and modify specific entities and attributes, enabling effective interaction with the integrated tools.

The orchestration API will enable the GUI to perform the following key functions through the NGSI-LD context broker:

1. Configuring the Tools

- o **Specifying Parameters:** The GUI will allow users to define specific parameters required for each tool's operation. This includes settings such as model parameters specific to each tool.
- o **Choosing Input Data:** Users will select relevant input data through the GUI, which will reference specific entities within the NGSI-LD context. This ensures that the selected data is compatible with the tool's requirements.

2. Executing a configured Tool

- o Users will initiate the execution of a configured tool through the GUI, which will modify the relevant entity attributes to indicate that the tool is ready for execution. The orchestration API also provides ways to set up some automatic execution method for configured tools.
- o The context broker will handle the execution process based on the defined entity states, attributes and history.

3. Interacting with the results

- o After execution, users will interact with the results generated by the tools through the GUI. The orchestration API will provide conventions for how results are stored and retrieved, allowing users to view, compare, and visualize data easily.
- o The GUI will create entities to represent results and their associated attributes for easy retrieval and analysis.

2.3 Definition of the conventions

In the context of the MARCLAIMED project, the term "tools" will be referred to as "DataService" throughout the remainder of this document. This terminology is adopted to provide a more generalized specification that can be applicable beyond the scope of the project.

The specification defines conventions for the creation of three types of entities:

1. **DataService:** This entity is created once for each available tool within the project, resulting in a total of six DataServices corresponding to the six tools. The DataService entity represents the tool itself and includes essential information such as:
 - o **name:** The name of the DataService.
 - o **description:** A brief overview of the DataService and its functionality.
 - o **endpoint:** The URL where the DataService can be called.
 - o **availableParameters:** A list of parameters that can be configured for the DataService.
 - o **expectedInputs:** The types of input data that the DataService requires for execution.
 - o **producedOutputs:** The outputs (attributes) that the DataService generates upon execution.
2. **DataServiceProcessing:** This entity represents an "instance" of a DataService, along with specific parameters and inputs. It signifies a DataService that is ready to be executed without further user input. The DataServiceProcessing entity includes:
 - o **isInstanceOf:** A link to the corresponding DataService it is based on.
 - o **hasSourceEntities:** Relationships to other entities in the context that contain all the required input data. Notably, it does not directly contain any values; instead, it references the necessary input data entities.
 - o **additionalParameters:** a list of parameters needed for the model, that are not retrievable in other entities

Importantly, a DataServiceProcessing can be executed multiple times. Since it does not contain actual data but rather relationships to other entities, executing the same DataServiceProcessing at different times can yield different results. This variability occurs because the underlying input data entities may change over time, reflecting new conditions or updated information.

The handling of the relationships to the source entities and the actual retrieval of input data is handled by the **Tool Management Adapter**, a module of the IDST, which is

external to the NGSI-LD context broker (IDST data core). Details about this adapter are provided later in the DataServiceProcessing execution part.

3. **DataServiceOutput:** This entity captures the results produced by the execution of a DataServiceProcessing instance. It contains:
 - o The output data generated by the execution of the DataServiceProcessing, represented as attributes (if the service predicts temperature, there will be a “temperature” attribute in the DataServiceOutput entity).
 - o **isResultOf:** A relationship to the associated DataServiceProcessing entity, establishing a clear relationship between the input configuration and the resulting outputs. This relationship will include a “executionTime” subproperty which will save the exact time of execution, allowing for a replay of the service at a previous point in time (which might yield different results if the input data have changed)

Example payloads for those 3 entities are provided in the annex.

Workflow Example

To illustrate the workflow, consider the following scenario:

- A user configures a DataService representing the PREDMAR tool, creating a DataServiceProcessing entity that is tailored to the data from demo site 1 using scenario X. This DataServiceProcessing is now ready for execution.
- The user can similarly create another DataServiceProcessing based on the same DataService (PREDMAR), but this time configured with the data from demo site 2 using scenario Y.
- When either DataServiceProcessing is executed, the input data is retrieved and is sent to the service endpoint, and the DataService processes the request, returning output data.
- The results of this execution are stored in a newly created DataServiceOutput entity, which is linked to the corresponding DataServiceProcessing, thereby maintaining a clear connection between the input parameters and the resulting outputs.

If the user later executes the same DataServiceProcessing instance again, it may produce different results if the underlying input data entities have been updated or changed since the last execution. This flexibility allows for dynamic analysis and decision-making based on the most current data available.

This structured approach to entity creation and management within the MARCLAIMED IDST ensures clarity and efficiency in the execution of DataService, facilitating effective water resource management and decision-making.

The next parts of the document will present the two main workflows surrounding these conventions: the **registration of a DataServiceProcessing** (the configuration of a DataService, leading to the creation of a DataServiceProcessing entity) and **the execution of the DataServiceProcessing** (the actual execution of the tool, leading to the creation of a DataServiceOutput entity).

3.0 DataServiceProcessing registration workflow

The process of registering a **DataServiceProcessing** involves configuring a specific instance of a DataService by selecting appropriate input entities and defining how it will be executed. This process is carried out through the GUI in the following key steps:

1. Selecting a DataService

The user begins by discovering from a list and selecting a **DataService**. The GUI retrieves this list by sending a request to the context broker (**GET /entities?type=DataService**). Once the user receives the list, they choose the desired DataService from the options displayed.

2. Selecting Input Entities

After selecting a DataService, the GUI retrieves the **expectedInput** attribute, which contains a list of required entity types (such as "River", "Sensor", ...) or attributes ("humidity", "temperature", ...). The GUI then sends a request for each entity type and/or (e.g., GET /entities?type=River, GET /entities?attrs=temperature) to retrieve the corresponding entities from the broker. The user then selects the appropriate entities from the returned options.

Note: if needed, step 1 and 2 can be inverted. The user might want to select an Asset, (e.g., a River), then the NGSI-LD context broker can discover all DataService, executable on a River entity, using such a query: GET /entities?DataService&q=expectedInput=="River". In this example, if the selected DataService has other expected inputs, they are retrieved and proposed, same as in the normal workflow.

3. Specifying the Execution Method

Once the required input entities are selected, the user defines how the **DataServiceProcessing** will be executed. The available execution options include:

- o **Triggered by the user:** The execution is manually initiated by the user through the user interface.
- o **CRON-based:** The execution follows a fixed schedule (e.g., daily, weekly).
- o **Triggered by a new measurement:** The execution is automatically triggered when a new measurement for a specific attribute is received (e.g., a new sensor reading or water level update).

4. Finalizing the Registration

The completed **DataServiceProcessing** entity is registered with the context broker, indicating that the configuration is ready for execution. This approach ensures that all necessary inputs are assigned to each DataService and that the execution method matches the user's preferences.

Once a DataServiceProcessing is created, it is not meant to be modified. Many existing DataServiceOutput entities might be linked to any given DataServiceProcessing, modifying the latter would mean that many existing results are linked to a wrong configuration. If a user wants to make another processing with different parameters, a new DataServiceProcessing should be created. Therefore, the GUI should only give the possibilities to create, delete or “duplicate + modify” a DataServiceProcessing.

A sequence diagram of this workflow is provided hereafter:

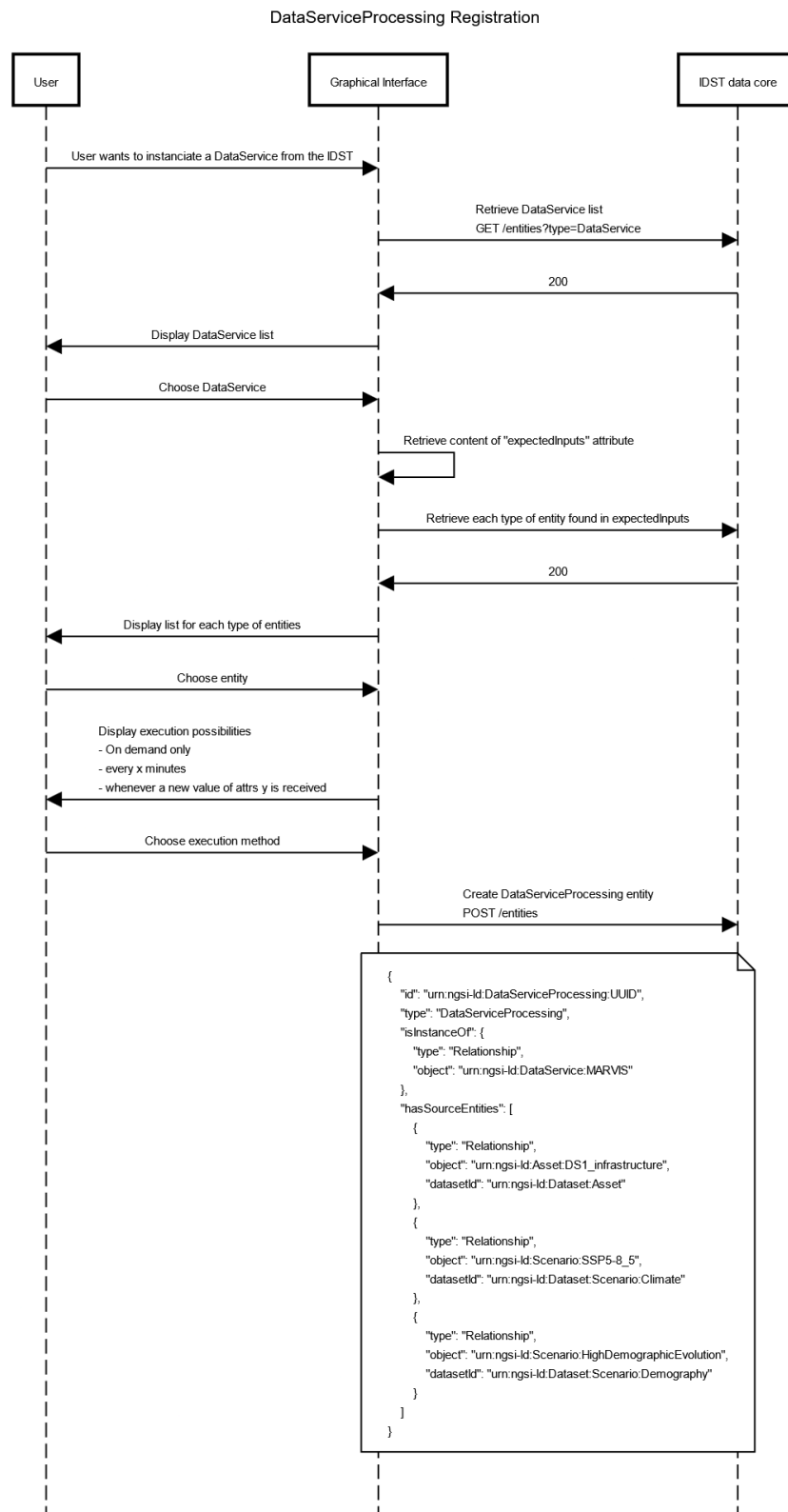


Figure 1: Sequence Diagram for DataServiceProcessing Registration

4.0 DataServiceProcessing execution workflow

The execution of a DataServiceProcessing involves 6 main steps:

1. Triggering of the processing
2. Retrieval of input data in the IDST datacore by the Tool Management Adapter
3. Conversion of the input data from NGSI-LD responses to the format expected by the service
4. Actual execution of the service on input data
5. Conversion of the service output data to NGSI-LD payload
6. Payload is sent to the IDST data core to save the results

4.1 *Triggering of the processing*

Different possibilities for execution method:

- **On demand:** the DataServiceProcessing can be executed by the user at any time from the graphical user interface
- **CRON based:** the DataServiceProcessing is executed based on a CRON job (i.e. on specific times, e.g. every day, every week, ...)
- **On new measurements:** the DataServiceProcessing is executed whenever a new value of a specific attribute is received (e.g. when a new sensor measurement has been received, when new lab analysis has been received, ...)

All 3 possibilities will be handled using **NGSI-LD subscriptions**. A subscription in NGSI-LD is a request made by a client to a context broker to receive notifications about changes to specific entities or their attributes. This allows applications to stay updated with real-time data without constantly querying the context broker. When triggered, subscriptions send a notification to a specific endpoint, which in this case will be the **Tool Management Adapter**.

The 3 execution possibilities will be using the capabilities of NGSI-LD subscription in different ways:

1. **On demand:** The DataServiceProcessing entity contains a "executionRequest" attribute. When a user chooses to execute a DataServiceProcessing from the GUI, this executionRequest is updated with a *true* value and the timestamp of the demand. This will trigger a subscription that will send a notification containing the content of the DataServiceProcessing entity to the Tools Management Adapter.

2. **CRON based:** a NGSI-LD subscription with the “timeInterval” parameter is created, it will regularly send a notification to the Tools Management Adapter, based on a CRON specified in the timeInterval parameter.
3. **On new measurements:** A subscription on the attribute that should be watched is created directly. It will send a notification to the Tool Management. It should be noted that this method would fit for a simple model but can be limited for a complex one with many different inputs, receiving a new measurement of a specific attribute does not ensure that each required input is up to date.

4.2 Retrieval of input data in the IDST data core

In the MARCLAIMED project, the execution of a **DataServiceProcessing** triggers a subscription that sends a notification to the **Tool Management Adapter (TMA)**, which is responsible for handling the input data retrieval and execution of the service. The TMA is a unique process within the IDST, but it operates with different configurations based on the specific requirements of each **DataService**.

Upon receiving the content of a **DataServiceProcessing**, the TMA goes through the entities listed in the **hasSourceEntities** relationship. These entities represent the input data needed for the service. The TMA queries each of these entities to retrieve the actual input data required for the execution of the **DataService**.

The TMA is highly configurable and can handle different types of entity queries depending on the nature of the required input data. Below are the main configurations and methods the TMA uses to query and retrieve data:

1. **Simple Entity Query (GET /entities/{entity_id})**

In cases where an entity is selected as a source entity and the user needs the current state of that entity, the TMA can simply send a **GET /entities/{entity_id}** query. This request retrieves the current values of all attributes of the entity. It's a straightforward query for retrieving the most recent data associated with that entity.

2. **Temporal Data Query (GET /temporal/entities/{entity_id})**

For scenarios where a time series is required—such as historical data for a specific entity—the TMA can use the **GET /temporal/entities/{entity_id}** query. This allows the TMA to retrieve the temporal evolution of the entity's attributes over a defined period. In these cases, the **DataServiceProcessing** configuration must specify the length of the time interval for which the data should be retrieved (e.g., last 24 hours, past week, etc.). The NGSI-LD specification also allows specifying some aggregation methods, allowing some resampling of data. Some transformations

might be too complex to be done directly in the query and should be performed in the conversion done by the TMA (detailed later)

3. Entity with Relationships

In some cases, a selected source entity may not contain direct data but instead includes relationships to other entities. The TMA must then identify and query the related entities separately. For example, a complex parameterization could involve an entity representing a demo site, which links to multiple other entities (e.g., sensors, infrastructure, water sources). In such cases, the user might select only this “demo site” entity for simplicity, and the TMA’s configuration would know about the sub-graph to query and thus how to follow the relationships and query the individual entities. The TMA would retrieve the IDs of these relationships and execute separate queries to gather all the necessary data from the linked entities.

By having different configurations based on the requirements of each DataService, the TMA ensures that it can handle various data retrieval methods and complexities in the system, enabling flexible and dynamic execution of DataServiceProcessing instances.

4.3 *Conversion data and actual execution of the service*

Once the input data is retrieved in NGSI-LD format, it must be converted into the specific format that the service expects. This conversion ensures compatibility between the system’s data model and the input requirements of the DataService, enabling successful execution.

This conversion part will obviously be specific for each DataService. The configuration DataServiceProcessing has been kept relatively simple (only selecting an entity, without concerns of what attributes are inside), to ensure an easy workflow for the user, meaning that it is the Tool Management Adapter that have the information of which attribute to retrieve in each type of entity that is received and what to convert it too.

With the input data properly formatted, the system proceeds to execute the selected DataService. This step involves running the service’s algorithms or processes on the input data, producing results based on the configuration set during the registration of the DataServiceProcessing.

After the service completes its execution, the output data must be converted back into NGSI-LD format. This conversion ensures that the results are compatible with the IDST data core, where they will be stored for future retrieval and analysis.

The final step involves sending the converted output data as an NGSI-LD payload to the IDST data core. The results are then stored and can be accessed by users or other system components for further use, visualization, or decision-making.

A sequence diagram for the whole workflow is shown hereafter (divided in 2 for readability; the full version is provided in annex):

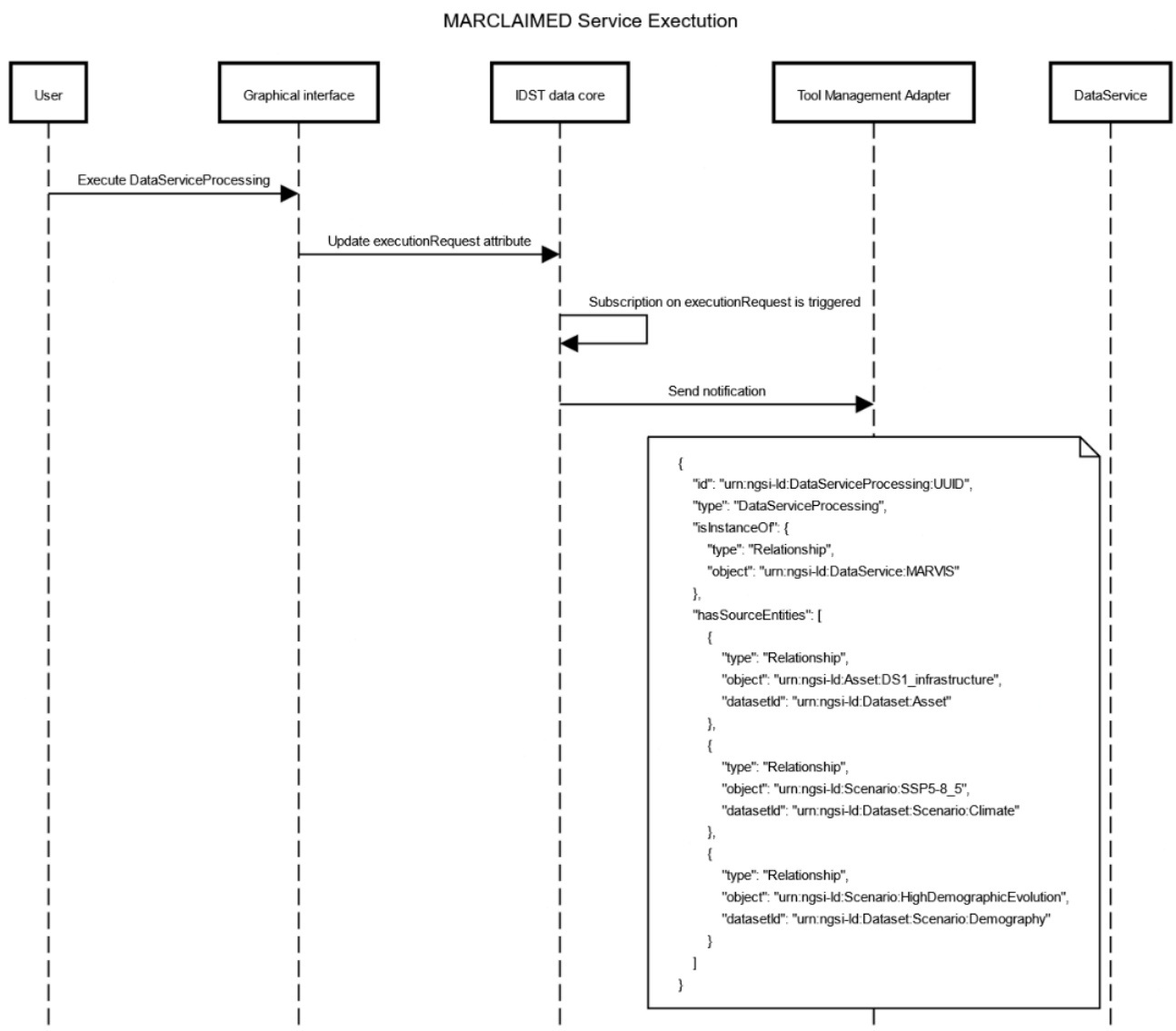
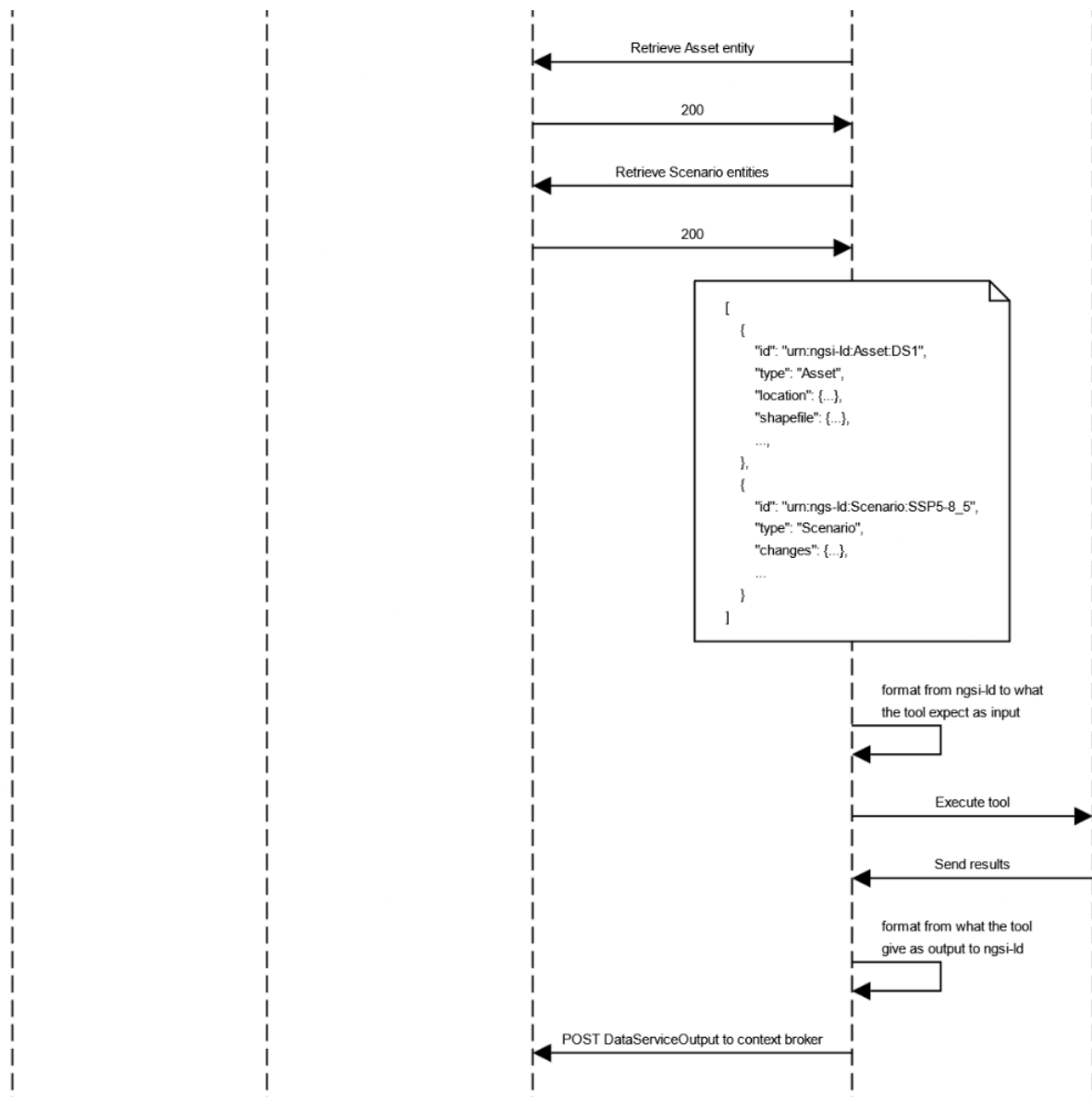


Figure 2: Sequence Diagram for DataServiceProcessing Execution (first half)



5.0 Interaction with the DataServiceOutput

This section presents how the results of the execution of DataServiceProcessing are represented in the context broker and how this representation allows to perform the action that are expected to be done on those results:

- Display results
- Retrieve all results for a specific DataServiceProcessing
- Retrieve all results for a specific DataService
- Compare 2 DataServiceOutput
- Link a result to its source entity
- Link a result to its input data

5.1 *The DataServiceOutput entity*

When it comes to deciding how and where the results of a processing will be stored in the context broker. 2 different possibilities are considered:

- **Storing the results in the source entities directly**, in a different dataset, using the datasetId capabilities of the NGSI-LD specification.
- **Storing the results in a new DataServiceOutput entity**, that is linked to the DataServiceProcessing entity

Those possibilities are meant to be adapted to different types of DataService. For example, in the context of MARCLAIMED, if we compare 2 of the DataService that will be deployed: MARVIS and MARINSURE.

The first one, MARVIS, is a virtual sensor that predicts values of a pollutant that is not measured, based on values of measurement of other parameters that are being measured, meaning we expect this DataService to be executed regularly to update new values. In this case, all the predictions will consist of a single value, or a set of a few values for a single attribute (the predicted parameter). Each execution will be in the continuity of the previous one, so we expect them to be all in the same place, in the source entity, to access the full time series easily. The predicted attributes can make use of NGSI-LD "datasetId", that allows to store different versions of a same attribute, in the same entity, allowing to still separate the predicted time series from the real measured one.

For the second one, MARINSURE, the DataService is more complex. It requires more inputs and produces more outputs. Each execution generates multiple time series of predicted attributes. Due to this complexity, the structure of the results cannot be saved entirely in the source entity without overwriting previous executions.. In this case, we don't expect the DataServiceProcessing to be executed continuously, it makes more sense to save the results separately in another entity: a DataServiceOutput

5.2 *Interaction with results*

The results stored in the DataServiceOutput can be easily retrieved with the capabilities of the NGSI-LD API. DataServiceProcessing entities do not have direct relationships to DataServiceOutput, meaning when doing a simple GET on a DataServiceProcessing will not provide any information on its produced results. The results must be retrieved using the “query” functionality (parameter q=) of the NGSI-LD API, which allows to retrieve entities based on a query such as this one: “retrieve all entities that have a relationship “isResultsOf” linking to this id” (the id of a DataServiceProcessing in our case).

Example for typical retrieval actions, needed to be done by the GUI to display the results to the user are given hereafter:

- **Retrieve all results from any processing**

GET /entities?type=DataServiceOutput

- **Retrieve all results from a specific DataServiceProcessing**

GET /entities?type=DataServiceOutput&q=isResultOf=={DataServiceProcessingId}

- **Retrieve all results for a specific DataService**

Must be done in 2 steps

1. GET /entities?type=DataServiceProcessing&q=isInstanceOf=={DataServiceId}
2. For each DataServiceProcessing entities retrieved:
GET
/entities?type=DataServiceOutput&q=isResultOf=={DataServiceProcessingId}

Two different sets of results produced by either the same DataServiceProcessing or different ones will both be a DataServiceOutput entity of the exact same format, allowing for easy and natural comparison between results.

5.3 *How to link output attributes to their source entities*

The DataServiceOutput entities may contain multiple different attributes that have been calculated/predicted by the DataService, and all those attributes refer to an entity in the data model. For example, if a DataService uses historical water height data from a River entity to predict some future values of water height, in this case the result of the DataService (the predicted future water height values) are saved in the DataServiceOutput, but they need to have a link to the initial River entity, otherwise the results are missing information.

In NGSI-LD, it is possible to create a relationship between a specific attribute of an entity and another entity, which would fit the need expressed above. We could add a relationship between the water height attribute of the DataServiceOutput and the initial River entity, indicating that this result refers to this entity. However, since water height is a temporal attribute, this would imply that the relationship is added to every single instance of the water height attribute, which is not ideal from a performance/size perspective. The proposed solution is to have instead a "sourceEntity" attribute to the DataServiceOutput, that is not temporal, and that provides the source entity for each attribute found in DataServiceOutput.

5.4 How to link a DataServiceOutput to its input data

The DataServiceOutput entity contains a relationship that links it to its DataServiceProcessing, so one can easily retrieve the parameters used to produce this DataServiceOutput. However, the DataServiceProcessing does not directly contain actual input data, it only contains links to the entities that do. If we want to retrieve the actual input data used to produce a DataServiceOutput, we simply resend the content of the DataServiceProcessing to the Tool Management Adapter, and the input data will be retrieved.

However, this solution does not work if the data in the source entities have changed between 2 executions. For example, a DataServiceOutput could use a weather forecast as its input, it produces a DataServiceOutput, but the next week, the forecast might be updated with new value, which will replace the old ones, meaning the actual input values used to produce the DataServiceOutput are no longer available in the context broker.

To tackle this issue, a possibility would be to use a "snapshot" functionality which has been proposed for inclusion into the NGSI-LD specification. A snapshot is fixing the state of (a part of) the context broker, allowing to keep the state of the source entities at the time of the execution, ensuring that the actual input data will never be overwritten and will always be available.

The snapshot functionality is still being discussed and is, as of the time of writing this document, not in the last version of the NGSI-LD specification.

6.0 CONCLUSIONS

The MARCLAIMED project, through its carefully designed workflows and advanced tools, is laying the foundation for a transformative approach to water resource management. The DataServiceProcessing registration and execution processes are integral to ensuring that the various tools within the IDST operate efficiently and adapt to the diverse needs of end-users. From configuring the input data sources to executing the services and retrieving the results, these workflows provide a streamlined and flexible approach to tackle complex water management challenges.

The Tool Management Adapter (TMA) plays a pivotal role in ensuring that data retrieval and service execution are tailored to the specific needs of each DataService. Whether through simple entity queries, retrieving time-series data, or navigating complex relationships between entities, the TMA allows for dynamic and customizable data interactions. This flexibility is essential in adapting to different scenarios, such as varying data availability, user preferences, or the complexity of the water management tasks at hand.

7.0 ANNEX: NGSI-LD payloads

DataService entity

```
{
  "id": "urn:ngsi-ld:DataService:MARVIS",
  "type": "DataService",
  "description": {
    "type": "Property",
    "value": "MARVIS tool, virtual sensor"
  },
  "endpoint": {
    "type": "Property",
    "value": "https://..."
  },
  "parameters": {
    "type": "Property",
    "value": {
      "coef1": 2,
      "coef2": 0.1,
      "mode": "...",
      "...": "..."
    }
  },
  "requiredInputs": {
    "type": "Property",
    "value": ["Asset", "Scenario"]
  },
  "expectedOutputs": {
    "type": "Property",
    "value": ["pollutant1", "..."]
  }
}
```

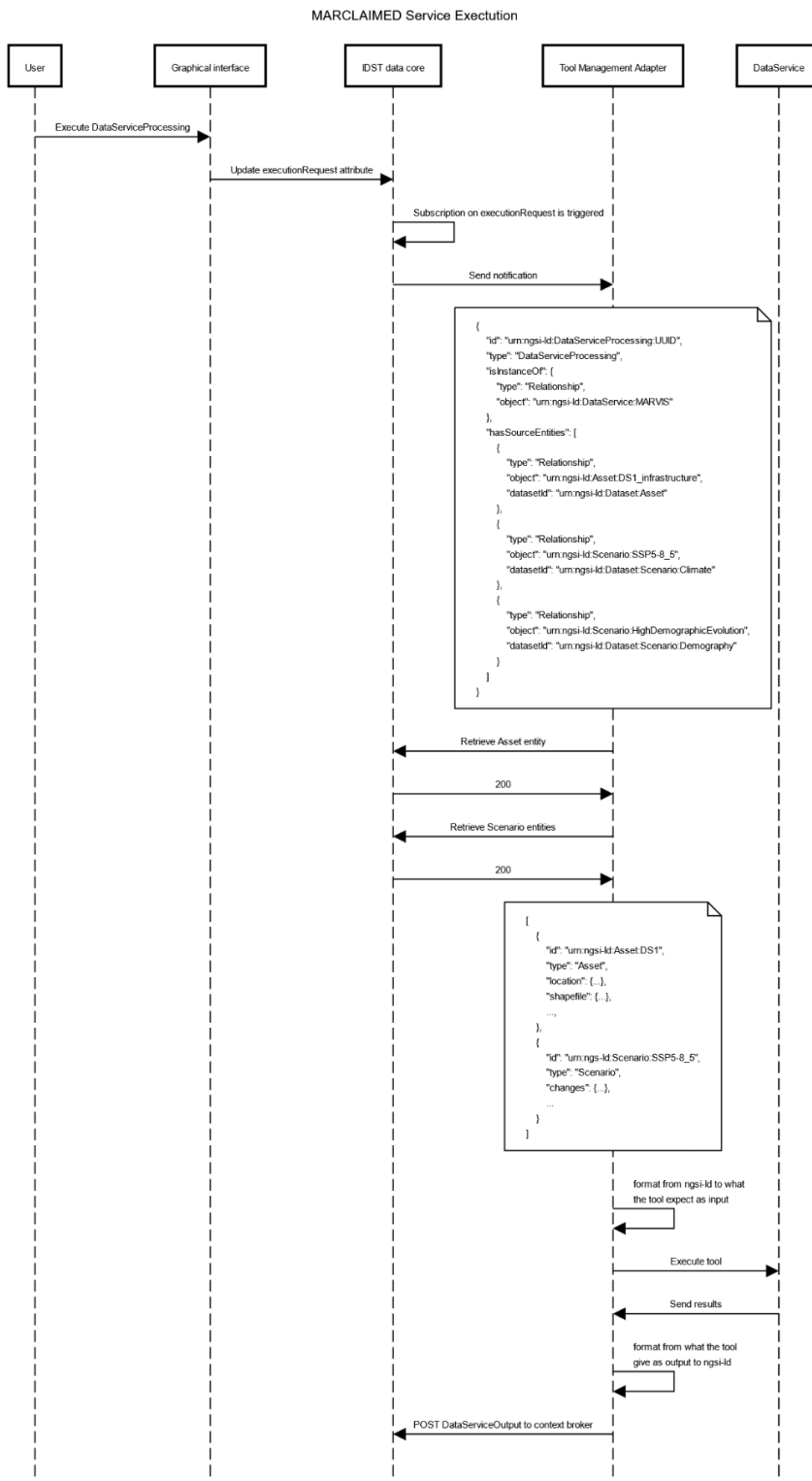
DataServiceProcessing entity

```
{
  "id": "urn:ngsi-ld:DataServiceProcessing:UUID",
  "type": "DataServiceProcessing",
  "isInstanceOf": {
    "type": "Relationship",
    "object": "urn:ngsi-ld:DataService:MARVIS"
  },
  "hasSourceEntities": [
    {
      "type": "Relationship",
      "object": "urn:ngsi-ld:Asset:DS1_infrastructure",
      "datasetId": "urn:ngsi-ld:Dataset:Asset"
    },
    {
      "type": "Relationship",
      "object": "urn:ngsi-ld:Scenario:SSP5-8_5",
      "datasetId": "urn:ngsi-ld:Dataset:Scenario:Climate"
    },
    {
      "type": "Relationship",
      "object": "urn:ngsi-ld:Scenario:HighDemographicEvolution",
      "datasetId": "urn:ngsi-ld:Dataset:Scenario:Demography"
    }
  ],
  "additionalParameters": {
    "type": "Property",
    "value": {
      "coef1": 2,
      "coef2": 0.1,
      "mode": "...",
      "...": "..."
    }
  }
}
```



DataServiceOutput entity

```
{
  "id": "urn:ngsi-ld:DataServiceOutput:UUID",
  "type": "DataServiceOutput",
  "isResultOf": {
    "type": "Relationship",
    "object": "urn:ngsi-ld:DataServiceProcessing:UUID"
  },
  "sourceEntity": {
    "type": "Property",
    "value": {
      "pollutantConcentration": "urn:ngsi-ld:River:1",
      "temperature": "urn:ngsi-ld:WeatherInformation",
      "precipitation": "urn:ngsi-ld:WeatherInformation"
    }
  },
  "pollutantConcentration": {
    "type": "Property",
    "value": 2.56,
    "observedAt": "2025-04-01T00:00:00Z"
  },
  "temperature": {
    "type": "Property",
    "value": 21,
    "unitCode": "CEL",
    "observedAt": "2025-04-01T00:00:00Z"
  },
  "precipitation": {
    "type": "Property",
    "value": 9,
    "unitCode": "MMT",
    "observedAt": "2025-04-01T00:00:00Z"
  }
}
```



MARCLAIMED will support decision makers to integrate Managed Aquifer Recharge (MAR) with Alternative Water Resources (AWR) in River Basin and Drought Management Plans supporting the adaptation and resilience of structural supply systems in the context of climate change. The main outcome will be the MARCLAIMED Integrated Decision Support Tool (IDST) that will provide solutions to tackle existing and emerging threats and support decision making and adaptation policies.



MARCLAIMED



**Funded by
the European Union**