D9.6 – Demonstrator of a simulation run by an experiment, and an experiment run by a simulation.

VERSION

VERSION	DATE
1.0	31 January 2024

PROJECT INFORMATION

GRANT AGREEMENT NUMBER	957189
PROJECT FULL TITLE	Battery Interface Genome - Materials Acceleration Platform
PROJECT ACRONYM	BIG-MAP
START DATE OF THE PROJECT	1/9-2020
DURATION	3 years
CALL IDENTIFIER	H2020-LC-BAT-2020-3
PROJECT WEBSITE	big-map.eu

DELIVERABLE INFORMATION

WP NO.	9
WP LEADER	Marzari (EPFL)
CONTRIBUTING PARTNERS	EPFL, KIT, DTU, SINTEF
NATURE	Demonstrator
AUTHORS	E. Flores, F. Liot, N. Marzari, G. Pizzi, F.F. Ramirez, S.K. Steensen, H. Stein, M. Vogler
CONTRIBUTORS	for a list of all the contributors to the second demonstration of FINALES please see the author list in the BIG-MAP Archive entry: https://archive.big-map.eu/records/j5wjm-q0r39, as well as the co-authors of Vogler et al., Matter 6, 2647 (2023), https://doi.org/10.1016/j.matt.2023.07.016
CONTRACTUAL DEADLINE	31/01-2024
DELIVERY DATE TO EC	24/01-2024
DISSEMINATION LEVEL (PU/CO)	PU

ACKNOWLEDGMENT

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 957189. The project is part of BATTERY 2030+, the large-scale European research initiative for inventing the sustainable batteries of the future.

ABSTRACT

In this deliverable, we discuss one of the main objectives of BIG-MAP: the development of a software infrastructure to deliver an autonomous laboratory, fully integrating simulations and experiments, a crucial ingredient to accelerate the design and discovery of new batteries.

The main goal originally set for this deliverable was the demonstration of a simulation run by an experiment and an experiment run by a simulation. However, during BIG-MAP, we pushed much beyond this simple demonstrator, delivering a fully autonomous platform where multiple independent active components (named tenants), be them simulations or experiments, interact seamlessly toward a common goal driven by the artificial intelligence algorithms developed in WP11.

The platform, FINALES, is designed to be agnostic and scalable beyond the boundaries of a single lab, and we demonstrate its application in two demonstrator runs, where tenants are distributed across Europe.

The first version of the FINALES platform (FINALES 1) was written from scratch to address the need for an agostic central server to coordinate all tenant communication. Its design was already briefly introduced in deliverable D9.2 "*Automated workflows linking experiments and modelling for representative samples of battery materials*"; following its complete implementation, its use was successfully shown in a first demonstrator run, which was published in M. Vogler *et al.*, Brokering between tenants for an international materials acceleration platform, Matter 6, 2647 (2023). The platform, its design and the demonstration are discussed below in this deliverable.

From this first demonstrator run, we learned several important lessons that led us to a complete rewrite of the code, resulting in the current production version (named FINALES 2). Such lessons and the resulting design of FINALES 2 are also described in detail in this deliverable. To demonstrate its application, we perform (and describe here) a second new demonstrator run, connecting together several distributed tenants (tenants are briefly summarised here, and discussed in full detail in deliverable D10.4). In this second run, we demonstrate tenants spanning robotic and automated experiments, simulations of various scales, machine-learning models, and physical sample transportation orchestrators.

With FINALES 2, BIG-MAP delivers a scalable and agnostic autonomous platform that can be applied to a variety of different autonomous laboratories. It can be easily extended to any of the workflow engines of BIG-MAP (that are well integrated, as discussed in deliverable D9.3, with two of them - AiiDA and Pipeline Pilot - already actively used with FINALES) and more generally to the workflows of the BATTERY2030+ community (J. Schaarschmidt *et al.*, Workflow Engineering in Materials Design within the BATTERY 2030+ Project, Adv. Energy Mater. 12, 2102638 (2022)) and, in the future, to any other relevant automated experiment (within BIG-MAP, see WP4 and WP5, and beyond, e.g. the Aurora BIG-MAP Stakeholder Initiative - see BIG-MAP deliverable D9.2).

In addition, in full alignment with the goals of WP9 of providing a secure data-sharing infrastructure that can accelerate data dissemination within the BIG-MAP consortium (and, more broadly, in BATTERY2030+), we successfully integrate FINALES 2 with the BIG-MAP Archive.

Thanks to this integration, data originating from the autonomous optimization and discovery runs of FINALES are uploaded without any human supervision as records in the BIG-MAP Archive, making all data immediately available to the whole consortium even before the autonomous runs are completed, thus providing the possibility for immediate data analysis and accelerated discovery.

Furthermore, in tight integration with the efforts of WP7 on the development of the BattINFO ontology and of a semantic search platform, we are working to extend the raw and parsed data that is submitted to the BIG-MAP Archive with semantic annotations using BattINFO. This will provide full semantic metadata annotation, in JSON-LD format, to any FINALES entry uploaded to the BIG-MAP Archive, without the need of any manual annotation by the researchers. Most importantly, this will make any FINALES dataset automatically findable, indexable, and searchable without human effort, e.g., by scripts and software platforms, making it possible to perform searches that are concept-aware in a global knowledge graph, to explore automatic data connections and suggestions, and to automatically perform data validation.

TABLE OF CONTENTS

1. FINALES – an intention-agnostic broker

Integration of theory and experiment is within the core of BIG-MAP. Hence, novel ways beyond the mere “theory discovers X and experiments verify” or “experiments discover Y and theory confirms” are necessary. Within the partners, there has thus been a larger effort to truly integrate experiment and theory towards “A triggers B”. Even more so, the goal of the "Fast Intention Agnostic Learning Server" (FINALES) is that we can build joint theory and experiment workflows that incorporate data from multiple sources at different degrees of fidelity. This is necessary as there are some physicochemical properties that are easily accessible by experiment whilst there are others that are easily accessible by theory. If one considers a relatively simple workflow in which one wants to assess the influence of the electrolyte formulation (excl. additives) on the conductivity and life cycle of batteries, one could come up with a sequence diagram like the one in Figure 1.

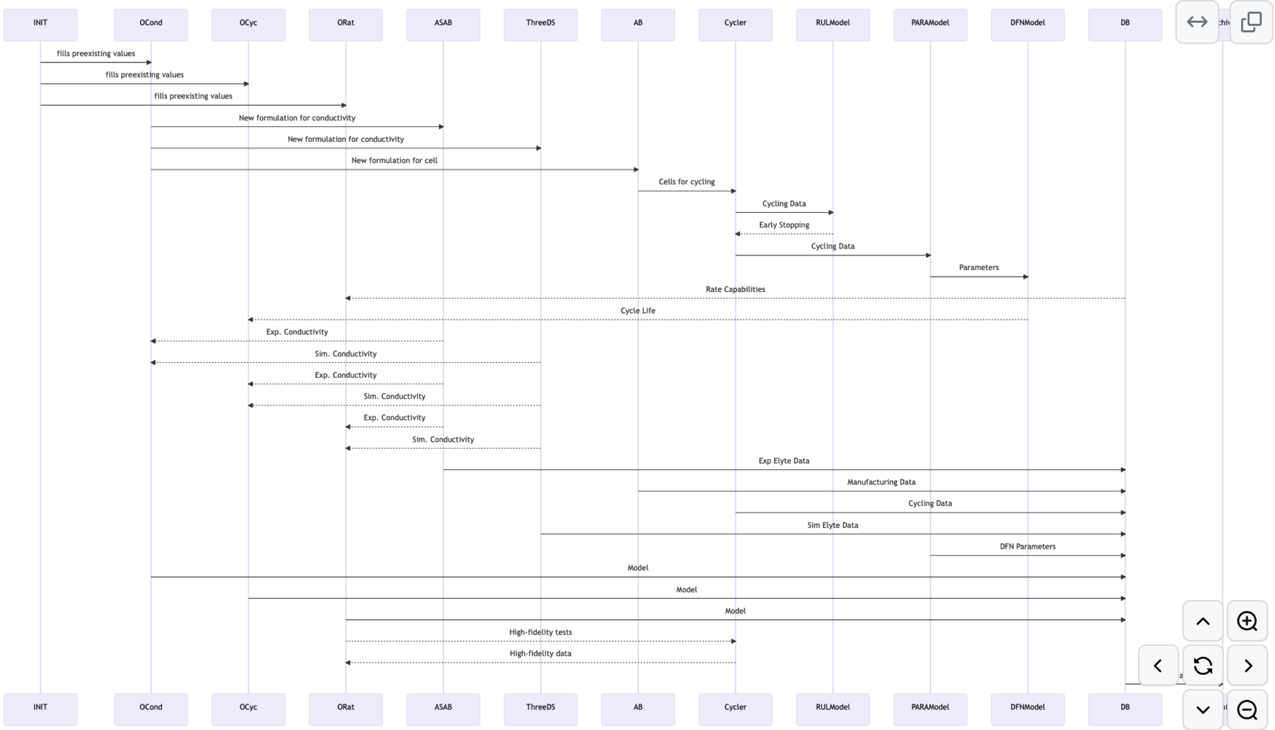


Figure 1. Sequence diagram for a relatively simple workflow to assess the influence of the electrolyte formulation on the conductivity and life cycle of batteries.

This diagram highlights that already a relatively simple workflow requires complex data structures. The mere addition of a single tenant to this workflow after the initial design necessitates significant efforts to integrate that tenant's data structures and interfaces. Hence, we adopted a centralised brokering approach in designing FINALES, where the data structure is set once, and every tenant writes an inheritance definition that is annotated by links to an ontology. This design creates a workflow in which the addition of new tenants is even possible on the fly. A downside is, however, the necessity of an orchestrating tenant that triggers parallel and sequential tenant tasks. Within the first run of FINALES (published in Vogler et al., Matter 6, 2647 (2023), https://doi.org/10.1016/j.matt.2023.07.016) we demonstrated the general feasibility of this approach for the joint optimization of density and viscosity in an aprotic electrolyte. This example was deliberately chosen as new formulations do not (necessarily) result in structural changes in the electrolyte. Building upon this, we then ran a second version (improved backend, better annotation, more general tenant specifications) that also included the manufacturing and testing as well as the analysis of batteries with the combinatorial variations of the electrolyte. This was done to study the influence of said formulation variation on the cycling life and, ultimately, SEI formation. Since this FINALES run incorporates all aspects of electrolyte design in both experiments and theory, manufacturing, testing, data analysis, knowledge extraction, explainability and planned integration with ontology annotation, we believe that this constitutes a true MAP to unravel the solid electrolyte interface genome.

This manifestation of a BIG-MAP goes beyond the originally envisioned direct calling of simulations from experiments and vice versa, as they can be called modularly and on the fly in a true MAP. This "overachieving" beyond our initial goals results from intense discussions that were only possible (and necessary) because there was a critical mass of scientists from different domains working on different levels of fidelity.

2. FINALES 1 – the first implementation

The initial concept of FINALES has been reported in depth in deliverable D9.2 *Automated workflows linking experiments and modelling for representative samples of battery materials*
and has subsequently been publicly communicated in M. Vogler *et al.*, Brokering between tenants for an international materials acceleration platform, Matter 6, 2647 (2023).

3. FINALES 2 – the next generation

The second release of FINALES involved a full rewrite of the codebase, based on the lessons learnt from the first pilot run, while preserving the core ideas of the project. This meant keeping it as a passive message broker whose main task is maintaining an internal queue of requests and replies while preserving all generated data via an append-only mechanism. On the other hand, the primary objective of refactoring was to create a more generalizable foundation that would allow to easily expand the use cases to new tenants and different research projects, as well as to make it ready to be connected with ontologies and semantic annotations.

One of the biggest changes involved abstracting the REST-API interface to a more project-agnostic model. The previous version communicated by exchanging "measurement" objects that directly contained parameters for very case-specific parameters, such as temperature or formulation. For FINALES 2, these specific parameters were removed from the base communication protocols and the concepts of **"Request"** and **"Result"** are used to represent the transfer of data between tenants. The change is summarised in Figure 2. "Request" and "Results" still have intrinsic fields (such as UUID, creation time, current status, etc. which will be addressed further below), but these are all independent of the actual content of the request (what kind of data or scientific process is required).

Additionally, each "Result" contains the UUID of the "Request" it is addressing, to be able to match them up.

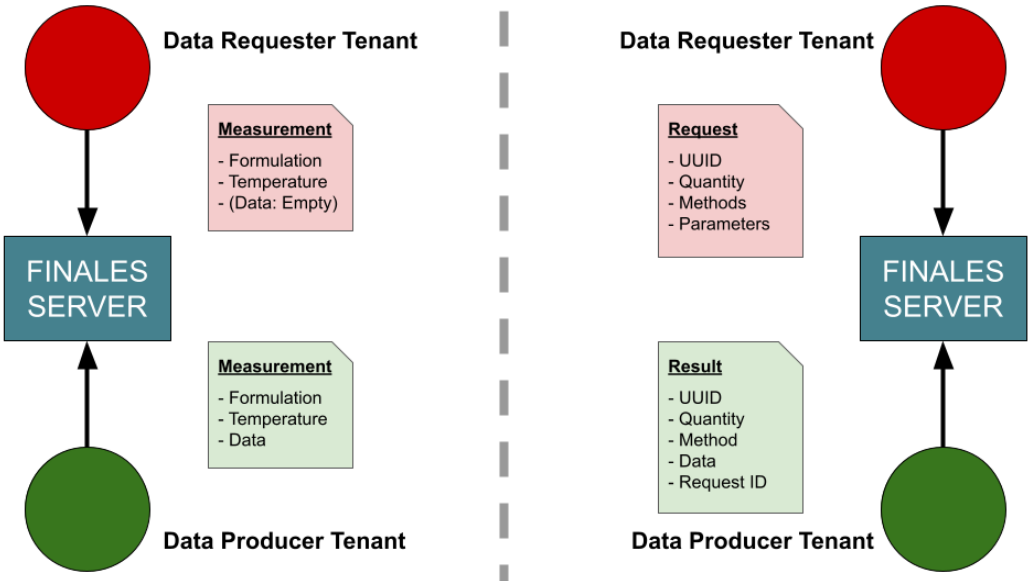


Figure 2. Left: Old design (FINALES 1), with hardcoded properties. Right: new design (FINALES 2), with a generic request and result, only including generic fields, that can be easily generalized to any type of request, measurement or simulation. The figure presents a reduced set of keys for Request, Result and Measurement objects.

The Pydantic schemas that describe the objects associated with scientific parameters and data produced were removed from the interface and thus also removed from being hardcoded into the server codebase. In order to not lose the validation step performed by the server, the concept of **"Capabilities"** was introduced. A "Capability" is identified by a **"Quantity"** and a **"Method"** pair (the first representing the property being referenced, and the second the method for obtaining it), such as "Density - Vibrating Tube Densitometry" or "Conductivity - Molecular Dynamics" (see Figure 3 for a schematic depiction). It then contains two JSON schemas[1] with the information to validate the input parameters in the "Request" and the data posted back as a "Result". Like with the Pydantic schemas, this information consists of a description of all the fields that can be present in the content of the message, as well as the data types expected. The important difference being that these "Capabilities" can now be added, removed or updated in the server at runtime, without needing to shut it down or modify its source code.

[1] JSON Schema is a declarative language to annotate and validate JSON documents (https://json-schema.org/)

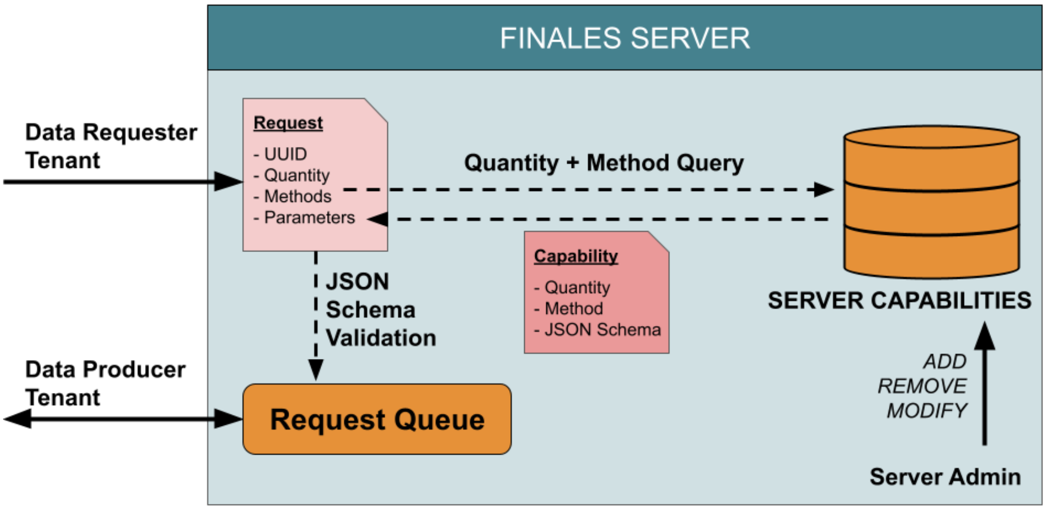


Figure 3. Flow of data requests and responses, using the new functionality of server capabilities.

While this new validation mechanism utilising JSON schemas (instead of Pydantic schemas) makes the server easier to expand to different use cases, the schemas themselves are harder to produce and manipulate due to the format being more difficult to parse by humans. In order to deal with this problem, a separate repository “FINALES2_schemas” was created with the purpose of facilitating the manipulation of the schemas required for our use case. There the schemas are still created using the more intuitive Pydantic classes and are then exported into the JSON format to be used by the server.

On the backend, a SQLite relational database is used to store all information, and the python package SQLAlchemy is employed to interact with it. The database consists of the following tables (shown in more detail in Figure 4, where also the fields of each table are shown, as well as the relationships between them): **Quantities**, **Tenants**, **Requests**, **Results**, **LinkQuantityResult**, **LinkQuantityRequest**, **StatusLogRequest**, and **StatusLogResult**. Initialization of the SQLite database is embedded in the backend. In our implementation, future deployment of another database application, such as MongoDB, is designed to be easy since all backend functionality through SQLAlchemy will remain unaffected. Indeed, designing the backend using SQLAlchemy creates a layer of abstraction where no direct SQL queries are written, which also has the additional advantage of being protected against SQL injection from the exposed API endpoints, through an integrated quoting mechanism in the SQL engine.

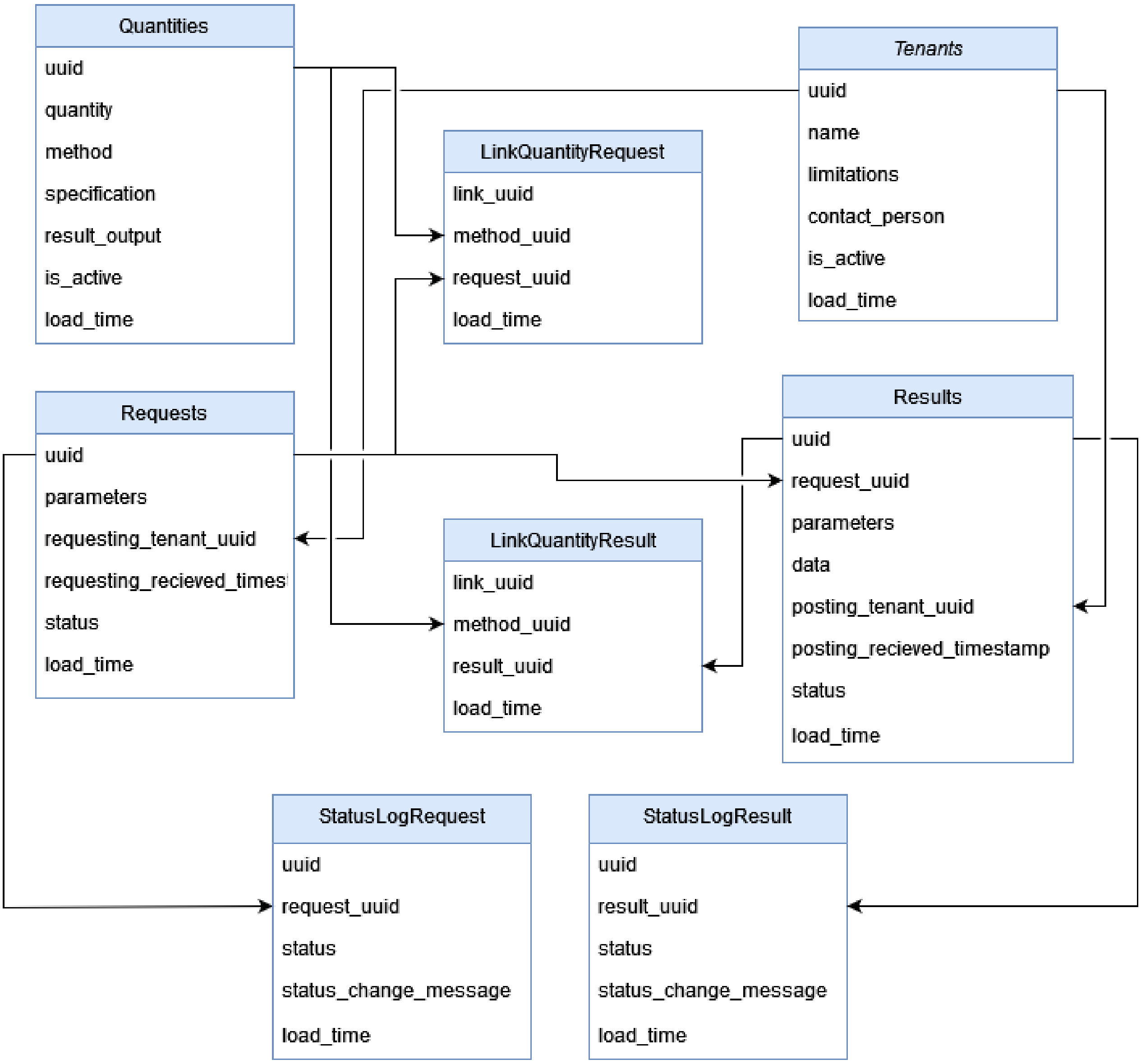


Figure 4. Database schema of the SQLite database of the FINALES 2 server.

As shown in the database schema above, a common feature among all the tables is a "uuid" column as a unique primary key, and a "load_time" column, which is the server timestamp of when the row was appended. The **Quantities** table is where all “**Capabilities**” concepts are defined within an instance of FINALES. When adding a method (a responsibility of the server administrator), one must specify the quantity being measured/simulated, the method used, the specification/limits/validation rules on the needed input parameters to perform the action, and finally how the output will be structured. If the method is defined in the MAP, registration of a tenant is done by specifying what quantity/method the tenant provides and providing information on its specific limitations following the JSON structure in the corresponding **Quantities** table entry. Once the **Quantities** and **Tenants** tables are populated, it is possible to add a request to the **Requests** table. Here the parameters for the request are stated, once again following the structure and fields defined for the quantity. For traceability, the "uuid" of the tenant adding the request is saved in a column as a foreign key.

A posted result will be stored in the **Results** table, where the columns "parameters" and "data" contain the actual experimental/simulation parameters and the actual delivered data, with both the "posting_request_uuid" and "request_uuid" added as foreign keys for provenance tracking.

From the lessons learnt from the first iteration of FINALES, full traceability through timestamps is used, a data quality field is introduced for results, as well as a status field of the request and result entries, which can be changed with the possibility of adding metadata handled by the field “status_change_message” to explain the change. Administrating the status update through an external table to the **Results/Requests** table allows for traceability and conforms to the append-only approach that, combined with CRUD guarantees provided by the database, allows for concurrent-safe storing of multiple status updates at the same time, from different processes.

To accommodate direct programmatic access to all functionalities of the multi-tenant FINALES server to request-posting, result-posting and archiving tenants, we expose 16 endpoints, all of them protected by user authentication. The endpoints all adopt JSON as the format for communication, with the only exception being a specialised endpoint used to dump the entire SQLite database file that can thus be directly downloaded in its native format. This endpoint has been designed specifically for the archiving tenant, discussed in detail below, and is protected through a specialised key. Furthermore, the data can subsequently be processed by the archiving tenant to annotate the data with ontological semantic annotations; the connection with ontologies is discussed in more detail in Sec. 4.3.

The design described above serves the purpose of enabling the user to request results rather than step-by-step procedures. This means that a user may request a value for a quantity without the necessity to take care of the details of the measurement. Specifically, this is relevant if the generation of the requested data requires a workflow to be run. In this case, the workflow may be handled by a data-providing tenant rather than requiring the user to request the intermediate results in the correct sequence. In our demonstration, the Overlort tenant acts as a workflow handler for an experimental workflow. However, one could also think of analogous tenants orchestrating digital workflows.

3.1 Tenants

Detailed descriptions of the tenants which provide an interface to FINALES, as available in December 2023, can be found in deliverable D10.4, “*Tiered theory and experiment screening pipeline as first test case for automatic reasoning calibration”*. Therefore, this chapter will only provide a short summary of the purpose of each tenant.

3.1.1 AiiDA

The AiiDA tenant provides conductivity data based on the model reported by Rahmanian et al.[2]. Since the model was obtained from a dataset based on a different chemical space than the one used in the FINALES context, the output is considered a low-fidelity estimate.

[2] Rahmanian, F.; Vogler, M.; Wölke, C.; Yan, P.; Winter, M.; Cekic-Laskovic, I.; Stein, H. S. 22

3.1.2 Arbin Battery Cycler

The Arbin Battery Cycler tenant offers the capability to cycle coin cells using a standardized cycling protocol. The cycling data and a certain degree of analysis are provided automatically to the MAP.

3.1.3 ASAB

The tenant for the **A**utonomous **S**ynthesis and **A**nalysis of **B**attery electrolytes (ASAB) is used to automatically formulate electrolyte solutions from stock solutions and analyse them regarding their density, viscosity and ionic conductivity. Furthermore, this tenant is also allowed to request only the formulation of an electrolyte solution, so the tenant provides the solution in a vial for further use.

3.1.4 AutoBASS

The **Auto**nomous **B**attery **As**sembly **S**ystem (AutoBASS) tenant assembles CR2032 coin cells fully automatically. It can handle batch sizes of up to 64 cells. This tenant takes all the cell components as inputs and records and reports all the parameters of the assembly as the output.

3.1.5 Degradation Model

This tenant deploys a machine learning model to predict the end of life (EOL) of a battery cell based on its performance in the first few cycles after formation. This predicted EOL value is reported to FINALES as a result.

3.1.6 Transportation

This tenant handles requests for the transport of physical samples. It is implemented as a basic chat application based on sockets to inform humans about a pending transportation request. It also accepts messages about executed transports and handles the posting of the result to FINALES.

3.1.7 Overlort

The **Overl**ooking **Or**chestrating **T**enant (Overlort) handles the workflow associated with assembling and cycling coin cells and the early prediction of their end of life (EOL). To fulfil this task, the Overlort tenant is implemented as a pure software tenant, which breaks down the request for an early prediction of EOL into the individual sub-requests necessary to formulate an electrolyte, assemble the coin cells, cycle the cells, predict the EOL and the involved service requests like, e.g., the transportation or the cells.

3.1.8 Optimiser

The role of the optimiser is to request new measurements from the data-producing methods available in the MAP with the aim of optimizing some predefined objective(s). In general, the optimiser is configured to iteratively consume previously observed measurement results from FINALES, use the data to train a machine learning model, and apply a Bayesian optimisation procedure to propose new measurement parameters which are submitted to FINALES as new measurement requests. The specific optimisation task, including what data to consume and which quantities to optimise, is specified in a configuration file.

3.1.9 Molecular dynamics simulations

This tenant is implemented in BIOVIA Pipeline Pilot. It performs molecular dynamics simulations and provides the resulting values for ionic conductivity to FINALES.

3.2 Demonstrator run

The demonstration of FINALES 2 focused on the generation of data to get insights into battery materials. To achieve this, the demonstration included two independent optimisation tasks. The first task was chosen to be an optimisation of the conductivity of electrolyte formulations composed of ethylene carbonate (EC), ethyl methyl carbonate (EMC), and lithium hexafluorophosphate ($LiPF_6$). The optimiser was set up to vary the molecular fractions of each of these components. The stock solutions selected for the experimental ASAB setup were chosen to be 1.5 M $LiPF_6$ in EC:EMC (1:1 by weight), 1.5 M $LiPF_6$ in EMC, EC:EMC (1:1 by weight) and EMC.

All the stock solutions were commercially obtained from E-Lyte Innovations GmbH, Münster, and used as received. Data generated from molecular dynamics simulations were provided by the Molecular Dynamics tenant implemented and run by 3DS. The optimiser (OCond) considered all the available data from experiments as well as from simulations.

In the second task, more tenants were connected to the MAP, in addition to those used in the context of the first task and mentioned above. The additional tenants include the AutoBASS tenant for assembling coin cells, the Arbin Battery Cycler tenant for cycling the cells, the Degradation Model tenant predicting the end-of-life (EOL) of the coin cells based on their performance in approximately the first 40 cycles. Furthermore, the Overlort tenant was connected to the MAP to keep track of the workflow and trigger the aforementioned tenants once all their inputs are available. In this task, the optimiser (OEOL) was requested to maximise the EOL of the coin cells by varying the formulation of the electrolytes used in the coin cells.

In addition to the tenants mentioned in the above paragraphs, an archiving tenant, developed and operated by EPFL, was connected to the MAP. This tenant created an entry in the BIG-MAP Archive and added new versions to it on a periodic schedule (typically daily over the period when experiments and simulations were running), to back up and make immediately available to the whole BIG-MAP consortium the most recent version of the database, and in particular of the results, requests and capabilities registered with FINALES.

The findings obtained from both tasks are summarised in Table 1. Details regarding the results may be found in deliverable D10.4 “*Tiered theory and experiment screening pipeline as first test case for automatic reasoning calibration*”.

3.2.1 Overview

The key findings from the second demonstration as of December 2023 are summarised in Table 1, which can also be found in deliverable D10.4 “*Tiered theory and experiment screening pipeline as first test case for automatic reasoning calibration*”.

Table 1. Overview of key findings from the second demonstration.

Optimisation task	Components of the electrolytes	Involved Tenants	Results	Future
Maximisation of conductivity	EC, EMC, $LiPF_6$	• Molecular dynamics • ASAB • Optimiser (OCond) • Archiving	• The considered error in the experimental data seems lower than the uncertainty of computational results. • The MAP re-discovered the known trend between the $LiPF_6$ concentration and the conductivity, as reported by Ding *et al.*[7].	• The FINALES concept shall be applied to larger MAPs investigating novel research questions • More complex optimisation tasks can be considered in the future.
Maximisation of EOL		• Molecular dynamics • Overlort • Cycler • ASAB • AutoBASS • Degradation model • Optimiser (OEOL)	• Increasing the molality of $LiPF_6$ results in a higher EOL. • The EOL observed for cells using the electrolyte formulation identified as optimal regarding its conductivity seems to show higher EOL than cells using electrolytes with non-optimal conductivity.	

4. Connections with data repositories

4.1 The BIG-MAP Archive

The BIG-MAP Archive (https://archive.big-map.eu) is a restricted-access digital platform that enables storage and sharing of research data generated by the BIG-MAP project, to provide immediate access to all data to the whole consortium, even before the results are published, in order to accelerate battery discovery.

To achieve the goal of making all results generated in a FINALES run immediately available to any other BIG-MAP researcher, already while the FINALES autonomous optimisation is running, we have developed an additional tenant, the "archiving tenant". It consists of a Python command line client that can periodically be executed to fetch all data from FINALES and upload it directly to the BIG-MAP Archive. More technical details on the archiving tenant can be found in deliverable D10.4 (Sec. 2.9).

Once a new FINALES instance is set up and starts running, the tenant can be started as well, e.g., with a periodic cron-like schedule (e.g., every second day). Such a tenant is configured via a YAML file that includes additional information and metadata that is not automatically retrievable from FINALES itself, such as the title and description of the entry to add in the BIG-MAP Archive, authors list and affiliations, as well as information on the URL needed to reach the relevant FINALES instance.

Every time the tenant is executed, the information from the YAML configuration file is merged with the actual data fetched "live" from the FINALES server (obtained via the relevant REST API endpoints) and then uploaded into the BIG-MAP Archive. Care is taken to create a new BIG-MAP Archive record at the very first execution, while new versions of the same record are instead created at every subsequent execution. In order to reduce storage and back-up time, logic is implemented to upload a file only once, if its content did not change from one version to the next.
Screenshots of some of the resulting BIG-MAP Archive entries that have been automatically created for the demonstrator run discussed in the previous section are reported in deliverable D10.4.
Since data of an entry are made available to the whole BIG-MAP community as soon as they are uploaded in the BIG-MAP Archive (with easy access both to researchers via the web GUI, and code and scripts thanks do the REST API access), the archiving tenancy achieves the goal of making results of the FINALES runs immediately available to the whole consortium, almost "live" during the optimisation process. Moreover, since the full FINALES SQLite database is also dumped in every version, in addition to the relevant data (such as requests and results) in JSON format, the tenant also ensures full data backup, avoiding major data loss in case of issues with the FINALES server.

4.2 Semantic annotations: The BattINFO ontology

The FINALES API guarantees syntactic interoperability across the data and requests it mediates. By enforcing a common exchange protocol and syntax, FINALES enables exchanging information among a large variety of tenants, while remaining agnostic about the meaning of what is being exchanged. Such design choice warrants great flexibility, but potentially at the expense of reduced interoperability of data for other applications. In other words, API-compliant data remains interoperable for the purposes of requests and exchange within the FINALES infrastructure, but it might become more difficult to access for, e.g., data analysis, aggregation, and reuse if the exact data schema defined and used are not properly documented or accessible.

Making data interoperable to other infrastructures and applications requires *unambiguous understanding of its meaning*; e.g., is the "conductivity" key of the FINALES API ionic, electronic, thermal? Ontologies provide the means to achieve such semantic interoperability. An ontology is a collection of concepts, categories and relationships that describe a domain. Within the BIG-MAP project we have developed a Battery INterFace Ontology (BattINFO)[3], with thousands of concepts and relations describing the domain of electrochemistry and batteries. Each concept exhibits an elucidation that defines it, and a unique identifier that makes it findable and unique. Crucially, by linking keys and values from research (meta)data to concepts in the ontology, we explicitly assign meaning to the (meta)data. If the API key "conductivity" is now mapped to the BattINFO concept "IonicConductivity" it becomes clear what the API key means to both i) humans, by consulting the definition of the concept in the ontology and ii) software by examining the unique identifier and its location as a node in the machine-readable ontology graph. In summary, the semantic description of a resource (data, metadata, document, etc.) involves linking the parts of the resource to a controlled vocabulary, such as the one provided by BattINFO. In this way, humans and computers can understand the meaning attributed to the resource and its parts unambiguously.

[3] https://www.big-map.eu/dissemination/battinfo

To provide semantic meaning to the FINALES concepts, we are thus developing scripts to translate the data output from FINALES into a linked-data format, which supports mapping data to controlled vocabularies. We adopt the JSON-LD format as it is open-source, human-readable (semi-structured ASCII text), aligned with the established usage of JSON for data exchange in FINALES, and it explicitly focuses on data linking. Our approach involves utilising EU-wide vocabularies to articulate semantic representations of physical, chemical, and battery-related constructs:

- The Elementary Multiperspective Materials Ontology (EMMO)[4]
- The Battery Interface Ontology (BattINFO)[5]
- The Ontology for Simulation, Modelling and Optimization (OSMO)[6]
- Characterisation Methodology Domain Ontology (CHAMEO)[7]

In addition, we use widely established vocabularies for:

- knowledge representation: OWL[8], RDF[9]
- data types XSD[10]
- description of resources (SCHEMA.ORG)[11]

Opting for EU-wide EMMO-derived ontologies ensures interoperability among concepts, and tight integration within the EU (meta)data landscape and beyond. Additionally, employing OWL, RDF, XSD, and schema.org for resource descriptions, widely prevalent online, enhances web-based resource accessibility. Consequently, the JSON-LD formatted (meta)data produced based on the data generated in a FINALES run attains both analytical interoperability and online discoverability once it becomes public on the web (e.g. by being manually or automatically published on the BIG-MAP Archive, see previous section, and to public data repositories, such as the Materials Cloud Archive[12], a recommended repository for Materials Science data by the EU Commission's journal Open Research Europe; more on the integration is discussed in the next paragraph).

[4] https://github.com/emmo-repo/EMMO

[5] https://github.com/BIG-MAP/BattINFO

[6] https://purl.org/vimmp/osmo

[7] https://github.com/emmo-repo/domain-characterisation-methodology

[8] https://www.w3.org/OWL/

[9] https://www.w3.org/TR/rdf12-schema/

[10] https://www.w3.org/TR/xmlschema11-2/

[11] https://schema.org/

[12] https://archive.materialscloud.org

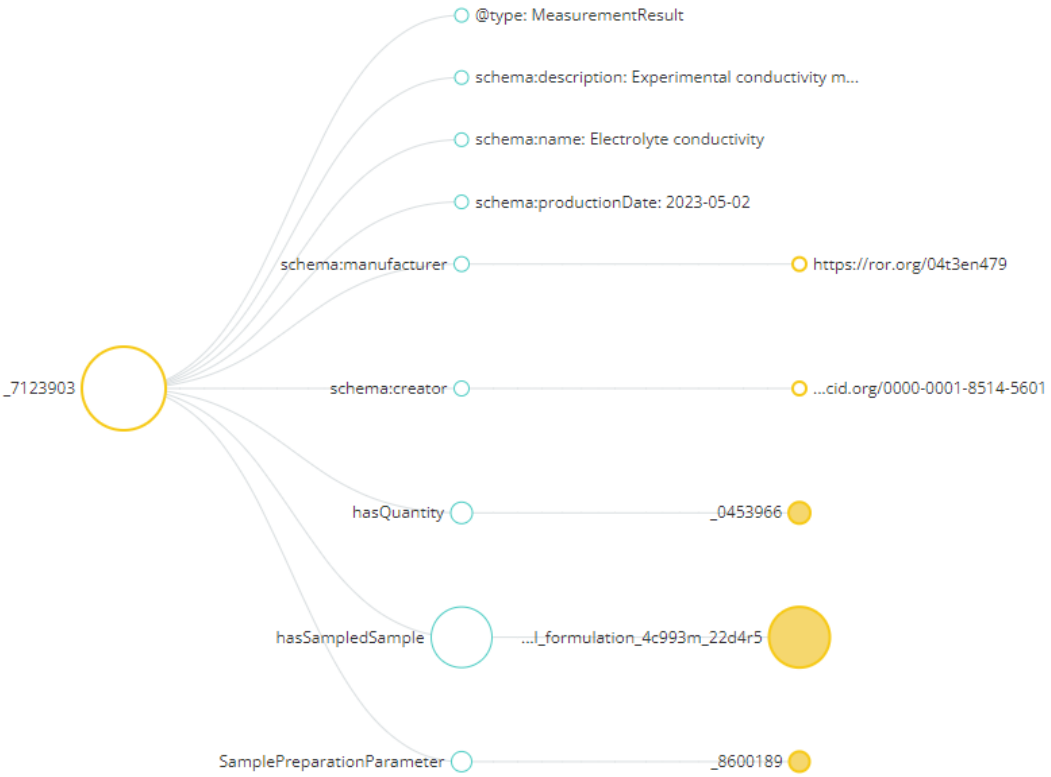
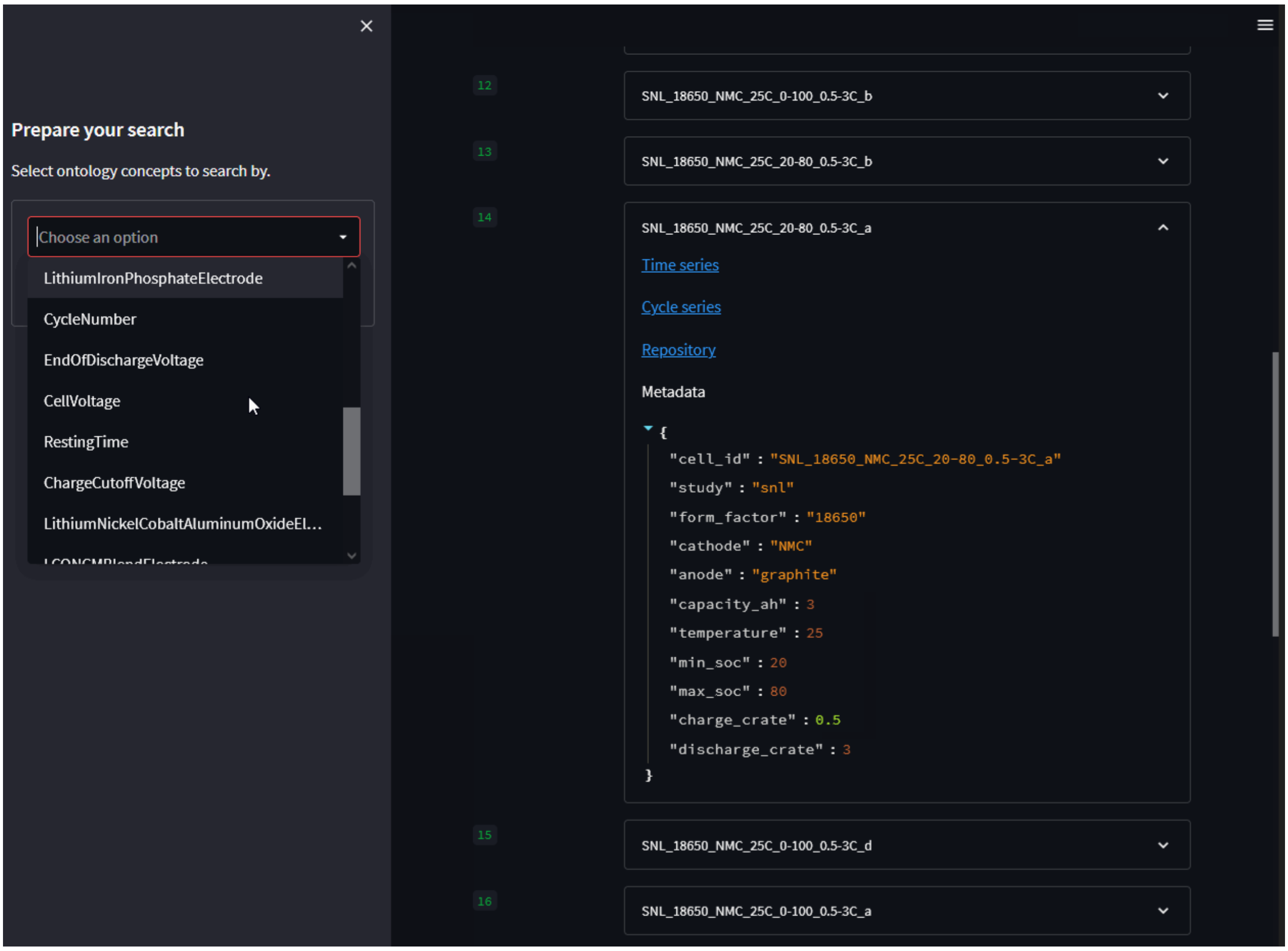

@type: MeasurementResult
schema:description: Experimental conductivity m...
schema:name: Electrolyte conductivity
schema:productionDate: 2023-05-02
schema:manufacturer
https://ror.org/04t3en479
_7123903
schema:creator
...cid.org/0000-0001-8514-5601
hasQuantity
_0453966
hasSampledSample
...l_formulation_4c993m_22d4r5
SamplePreparationParameter
_8600189

Figure 5. Graph visualisation of the JSON-LD description of a conductivity measurement. The description exhibits a hierarchical structure of objects, each with a type mapped to concepts in either of the vocabularies.

4.3 Integration of automated data collection and sharing with semantic metadata annotations

In our current implementation, data generated from FINALES outputs is stored in the BIG-MAP Archive alongside a JSON-LD file; that is, each BIG-MAP Archive entry will also have a JSON-LD file. Figure 5 outlines a graphical representation of an example conductivity measurement. As described in the previous section, the file exposes the mapping between data keys and the corresponding ontology concepts (i.e., those shown in Figure 5). This makes the entry ready for machine-actionable processing, parsing, ingestion and collection in a semantic search engine. Indeed, as a future next step, software can be implemented to crawl through BIG-MAP Archive entries, read each JSON-LD file, and mapping data to a common, project-wide graph database of research resources and concepts, i.e., a knowledge base.



Prepare your search

Select ontology concepts to search by.

12 SNL_18650_NMC_25C_0-100_0.5-3C_b

13 SNL_18650_NMC_25C_20-80_0.5-3C_b

14 SNL_18650_NMC_25C_20-80_0.5-3C_a

Time series

Cycle series

Repository

Metadata

```
{
  "cell_id" : "SNL_18650_NMC_25C_20-80_0.5-3C_a"
  "study" : "snl"
  "form_factor" : "18650"
  "cathode" : "NMC"
  "anode" : "graphite"
  "capacity_ah" : 3
  "temperature" : 25
  "min_soc" : 20
  "max_soc" : 80
  "charge_crate" : 0.5
  "discharge_crate" : 3
}
```

15 SNL_18650_NMC_25C_0-100_0.5-3C_d

16 SNL_18650_NMC_25C_0-100_0.5-3C_a

Figure 6. Demonstration of a small-scale application enabling semantic search of publicly available battery datasets. More information can be found on the corresponding GitHub page[13].

Once built, the knowledge base will unlock semantic search for users, using, e.g. the Semantic Search platform developed in WP7 and shown in Figure 6. Such a platform has multiple advantages over a traditional infrastructure. For one, search is facilitated as the JSON-LD mappings ensure data is described via a set of commonly agreed indexes: the concepts defined in the used ontologies and vocabularies. Moreover, users can find heterogeneous resources in a single place - datasets, articles, researchers, institutions, work packages, etc - since the graph database scales naturally to any resource irrespective of its data structure. In addition, users can explore “data recommendations” since node distances within the knowledge graph are a natural measure of similarity. Last, the logic-based topology of ontologies enables enforcing semantic validation procedures, i.e., identifies mislabeled datasets, incompatible entries, and/or corrupted descriptions.

5. Conclusions and outlook

[13] https://github.com/BIG-MAP/Demo-BatteryDataSemanticSearch

In this deliverable, we have presented the BIG-MAP platform to enable autonomous battery discovery and optimization.

The platform is centred around the FINALES software, designed from scratch within BIG-MAP as an agnostic platform and demonstrated successfully by connecting about 10 different “tenants”, i.e., active components (simulations or experiments) distributed across Europe.

The platform has then been fully redesigned, resulting in the current production-ready version (FINALES 2) by taking into account all lessons learned in this first demonstrator run.

All tenants have been updated, more have been added, and a second demonstrator run (fully driven by the machine-learning algorithms of WP11) has successfully proved the functionality and generality of FINALES 2’s new design.

Thanks to the redesign, not only are automation and integration of experiments and simulations addressed and successfully achieved but also secure data sharing; in addition, we demonstrate our work towards automated semantic metadata annotation. This is obtained thanks to tight integration with the other BIG-MAP platforms and technology, with the goal of delivering a fully integrated BIG-MAP software infrastructure.

In particular, data from FINALES 2 can be automatically shared into the BIG-MAP Archive without any human intervention and during the autonomous runs. In combination with machine-actionable semantic annotations using the BattINFO ontology and other relevant ontologies, this results in all data produced by FINALES will be immediately available to any member of the BIG-MAP consortium and become seamlessly searchable in a global BIG-MAP knowledge graph, due to the semantic search capabilities provided by WP7.

Thanks to the set of capabilities described in this report, and as it is proven by the demonstrator runs discussed in this deliverable, FINALES 2 is now ready to be applied to many more autonomous optimization loops (from other BIG-MAP WPs, from the BIG-MAP Stakeholder Initiatives, and more generally in BATTERY2030+ and beyond), ultimately contributing in a crucial way to the acceleration of novel battery discovery.