Rio Oil & Gas Expo and Conference 2020

ISSN 2525-7579

Conference Proceedings homepage: https://biblioteca.ibp.org.br/riooilegas/en/

Technical Paper

Integrating a geomechanical collaborative research portal with a data & knowledge retrieval platform

Yenier Torres Izquierdo[1]
Grettel Montegudo García[2]
Bruno Alves Novelli[3]
Melissa Lemos Cavaliere[4]
Maria Julia Dias de Lima[5]
Marco Antonio Casanova[6]
Deane Roehl[7].

1. PUC-RIO, INSTITUTO TECGRAF, . RIO DE JANEIRO - RJ - BRASIL, yenier@tecgraf.puc-rio.br
2. PUC-RIO, INSTITUTO TECGRAF, . RIO DE JANEIRO - RJ - BRASIL, ggarcia@tecgraf.puc-rio.br
3. PUC-RIO, INSTITUTO TECGRAF, . RIO DE JANEIRO - RJ - BRASIL, bnovelli@tecgraf.puc-rio.br
4. PUC-RIO, INSTITUTO TECGRAF, . RIO DE JANEIRO - RJ - BRASIL, melissa@tecgraf.puc-rio.br
5. PUC-RIO, INSTITUTO TECGRAF, . RIO DE JANEIRO - RJ - BRASIL, mjulia@tecgraf.puc-rio.br
6. PUC-RIO, INSTITUTO TECGRAF, . RIO DE JANEIRO - RJ - BRASIL, casanova@tecgraf.puc-rio.br
7. PUC-RIO, INSTITUTO TECGRAF, . RIO DE JANEIRO - RJ - BRASIL, deane@tecgraf.puc-rio.br

Abstract

ERAS is a collaborative environment where researchers access, execute and monitor analytical and numerical simulations of geomechanical problems. Usually, Petroleum engineers would spend more than half of their time in organizing, integrating, assembling, and searching relevant data that help them optimize operations in this environment. This work presents how this bottleneck was leveraged integrating ERAS with Danke, a Data and Knowledge Retrieval Platform developed in TecGraf/PUC-Rio Institute.

Keywords: Data integration. Big Data. Knowledge Retrieval. Scalability. Keyword Search

Received: March 11, 2020 | **Accepted:** Jun 06, 2020 | **Available online:** Dec 01, 2020
Article Code: 421
Cite as: Rio Oil & Gas Expo and Conference, Rio de Janeiro, RJ, Brazil, 2020 (20)
DOI: https://doi.org/10.48072/2525-7579.rog.2020.421

1. Introduction

Geomechanics is the theoretical and applied science that investigates the mechanical behavior of geological material. Its role is crucial to the Oil & Gas industry, from exploration to abandonment, and from the microscale to the modeling of reservoirs, fields, and basins. Research in this area provides a fundamental comprehension of subsoil geomechanical phenomena, which can optimize the oil and gas industry's expenditure while reducing geomechanical risks associated with drilling, completion, and development plan.

The ERAS Portal [Lima et al. 2018] is a collaborative environment where researchers access software artifacts, such as models, documents, and computational resources, to retrieve, execute and monitor their analytical and numerical simulations of geomechanical problems. This is a typical Big Data scenario, where the users manipulate a massive volume of data, in different formats. Usually, Petroleum engineers would spend more than half of their time organizing, integrating, assembling, and searching relevant results that help them optimize operations. Nevertheless, this bottleneck could be leveraged by integrating the ERAS Portal with Danke, a platform for data and knowledge retrieval, developed at the Tecgraf/PUC-Rio Institute.

Currently, ERAS stores structured data about projects, simulations, algorithms, jobs in a relational database and offers two mechanisms to retrieve data. The first mechanism is a traditional filter form that users must fill with their search parameters specifying exact data, such as a "simulation solver name" or a "project research area". The second mechanism permits users to write queries based on FIQL[1] (Feed Item Query Language), a syntax for expressing filters in a compact and HTTP URI-friendly. Both mechanisms offer limited search capabilities, since in both cases users need to be aware of the data structure and, in the second case, they must also know the syntax of FIQL queries. On the other hand, Danke offers a keyword search engine, which is an attractive alternative, when compared with these traditional database interfaces, since users do not need to fill numerous "boxes" (in the first approach) and do not need to know the way data are organized and the syntax of the query language (in the second approach). Users specify a few terms, called keywords, and it is up to the system to access the database and retrieve the data that best matches the list of keywords [Izquierdo et al. 2018].

This paper first introduces the approach adopted to integrate ERAS and Danke, considering the ERAS scalability and data decentralization requirements. It describes the implementation issues involved with data ingestion, data preparation and data retrieval, and details how the communication between ERAS and Danke is performed. The main contribution of this work is to equip ERAS with a data retrieval mechanism that does not require users to have specific technical skills for searching, retrieving and exploring data. The paper ends with an experiment that shows how ERAS users can take advantage of Danke services.

2. Integrating ERAS and Danke

2.1. Architecture

To process Big Data and handle a large number of user requests, the ERAS architecture is based on the microservice pattern. A microservice is a decoupled and autonomic software, having a specific functionality in a bounded context [Cerny et al. 2017]. In the ERAS architecture, each microservice can scale independently. From the data point of view, microservices should be equipped with

[1] https://tools.ietf.org/html/draft-nottingham-atompub-fiql-00

dedicated data collections [Dragoni et al. 2017]. Hence, a data retrieval mechanism for ERAS must deal with multiple and independent data collections. ERAS manages structured and unstructured data. Structured data are stored in PostgreSQL databases, which also contain metadata about projects, simulators, simulation execution jobs, parameters, etc. Unstructured data refers to input and output files used as parameters in simulation jobs.

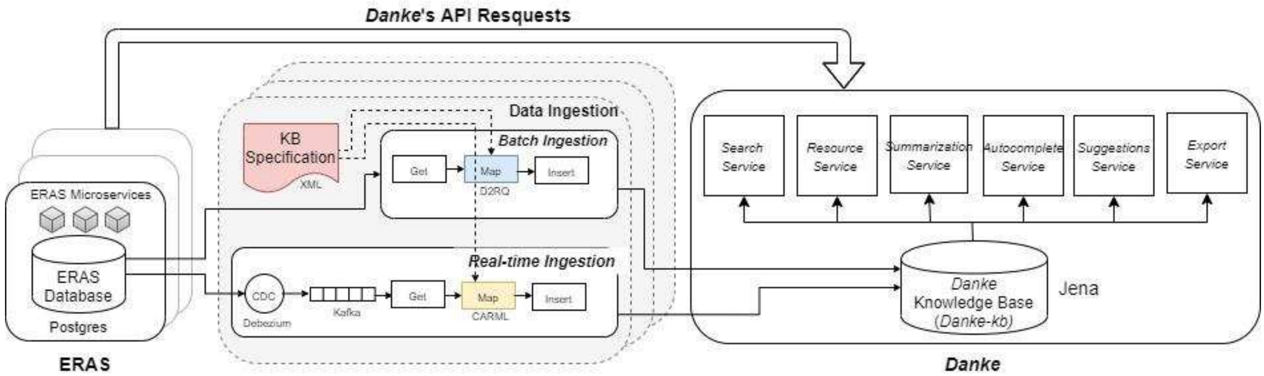
Danke processes large volumes of data (in different formats) from various sources, which facilities integration with ERAS. Danke works with a centralized knowledge base (Danke-kb), created by extracting and transforming source data collections, which can be structured, semi-structured, and unstructured. Danke-kb can be deployed either as an RDF dataset with a schema or as a relational database.

As ERAS can handle multiple and independent data collections, Danke-kb is based on an integrated schema (created from various ERAS data collections) using an RDF graph. RDF is an attractive choice since it can be used for integrated schema descriptions as well as providing a unified view of data. Thus, the ERAS data collections are triplified following specific mappings for each collection, defined as KBSpecification, and the triples are stored into a centralized triple store. In this case, the triple store selected for Danke-kb was Jena TDB (https://jena.apache.org).

The KBSpecification describes the mapping from an ERAS original collection to RDF data. The KBSpecification contains the definition of all entities and properties of the Danke-kb RDF schema. It also specifies and indexes, additional metadata information, and data enrichments.

Figure 1 depicts an overview of the architecture integrating ERAS and Danke. Section 2.2 describes the approach to build Danke-kb from ERAS databases. Section 2.3 explains the communication between ERAS and Danke via service requests to the Danke API. Finally, Section 2.4 shows an illustrative example of data retrieval in ERAS after the integration with Danke.

Figure 1 – Overview of the architecture to integrate ERAS and Danke



Source: produced by authors.

2.2. Danke-kb ingestion

Figure 1 presents how ERAS and Danke are integrated from the data point of view. Conceptually, the ingestion processes extract data from the ERAS data collections and load them into the Danke-kb following the specification mappings declared in the KBSpecification.

For each data collection, the KBSpecification describes the mappings from ERAS original collection to RDF data. Danke can automatically generate a default KBSpecification with a direct mapping of the ERAS relational database to RDF. However, some customizations on the mappings are usually made to successfully perform keyword search based on the user intention queries and the expected answers. In fact, it is common to search keywords that are not named exactly in the same way as in the ERAS original data collections. For example, Boolean attributes can generally be transformed into keyword-searchable values, such as the attribute exit_status of entity job in the ERAS database is a Boolean attribute that registers the execution exit status. Therefore, in the KBSpecification, the attribute values are mapped to terms familiar to the users: 0 (false) is mapped to "Failure", and 1 (true) is mapped to "Success". Hence, Danke provides a data integration strategy that allows data from a variety of sources and formats to be seamlessly integrated, enriched, searched, analyzed, and re-purposed.

Two strategies were adopted to ingest data from ERAS into the Danke-kb: batch and real-time ingestion. Batch ingestion runs in batch mode and loads data that is already stored in the original data collections into the Danke-kb. This process usually involves database queries and includes some transformations (ELT processes). By contrast, the real-time ingestion monitors the original data collections and reacts every time the data changes, updating Danke-kb.

In both batch and real-time ingestion, Machine Learning and Deep Learning algorithms can also be applied over the processing data to, for example, prepare them for analysis, detect anomalies, and find intrinsic links with other data.

The batch ingestion uses the D2RQ platform (http://d2rq.org) to query and extract data from non-RDF databases as virtual, read-only RDF graphs. The process first converts the KBSpecification file to a Turtle file with the D2RQ mapping language declarations. Then, it calls D2RQ functions passing as input the Turtle file with the specific mappings required to triplify the ERAS relational data. Finally, it saves the triples into the Jena RDF store.

The real-time ingestion continuously monitors the ERAS database using a Change Data Capture (CDC) platform that streams every row-level change in the same order they were committed to the original database. Change Data Capture is a technology that captures inserts, updates, and deletes into changed logs. Debezium (https://debezium.io) was used as the CDC platform in the ERAS and Danke integration. Therefore, the program uses CARML, a Java Maven library that transforms the updated row data to RDF triples following an RML mapping [Dimou et al. 2014], a superset of R2RML – a W3C-recommended mapping language that maps relational data to RDF. Thus, this program first translates KBSpecification mappings to RML mappings. So, when there is a data change in the ERAS database, Debezium sends the information streams with these changes to a Kafka broker (https://kafka.apache.org) as a new topic. Debezium was configured with Confluent's Avro Converter to minimize the amount of information generated. The process continues to deserialize the data captured by the broker, creating a structured data file with these changes. Then, the CARML functions to triplify structured data are called by passing as input the structured data file previously generated and the RML mappings. Finally, the program saves the triples into the Danke-kb. A methodology described in [Schmutz 2018] is similar to the adopted.

As pointed out, D2RQ and CARML are technologies to map structured data to RDF data. However, D2RQ only allows relational data as input; by contrast, CARML manages RML files, which support a broader range of data format input. D2RQ is more appropriate to triplify large relational datasets, as ERAS data collections. Hence, it was adopted in the initial ingestion process. By contrast, CARML has poor performance for large data sources, but it is recommended for incremental ingestion processes. Moreover, adopting CARML for incremental ingestion avoids creating an auxiliary in-core database, which would not be the case if D2RQ were used.

2.3. Danke services

The communication between ERAS and Danke is via service requests to the Danke REST API (Application Programming Interface). Table 1 lists the definitions of the available Danke API Services and their methods.

Table 1 – Danke API Services: Methods Definition

Line	Service	Method
1	Search	GET /search/queries
2		POST search/results
3		POST search/results/metadata
4		POST search/results/export
5		GET /search/suggestions
6		GET /search/autocomplete
7		POST /search/results/summarization
8	Resource	GET /resource/{id}
9	Resource	GET /entities
10	Entities	GET /entities/{id}

Source: produced by authors.

The core of Danke data retrieval is a keyword search engine, which is based on an improved version of the strategy described in [Izquierdo et al. 2018]. Given a user-specified list of keywords (k_1, k_2, ..., k_n), it returns an ordered result list as a table, with the most relevant results at the top. The answer may be the result of joining several database resources, that is, an answer to a keyword query need not be constructed out of a single resource. The service first discovers the user's intention (i.e., what the user wants to query and to receive as an answer) and builds a conceptual query to represent it. It then synthesizes a concrete query (in SPARQL) and finally, it executes the synthesized query to get the results. The query compilation process is fully automatic, avoiding user intervention.

After the integration with Danke, ERAS offers a user interface with a text input where users can type the desired keywords to query the database. ERAS sends a request to Danke using the method /search/queries (Table 1, line 1), giving the keywords typed by the users as input, and receiving a conceptual query as output. Then, ERAS calls /search/results (Table 1, line 2), passing the conceptual query as a parameter and obtaining the query results.

After accessing the results, Danke allows users to access a specific element and its linked data, stored in Danke-kb. For this purpose, ERAS calls Danke using the method /resource/{id} (Table 1, line 8), responsible for this task. It has a parameter id that identifies the desired element.

Since a query answer may be a long result set, with many lines, Danke has a built-in data summarization algorithm that produces an overview of the result to the user. The method /search/results/summarization (Table 1, line 7) computes a summary, indicating quantities and percentages, based on the values of the properties in the results. ERAS can present the summarization results in different ways, such as graphics.

Danke helps users formulate keyword queries through an autocomplete service and a suggestion service. The autocomplete service recommends keywords based on text already typed by the users and the metadata information stored in Danke-kb. Table 1 shows, in line 6, the method /search/autocomplete, which receives a text as input and returns an ordered list of keywords to autocomplete the inputted text. The suggestion service recommends an ordered list of keywords based on their relevance that can be used to create, for example, a word cloud, where the size of each word indicates the keyword frequency or importance. Table 1 shows, in line 5, the method /search/suggestions, responsible for this service.

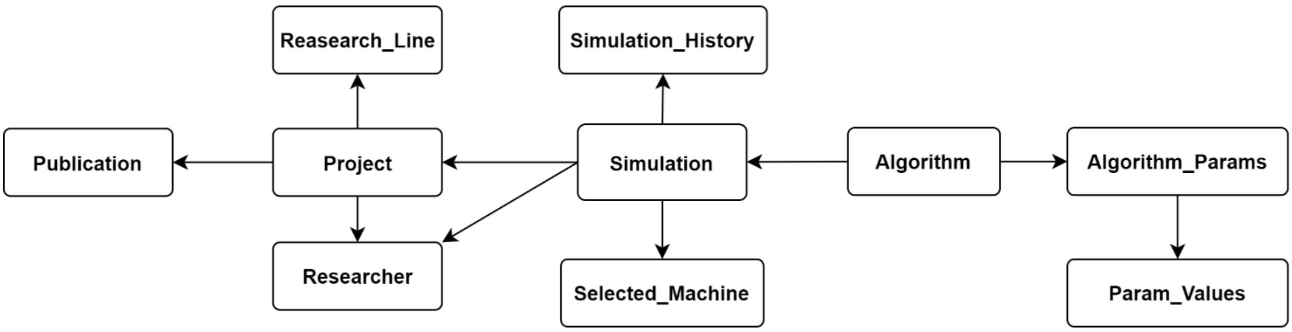
ERAS can also call the export service to obtain the query results in different formats. ERAS requests Danke using the method /search/results/export (Table 1, line 4), passing the conceptual query and the desired format (such as .csv and .xls) as input parameters.

3. Searching, retrieving, and exploring the ERAS data using Danke

This section describes an experiment with a sample of the Danke-kb that shows how Danke provides a data retrieval mechanism that does not require users to have specific technical skills for searching, retrieving, and exploring data.

As discussed before, the Danke-kb was built by extracting data from multiple and independent ERAS data collections, provided from different ERAS microservices. This experiment has a sample with data about research lines, projects, simulations, algorithms, researchers, and publications. Figure 2 depicts a partial RDF schema diagram of the ERAS sample data.

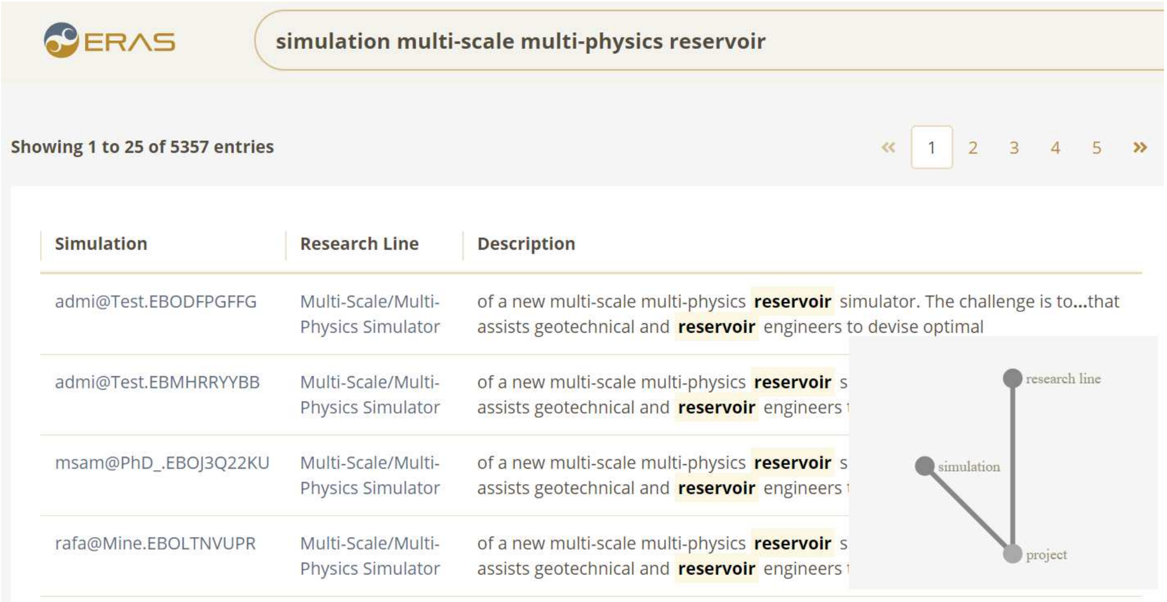
Figure 2 – RDF schema of the Danke-kb built from the ERAS data



Source: produced by authors.

The cost-effective management of Oil and Gas reservoirs is the motivation for developing new multi-scale multi-physics reservoir simulators. ERAS users are usually involved with the complexity of the simulation models that includes a wide range of scales in time and space. Therefore, they usually execute simulations and review previous analyses to reuse some data in new ones. Using ERAS integrated with Danke, they can query simulations of this research line, for example with "simulation multi-scale multi-physics reservoir". Figure 3 shows the keyword query results in tabular form.

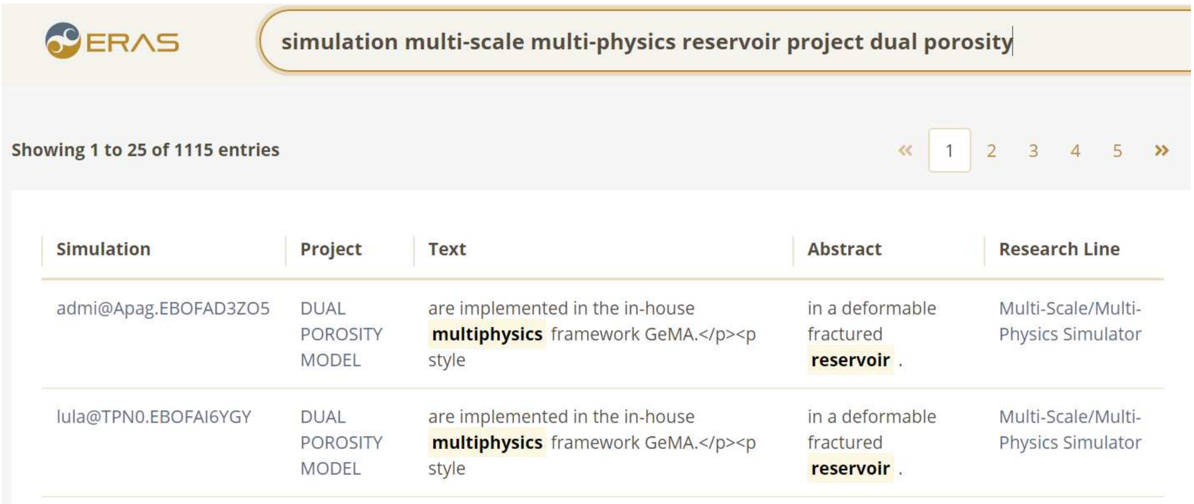
Figure 3 – Sample 1 of a data retrieval in ERAS Portal after integration with Danke



Source: produced by authors.

The answer of this query was the result of linking research line, project and simulation resources, as represented in the graph on the right bottom of the Figure 3. The user can filter the results adding more keywords, for example, with a project name such as “simulation multi-scale multi-physics reservoir project dual porosity”, as shown in Figure 4.

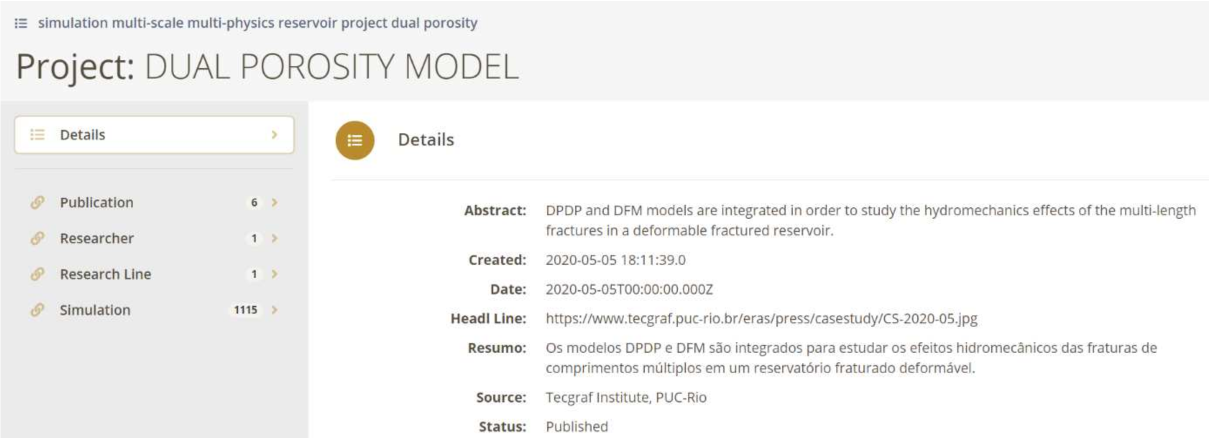
Figure 4 – Sample 2 of a data retrieval in ERAS Portal after integration with Danke



Source: produced by authors.

ERAS allows the user to navigate through the database, starting from rows of the answer. After the user clicks in a target resource, for example, the name of the project, ERAS responds with all information about the resource. Figure 5 shows the details of the project labeled “dual porosity model” and its linked data, as publications, researchers, research line, and simulations.

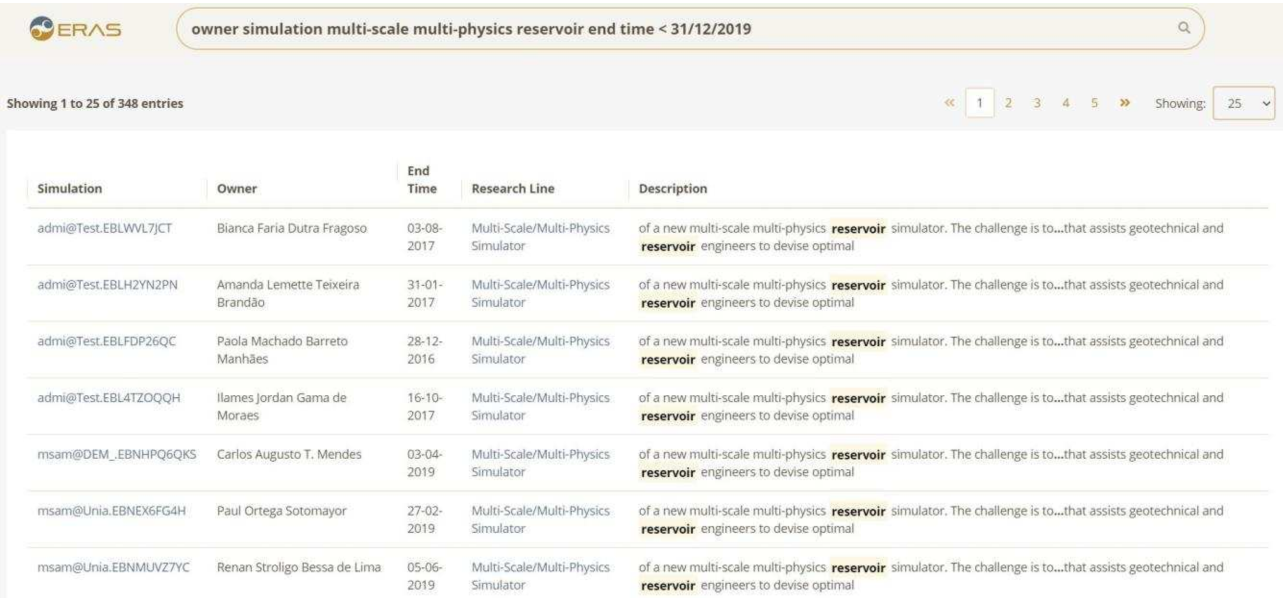
Figure 5 – Detail and linked data of a project



Source: produced by authors.

Since answers may be long tables, with many lines, ERAS can use the data summarization service that gives an overview of the answer to the user. For instance, if a user desires to retrieve the owners of simulations about the research lie multi-scale multi-physics ended before 2020, the query could be written as "owner simulation multi-scale multi-physics reservoir end time < 31/12/2020" and ERAS shows a result list illustrated in Figure 6 and a graphic illustrated in Figure 7. The graphic summarizes the number of simulations per owner.

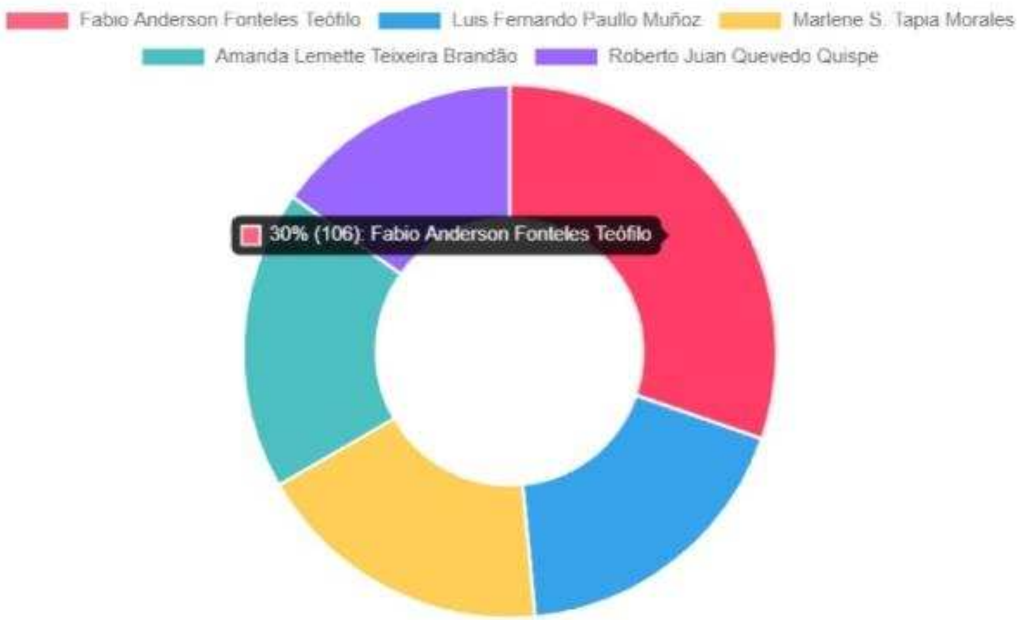
Figure 6 – Sample 3 of a data retrieval in ERAS Portal after integration with Danke



Simulation	Owner	End Time	Research Line	Description
admi@Test.EBLWVL7JCT	Bianca Faria Dutra Fragoso	03-08-2017	Multi-Scale/Multi-Physics Simulator	of a new multi-scale multi-physics **reservoir** simulator. The challenge is to...that assists geotechnical and **reservoir** engineers to devise optimal
admi@Test.EBLH2YN2PN	Amanda Lemette Teixeira Brandão	31-01-2017	Multi-Scale/Multi-Physics Simulator	of a new multi-scale multi-physics **reservoir** simulator. The challenge is to...that assists geotechnical and **reservoir** engineers to devise optimal
admi@Test.EBLFDP26QC	Paola Machado Barreto Manhães	28-12-2016	Multi-Scale/Multi-Physics Simulator	of a new multi-scale multi-physics **reservoir** simulator. The challenge is to...that assists geotechnical and **reservoir** engineers to devise optimal
admi@Test.EBL4TZOQQH	Ilames Jordan Gama de Moraes	16-10-2017	Multi-Scale/Multi-Physics Simulator	of a new multi-scale multi-physics **reservoir** simulator. The challenge is to...that assists geotechnical and **reservoir** engineers to devise optimal
msam@DEM_.EBNHPQ6QKS	Carlos Augusto T. Mendes	03-04-2019	Multi-Scale/Multi-Physics Simulator	of a new multi-scale multi-physics **reservoir** simulator. The challenge is to...that assists geotechnical and **reservoir** engineers to devise optimal
msam@Unia.EBNEX6FG4H	Paul Ortega Sotomayor	27-02-2019	Multi-Scale/Multi-Physics Simulator	of a new multi-scale multi-physics **reservoir** simulator. The challenge is to...that assists geotechnical and **reservoir** engineers to devise optimal
msam@Unia.EBNMUVZ7YC	Renan Stroligo Bessa de Lima	05-06-2019	Multi-Scale/Multi-Physics Simulator	of a new multi-scale multi-physics **reservoir** simulator. The challenge is to...that assists geotechnical and **reservoir** engineers to devise optimal

Source: produced by authors.

Figure 7 – An example of the query result summarization



Source: produced by authors.

The integration of ERAS with Danke avoids ERAS users write RSQL[2] (a RESTful interface for searching about simulations is available in ERAS) or SPARQL queries to retrieve data. For example, to retrieve the same result list provided by Danke using the keywords "simulation multi-scale multi-physics reservoir", users should write an RSQL query similar to:

```
simulation.project==project.id;project.research_line==research_line.id;research_line.description=="*reservoir*"
```

or a SPARQL query as (prefixes and other query details were omitted):

```
select distinct ?sim ?simulation_label ?research_line ?research_line_label ?research_line_description
where{
	?simulation rdf:type eras:Simulation ;
			rdfs:label ?simulation_label .
	?research_line rdf:type eras:Research_Line ;
			rdfs:label ?research_line_label;
			eras:research_line/description ?research_line_description .
	?simulation eras:simulation/project ?project .
	?project eras:project/research_line ?research_line
	FILTER regex(str(?research_line_description), "reservoir", "i")}
}
```

Note that users must be familiar with structured languages, for both cases (RSQL or SPARQL), which is often uncommon for geomechanical experts.

[2] RSQL is a query language for parametrized filtering of entries in RESTful APIs based on FIQL

4. Final Considerations

This paper described the benefits accrued by ERAS users by having an easy and efficient mechanism for data search, retrieval, and exploration using Danke platform. It first presented the strategy adopted to integrate ERAS with Danke, which is based on decisions that reinforce the data decentralization and scalability requirements of ERAS microservice architecture. It also covered the preparation of Danke-kb, whose data are ingested following a batch and real-time triplification process of the ERAS relational data, since Danke-kb is an RDF store. The paper also listed the services available in Danke and described how the communication between ERAS and Danke is performed. The paper concluded with an example based on real information needed by ERAS users that illustrates how simple it is to retrieve data in ERAS after integration with Danke. Note that a similar solution can be adopted for other systems, with an ERAS-like architecture, that intends to provide users with a data retrieval mechanism to their data collections.

ERAS is under development, and other microservices will provide new data collections that need to be integrated with the current solution. However, the effort to scale the proposed integration solution to add new data collections should be minimal. As future work, we plan to include unstructured data (data used in the input and output of the simulations) in the integration solution so that they are available to be queried. Finally, data provenance is relevant for ERAS users. Therefore, we plan to equip Danke with provenance ontologies to enrich the search vocabulary and, thereby, to better support ERAS users.

5. Agradecimentos

The authors gratefully acknowledge support from Shell Brasil through the “Coupled Geomechanics” project at Tecgraf/PUC-Rio Institute and the strategic importance of the support given by ANP through the R&D levy regulation.

Referências

Cerny, Tom, Donahoo, M., & Trnka, M. (2018). Contextual understanding of microservice architecture: current and future directions. *ACM SIGAPP Applied Computing Review*, *17*(4), 29–45. https://doi.org/10.1145/3183628.3183631

Dimou, A., Sander, M.V., Colpaert, P., Verborgh, R., Mannens, E., & de Walle, R.V. (2014). RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data. In *Proceedings of the 7th Workshop on Linked Data on the Web* (Vol. 1184, p.). Seoul, Korea. Retrieved fromhttp://ceur-ws.org/Vol-1184/ldow2014_paper_01.pdf

Dragoni N. et al. (2017). Microservices: yesterday, today, and tomorrow. In Present and ulterior software engineering. In *Present and Ulterior Software Engineering*. Mazzara M., Meyer B.: Springer, Cham. Retrieved from https://link.springer.com/chapter/10.1007/978-3-319-67425-4_12#citeas

Garcia, G.M. (2020). *A Keyword-based Query Processing Method for Datasets with Schemas*. Rio de Janeiro: PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO - PUC-RIO. Retrieved from https://www.maxwell.vrac.puc-rio.br/colecao.php?strSecao=resultado&nrSeq=48728@1

Izquierdo, Y. T. et al. (2018). Quiow: A Keyword-Based Query Processing Tool for RDF Datasets and Relational Databases. (Vol. 11030, pp. 259–269.). Presented at the Database and Expert Systems Applications, Regensburg, Germany. https://doi.org/10.1007/978-3-319-98812-2_22

Lima, M. J. et al. (2018). Functional Requirements for Developing ERAS-A Portal to Support Collaborative Geomechanical Simulations (Vol. , pp. 1–8). Presented at the 12 Brazilian e-Science Workshop (BreSci), Anais do XII Brazilian e-Science Workshop. https://doi.org/10.5753/bresci.2018

Sampaio, A. R., et al. (2019). Improving microservice-based applications with runtime placement adaptation.*Journal of Internet Services and Applications*, *10*(4). https://doi.org/10.1186/s13174-019-0104-0

Schmutz, G. (2018). Ingesting streaming data into Graph Database (Vol. , p.). Presented at the Analytics and Data Summit 2018, San Francisco. Retrieved from https://www.slideshare.net/gschmutz/ingesting-streaming-data-into-graph-database