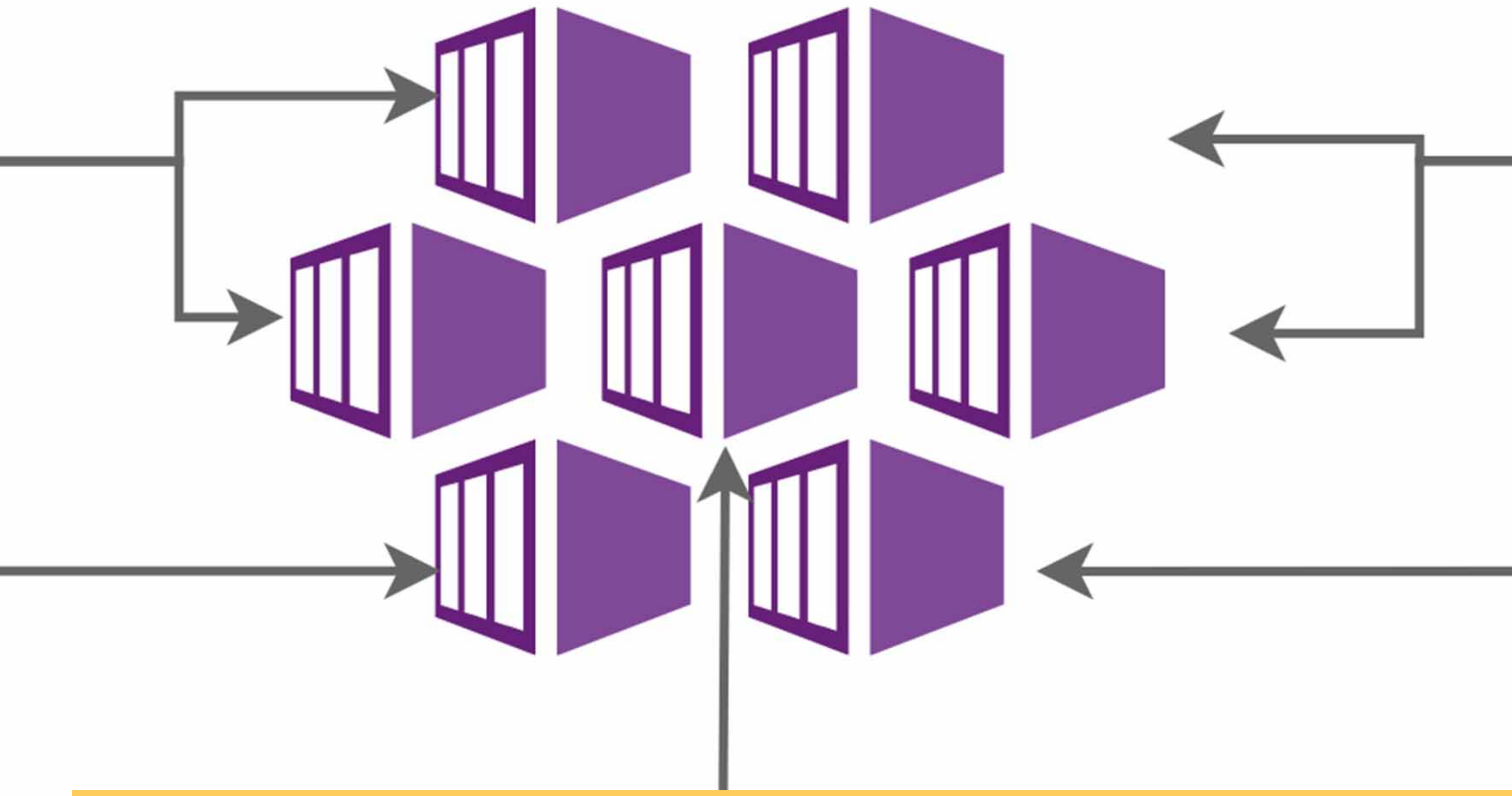
ISSN 2039-7941
anno 2023 numero 476



RAPPORTI
TECNICI INGV

OEDataRep: the new version of the Osservatorio Etneo Open Data Repository

476

ISTITUTO NAZIONALE DI GEOFISICA E VULCANOLOGIA

REGISTRAZIONE AL TRIBUNALE DI ROMA N.174 | 2014, 23 LUGLIO
© 2014 INGV Istituto Nazionale di Geofisica e Vulcanologia | Rappresentante legale: Carlo DOGLIONI
Sede: Via di Vigna Murata, 605 | Roma

ISTITUTO NAZIONALE DI GEOFISICA E VULCANOLOGIA

ISSN 2039-7941
anno **2023** numero **476**

RAPPORTI
TECNICI INGV

OEDataRep: the new version of the Osservatorio Etneo Open Data Repository

Fabrizio Pistagna[1,*], Mario Torrisi[1], Carmelo Cassisi[1], Mario Locati[2], Placido Montalto[1]

[1]INGV | Istituto Nazionale di Geofisica e Vulcanologia, Sezione di Catania - Osservatorio Etneo
[2]INGV | Istituto Nazionale di Geofisica e Vulcanologia, Sezione di Milano

[*]*Corresponding author*

Accepted 21 November 2023 | *Accettato 21 novembre 2023*

How to cite | *Come citare* F. Pistagna, M. Torrisi, C. Cassisi, M. Locati, P. Montalto, (2023). OEDataRep: the new version of the Osservatorio Etneo Open Data Repository. Rapp. Tec. INGV, 476: 1-34, https://doi.org/10.13127/rpt/476

Cover | *In copertina* OEDataRep main features: Time Series definition and integration and a modern Microservices oriented deployment (detail)

476

INDEX

Abstract

The use of a generalist, non field-specific Open Data based Repository, started at the INGV Osservatorio Etneo of Catania (INGV-OE) following the introduction of a first prototype, called DataRep in late 2021. During this experimental period, DataRep was used as the institutional data repository of the INGV-OE, testing its functionalities, usage procedures and the interaction with the national INGV metadata catalog ("Data Registry") through its associated Metadata Editor software. Due to the heterogeneous nature of data types managed at INGV-OE, mainly structured as temporal data (also known as time-series) or catalogs of events (e.g. earthquakes, eruptions), there has been an increasing interest in making the data repository interact with another key software developed and distributed by the INGV-OE: the TSDSystem (TimeSeries Database System) framework. Among other features, the TSDSystem facilitates the collection of time-series from several sources, supporting their standardization within a unique and coherent database structure, and allowing the retrieval of data in a convenient way by easily providing joint requests of multiple time-series at once that may be displayed on the same time axis. From the interoperability of the three software involved - the Metadata Editor, the data repository and the TSDSystem - users may uniquely interact with the *web* graphical user interface of the Metadata Editor in order to describe the metadata associated with a future publication and upload them together with the related files to the data repository. The latter will then act as a client of the services exposed by the TSDSystem. Following this workflow, datasets that include time-series data published to the repository, will also be put atomically into the TSDSystem database. Likewise, the same time-series data will be retrieved from the TSDSystem and displayed by the repository web interface. The integration of the three software platforms is at the core of a new version of the DataRep repository that is now being phased out in favor of a new implementation called OEDataRep, the acronym of Osservatorio Etneo Open Data Repository. This new version involved a complete overhaul of the data repository software that has considerably improved both the underlying infrastructure and the graphical user experience.

Keywords Open Data Repository; Open Science; Time Series database

Introduction

The operations of gathering and collecting raw data from the multi-parametric monitoring network distributed throughout the Sicilian volcanic areas is a key task carried out by the INGV Osservatorio Etneo (INGV-OE) based in Catania. Data is subsequently analyzed, processed and cataloged in structured datasets. Generating datasets and making them available, possibly in real-time, is among the core responsibilities of INGV-OE because of their prime importance in the surveillance for civil protection purposes and scientific research.
With the aim of distributing such data following the Open Data model[1], researchers and technologists publish and distribute their scientific products as datasets on an institutional and publicly accessible data repository conforming to the Open Science paradigm [2], and adopting

[1] Open Data is data that is openly accessible, exploitable, editable and shared by anyone for any purpose. Open Data is licensed under an open license [1]. The Open Data model requires that scientific publications results, produced by research organizations or institutes must be freely accessible and usable by anyone, with the only constraint of citing the source.

the FAIR data principles [Wilkinson 2016] requiring data to be findable, accessible, interoperable, and reusable.
The first prototype of the repository was presented in late 2021 under the name DataRep and the work undertaken for its implementation, as well as its characteristics, have been described in the related technical report [Torrisi et al., 2022].
The work described in this technical report details both the implementation and the deployment[2] of a new implementation of the Open Data Repository in use at the INGV-OE. This new release is called OEDataRep, and is the result of an evolution of the aforementioned old prototype data repository (see chapter 2.1).
OEDataRep is the only generalistic, non-field-specific data repository available at INGV where researchers can deposit any type of data that is not managed by any other disciplinary-specific data repository[3].
Both the processed and raw data produced and acquired at the INGV-OE, will then be published in this new data repository following the FAIR data principles and freely distributed as the Open Access directives dictate.
The procedure to publish data in OEDataRep remains mostly unchanged, as documented in Torrisi et al. [2022], and involves the use of a Metadata Editor, a tool for entering metadata in the INGV institutional metadata catalog called Data Registry [5] managed by the Data Management Office. The INGV Data Registry supports the DataCite metadata schema [DataCite WG, 2021] [6], among other metadata standards.
From the user perspective, the dataset description and upload procedure is mostly unchanged with the new version: it involves the use of the Metadata Editor, which acts as an intermediary between the user and the OEDataRep repository. The Metadata Editor interacts with the data repository via API (Application Programming Interface) resources, in order to both create and modify the records using the defined metadata, and upload data. More details on the uploading procedure is provided in chapter 2.
Using the Metadata Editor as a gateway for the definition of a publication (record) on the OEDataRep repository has multiple advantages:

- It keeps the institutional Data Registry updated;
- It guarantees the coherency of metadata through all INGV data platforms;
- It allows an institutional validation[4] of which datasets are being published;
- It provides users a unique metadata platform for any type of data;
- It allows a centralized DOI (Digital Object Identifier) minting, codes of fundamental importance in the Open Science paradigm since they allow a worldwide unique identification and description of data using a well-recognised international standard.

The new implementation of the OEDataRep data repository has led to a better interactivity between the two systems (Metadata Editor and OEDataRep), which has simplified the previous process of loading the data (see chapter 2).
During the development of this new implementation of the OEDataRep data repository INGV-OE Information Technology developers collaborated with the INGV-OE IT engineers who developed the TSDSystem framework [Cassisi et al., 2015] [8]. The cooperation among the two groups led to improvements on both platforms on many aspects related to the processing and distribution of temporal-type datasets, the so-called time-series[5].

[2] Software Deployment is all of the activities that make a software system available for use. The general deployment process consists of several interrelated activities with possible transitions between them [3].

[3] List of field-specific data repository managed by INGV [in Italian] [4]

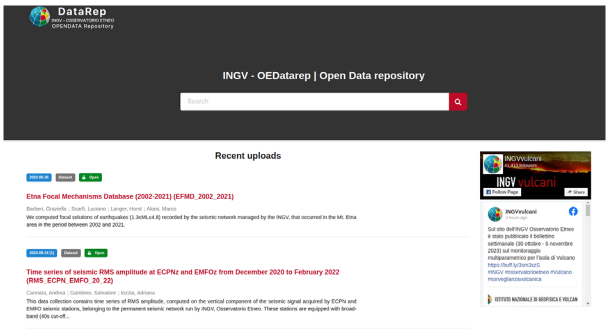
[4] The validation procedure required for publishing data is defined by the INGV Data Policy, see [7]

[5] A time-series is a series of data points indexed in time order, often graphed in charts or listed in structured text files [9].

The new implementation of the data repository has also involved a new deployment strategy totally oriented towards microservices pattern [10] [11] [Newman, 2021; Jaramillo et al., 2016], much more suitable and better exploiting the capabilities of the data center infrastructure available at the INGV-OE. This strategy was designed from the outset to achieve a production grade deployment, as it drastically increased the scalability in some of its components (services), to be resilient and reliable (see chapter 3).
All these improvements and the relative adopted deployment solutions are described in the following chapters.

1. OEDataRep: the Osservatorio Etneo Open Data Repository

The DataRep Open Data Repository has been online at the INGV-OE since late 2021, and its development has seen a close collaboration between members at the INGV-OE (both people from IT departments and researchers and technicians) and members at the Data Management Office. Since then, the prototype DataRep repository has been used by researchers, technologists and technicians affiliated to the INGV-OE for publishing datasets. From the feedback gathered by the usage of the prototype a series of issues emerged, leading to the need for a new release of the data repository which is described in the following chapters and is the main object of this technical report. In particular, the developers decided to improve the prototypal data and metadata upload procedure via the Metadata Editor [5] [Torrisi et al., 2022], and above all add further integration with other software used and developed at INGV-OE.
The work carried out during this phase also led INGV-OE software engineers to perform a reengineering of the TSDSystem framework [8] - one of the most widely used data management platforms within the INGV-OE - in order to embrace a modern architecture based on microservices and making it easier to integrate it within the deployment environment where the new implementation of the DataRep repository would reside.
During the trial period of the data repository prototype, the research community showed their appreciation of its functionalities and usefulness, leading to the long-term plan to bring the data repository to a more complete all-inclusive software platform with the aim of being the entry point to access the Osservatorio Etneo Open Data datasets. For that reason, the new name of the repository became OEDataRep (https://oedatarep.ct.ingv.it/) (Figures 1and 2).



INGV - OEDatarep | Open Data repository

Recent uploads

2023-06-30 Dataset Open

Etna Focal Mechanisms Database (2002-2021) (EFMD_2002_2021)

Barberi, Graziella ; Scarfì, Luciano ; Langer, Horst ; Aloisi, Marco

We computed focal solutions of earthquakes (1.3≤ML≤4.8) recorded by the seismic network managed by the INGV, that occurred in the Mt. Etna area in the period between 2002 and 2021.

2023-09-19 (1) Dataset Open

Time series of seismic RMS amplitude at ECPNz and EMFOz from December 2020 to February 2022 (RMS_ECPN_EMFO_20_22)

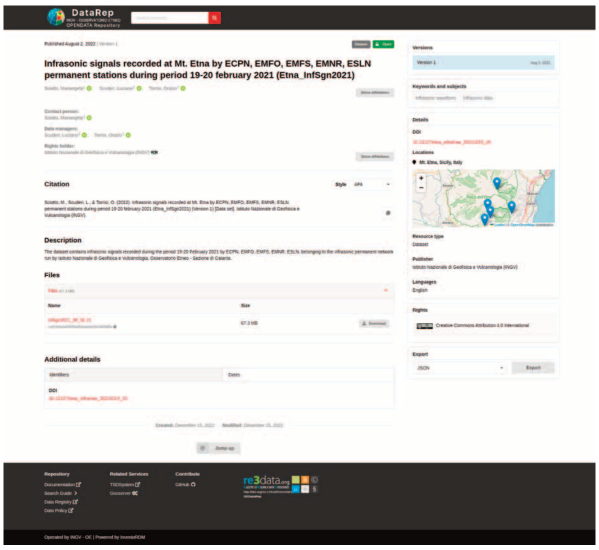
Cannata, Andrea ; Gambino, Salvatore ; Iozzia, Adriana

This data collection contains time series of RMS amplitude, computed on the vertical component of the seismic signal acquired by ECPN and EMFO seismic stations, belonging to the permanent seismic network run by INGV, Osservatorio Etneo. These stations are equipped with broad-band (40s cut-off...

INGVvulcani
3 hours ago

Sul sito dell'INGV Osservatorio Etneo è stato pubblicato il bollettino settimanale (30 ottobre - 5 novembre 2023) sul monitoraggio multiparametrico per l'isola di Vulcano https://buff.ly/3sm3xzS #INGV #osservatorioetneo #Vulcano #sorveglianzavulcanica

Figure 1 The new home page of the OEDataRep repository, presenting a list of the recent published record, including title, description, type of publication and a cover image on the right side.



DataRep
INGV - OSSERVATORIO ETNEO
OPENDATA Repository

Published August 2, 2022 | Version 1

Dataset | Open

Infrasonic signals recorded at Mt. Etna by ECPN, EMFO, EMFS, EMNR, ESLN permanent stations during period 19-20 february 2021 (Etna_InfSgn2021)

Sciotto, Mariangela[1]; Scuderi, Luciano[1]; Torrisi, Orazio[1]

Contact person: Sciotto, Mariangela[1]

Data managers: Scuderi, Luciano[1]; Torrisi, Orazio[1]

Rights holder: Istituto Nazionale di Geofisica e Vulcanologia (INGV)

Citation Style: APA

Sciotto, M., Scuderi, L., & Torrisi, O. (2022). Infrasonic signals recorded at Mt. Etna by ECPN, EMFO, EMFS, EMNR, ESLN permanent stations during period 19-20 february 2021 (Etna_InfSgn2021) (Version 1) [Data set]. Istituto Nazionale di Geofisica e Vulcanologia (INGV).

Description

The dataset contains infrasonic signals recorded during the period 19-20 February 2021 by ECPN, EMFO, EMFS, EMNR, ESLN, belonging to the infrasonic permanent network run by Istituto Nazionale di Geofisica e Vulcanologia, Osservatorio Etneo - Sezione di Catania.

Files

Files (67.3 MB)

Name	Size
[illegible]	67.3 MB

Additional details

Identifiers | Dates

DOI [illegible]

Created: December 15, 2022 Modified: December 15, 2022

Versions: Version 1, Aug 2, 2022

Keywords and subjects: Infrasonic waveform; Infrasonic data

Details

DOI [illegible]

Locations: Mt. Etna, Sicily, Italy

Resource type: Dataset

Publisher: Istituto Nazionale di Geofisica e Vulcanologia (INGV)

Languages: English

Rights: Creative Commons Attribution 4.0 International

Export: JSON

Repository: Documentation; Search Guide; Data Registry; Data Policy
Related Services: TSDSystem; Geoserver
Contribute: GitHub

Operated by INGV - OE | Powered by InvenioRDM

Figure 2 Screenshot of a record's landing page, with full metadata details describing the publication, including DOI and Citation method.

1.1 From Zenodo to InvenioRDM

The development of the new OEDataRep data repository involved an important evolution from the point of view of the software that composes it. The first DataRep repository release was based on the Zenodo software[6], an implementation of a general purpose Open Data Repository built on top of Invenio framework [Robinson et al., 2007; Caffaro et al., 2010; Morell et al., 2019; Sheoran et al., 2021][7]. Invenio is an open source software framework initially created by CERN to manage their institutional document repository, the so called CDS, CERN Document Server. From the effort of the CERN developers, the InvenioRDM project was born, which implements a generalist digital repository [Morell et al., 2019], defined as "The turn-key research data management repository" by their developers [14], with a well-defined governance [15].
While Zenodo is actually a repository service hosted by CERN (even though its code can be customized), InvenioRDM is a repository application that anyone can use to run a service similar to Zenodo. Furthermore, the current implementation of Zenodo's software architecture relies on outdated libraries leading to compatibility issues and inconsistencies, whereas InvenioRDM is a much more modern and reliable software platform. Additionally, InvenioRDM is designed to be modular and flexible, making it easy to customize and adapt to the specific needs of research communities. It provides a range of useful features including data curation, preservation, access control, and metadata management. The operations of extending and customizing InvenioRDM toward specific needs of research communities can be done in several ways at different levels of the software stack, and, very important when dealing with Open Source software, it is very well documented.
The need to create a more robust and flexible software platform that allows users to take full advantage of a next generation institutional Open Data Repository, led to the adoption of InvenioRDM.
A non-exhaustive comparative list of the main differences between Zenodo and InvenioRDM is listed in Table 1 [Morell et al., 2019].

[6] Zenodo is a general-purpose open repository developed under the European OpenAIRE program and operated by CERN [12]. It allows researchers to deposit research papers, data sets, research software, reports, and any other research related digital artefacts. For each submission, a persistent Digital Object Identifier (DOI) is minted, which makes the stored items easily citable.

[7] Invenio is an open source software framework for large-scale digital repositories that provides the tools for management of digital assets in an institutional repository and research data management systems [13].

Zenodo	InvenioRDM
Is a repository service	Is a repository application that can be used to run a repository service
Open source	Open source
Dated technology, nearly impossible to advance	Modern web architecture and standards that make it easy to deploy, maintain, and use
No end-to-end support for Next Generation Repository [NGR; Rodrigues et. at 2017]	Implements the Next Generation Repository (NGR) vision
Minimal interoperability	Integrates well with other Open Science infra-structures such as ORCID, DataCite and follows the OpenAIRE guidelines.

Table 1 Zenodo and InvenioRDM comparison.

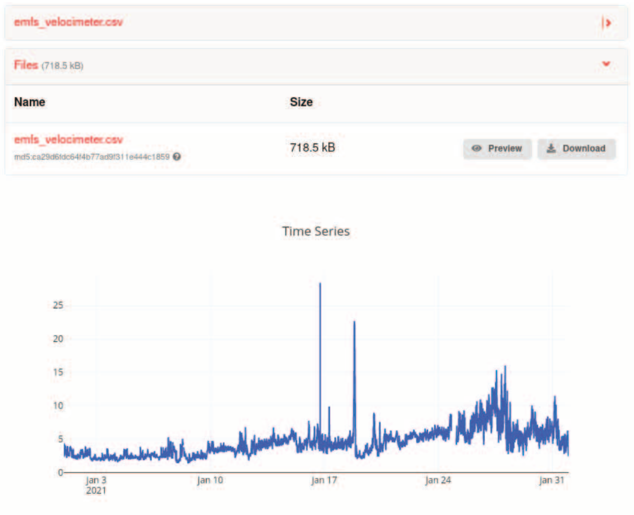
1.2 TSDSystem integration

Among the main activities related to the acquisition, collection and cataloging of raw data obtained by the environmental monitoring networks carried out by the staff of the INGV-OE, there is certainly much interest in the efficient management of time-series datasets. Within the INGV-OE, since 2015, a software system has been developed - TSDSystem [Cassisi et al., 2015] [8] - in order to store, catalog and manipulate, as well as querying the time-series data.
As reported in the last chapter of Torrisi et al. [2022], there was a plan to integrate the TSDSystem's capabilities within the Open Data Repository allowing the two software to interact with each other. The aim was to enable the two software to be more extensively interoperable in order to provide a single experience regarding the storage of temporal datasets. The idea was to atomically store a time-series dataset both on the OEDataRep repository - as a standard Open Data dataset file, and on a dedicated TSDSystem instance deployed in conjunction with the OEDataRep repository (see chapter 3.2) - thanks to its CRUD[8] set of RESTful APIs. This was the work carried out by the two OE Information Technology teams that worked on both systems and which is described in the following paragraphs.
By the integration between the two systems, the TSDSystem becomes an integral component of the OEDataRep repository, acting as a delegated backend service to efficiently store and manage time-series temporal data, while OEdataRep acts as a client that exploits TSDSystem RESTful APIs to store and query time-series data. A specifically developed InvenioRDM module [16] was created to provide the missing features required for the new implementation of the OEDataRep repository: introducing support to the metadata description of time-series data while publishing a new dataset. This custom module acts as an interface between the record creation on the data repository and the corresponding operation of creating an entry for the time-series dataset into the TSDSystem database.
The integration of OEDataRep and the TSDSystem made it possible:

- Regarding the user interface (UI) of the OEDataRep repository, to enhance the user experience by enabling the visualization of time-series data on the repository record landing pages through interactive charts (Figure 3). Moreover, users can now seamlessly preview and explore time-series data within the same repository's record landing page web view.
- Regarding the backend, to drive the operations of uploading the time-series resources into the TSDSystem in an atomic operation, ensuring synchrony between the data

[8] The acronym CRUD (Create, Read, Update, and Delete) refers to the major operations implemented by databases.

described in a repository record (basically an Open Data time-series dataset) and the same data loaded and archived (in a structured way on an ad-hoc relational schema) on the TSDSystem database.



emls_velocimeter.csv

Files (718.5 kB)

Name	Size	
emls_velocimeter.csv md5:ca29d6fdc64f4b77ad9f311e444c1859	718.5 kB	Preview Download

Figure 3 Example of a time-series chart preview associated to a dataset as can be seen in OEDataRep web page. The temporal data contained in the uploaded file (.csv in the example above) are uploaded to the TSDSystem, and then retrieved from the same set of REST APIs in order to be displayed as in the interactive chart above (zoomable).

1.2.1 TSDSystem overview and components

Here, is provided an overall description of the TSDSystem framework to better understand how it was re-engineered to fit with a more scalable environment and make it easier to integrate in a deployment methodology based on infrastructure as code[9] pattern [17] (described in the next chapter), a standard approach in software development at INGV-OE.
The TSDSystem framework aims to offer a solution for managing and storing data obtained from environmental monitoring networks based on sensors. Conceptually speaking, it could be subdivided in two core services:

- Backend service, based on the TimescaleDB [18][19] flavor of PostgreSQL [20]: an open-source time-series database developed by Timescale Inc. It is written in C and extends PostgreSQL.
- Frontend (or Application) service, which implements and exposes the REST APIs resources to interact with the system. This service consists of a set of REST APIs that facilitate the handling of time-series data using the standard HTTP protocol, enabling CRUD operations on data residing on the backend database.

Developed and maintained at INGV-OE, the TSDSystem is adopted by INGV sections of Catania and Palermo. TSDSystem is also used for the activities envisaged by Working Packages 6 and 8 of the DPC ("Dipartimento Protezione Civile") - INGV agreement[10].
As mentioned, during the design of the new OEDataRep, the TSDSystem was re-engineered, shifting from a monolithic software design to a microservices architecture [20] based on multiple Docker containers [21]. A series of existing services has been subdivided into smaller, independent, services that can be deployed and scaled separately.

[9] Infrastructure as code (IaC) uses DevOps methodology and versioning with a descriptive model to define and deploy infrastructure, such as networks, virtual machines, load balancers, and connection topologies. Just as the same source code always generates the same binary, an IaC model generates the same environment every time it deploys.

[10] Convention for service activities in undertaking the agreement between the Department of Civil Protection and the National Institute of Geophysics and Volcanology for activities referred to in paragraphs A) relative to risk and hazard assessment (Italian original: https://istituto.ingv.it/images/Accordi-dpc/Convenzione_INGV_2022-2024.pdf).

The services that compose the new deployment of the TSDSystem (Figure 4) are:

- The main core TSDSystem Frontend application, written in PHP language, provides core functionality for interacting with the REST APIs. The Docker image that constitutes this service includes an HTTP Apache server [22] handling HTTP requests and serving the core business logic of the application.
- The counterpart of the TSDSystem application, is identified by the backend service, which is built upon a Docker image with the TimescaleDB database.
- Additionally, a proxy frontend component, powered by Nginx [23], is integrated as a supplementary service into the architecture. Proxy acts as a gateway, routing requests and managing communication between the client and the backend components.

Supplementary components have been added to the TSDSystem deployment, making it a full-fledged and comprehensive platform for time-series data:

- pgAdmin [24] service to facilitate underlying database management;
- Grafana [25] service to enhance the platform's capabilities by providing powerful data visualization and analytics features.

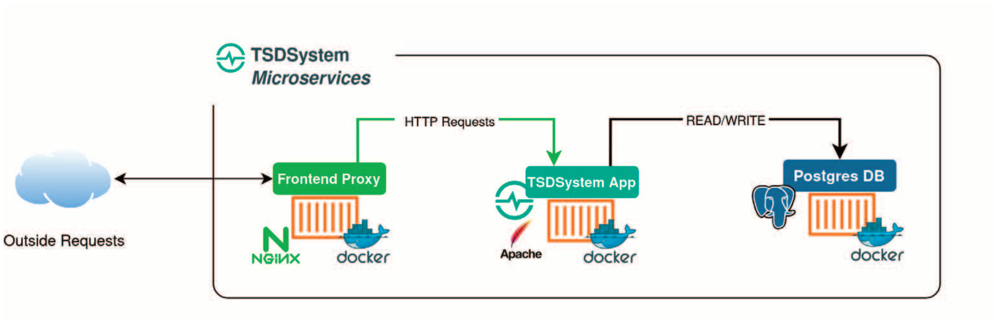


Figure 4 Services composing the new deployment of the TSDSystem based on a Microservices pattern, where several and uncoupled Docker containers work together to set up the final service. A Frontend (optional) service based on NGinx acts as reverse proxy and redirects requests to the main core TSDSystem App, a container based on an Apache web server which includes the whole TSD business logic; at the end there is a database service, implemented by a Postgres + Timescale DB.

1.2.2 OEDataRep and TSDSystem cooperation: Time-Series publication procedure

The aforementioned module that ensures the integration between OEDataRep and TSDSystem has been developed following the official guidelines provided by the InvenioRDM documentation [26]. It consists of a brand-new InvenioRDM service layer [27]. As stated in the official documentation, the service layer contains the domain and business logic of the application and is responsible for: authorization (i.e. checking permissions), business-level validation, control flow. The service layer usually works inside an Invenio module, a package named service. It may consist of several logic parts, according to its functionality. In the specific case of OEDataRep, the new module implements both a service component and a background task.
The module, developed in Python, exploits the Celery task queue [28] [29], distributed as an external service with InvenioRDM. Celery is an open source asynchronous task queue based on distributed message passing. The new module, called OEDataRep TimeSeries Loader [16] - code name "oedatarep-ts-loader" - defines an "execution unit" for Celery (formally a task), that will be executed asynchronously in the background. Moreover, the module represents a way to

exploit the entire InvenioRDM software stack; i.e. the “oedatarep-ts-loader” module also interacts with the ElasticSearch [30] software service within an InvenioRDM deployment.
The implemented logic flow consists of the following steps (Figure 5):

1. as soon as a new record containing time-series data has been created and published from the Metadata Editor, OEDataRep stores all the data and metadata as usual, but a new custom metadata will be specified (see next chapter for details) which defines the presence of a time-series dataset;
2. the “oedatarep-ts-loader” module, which implements the time-series loader service, defines a periodically scheduled task in charge of querying the OEDataRep ElasticSearch service. The task will check for new published records with the aforementioned custom time-series metadata defined, that are not yet synchronized on the TSDSystem.
    a. ts_published: false, a boolean type metadata which identifies time-series data to be loaded on the TSDSystem service;
3. if there are records with time-series data that have to be synchronized to the TSDSystem backend, a new asynchronous task will be created and added to the Celery queue with the “record_id” field as input; this task will be responsible for uploading the data and the associated metadata on the TSDSystem backend service.

The asynchronous task that loads time-series data:

1. uses the “record_id” input to retrieve the time-series data associated with the record by exploiting the InvenioRDM internal REST API resources [31];
2. interacts with the REST APIs resources of the TSDSystem instance associated with the repository, in order to first define (create) and then store (upload) the time-series data associated with the record.

The complete application flow describing the interactions between all the components of an OEDataRep deployment is shown in the Activity Diagram of (Figure 5). Publication flow begins (1) from describing the record’s metadata on the Metadata Editor. An additional metadata key, “ts_resource”, is specified for the definition of a time-series dataset. The custom InvenioRDM service “oedatarep-ts-loader” defines (2) a celery task which periodically checks for new published records with the “ts_resources” metadata set but not yet synchronized to TSDSystem backend (“ts_published” key set to false). When these conditions are met (3), the celery worker component will retrieve (4) the time-series data from the record and will PUT (5) them to the TSDSystem, through the appropriate REST APIs resources.

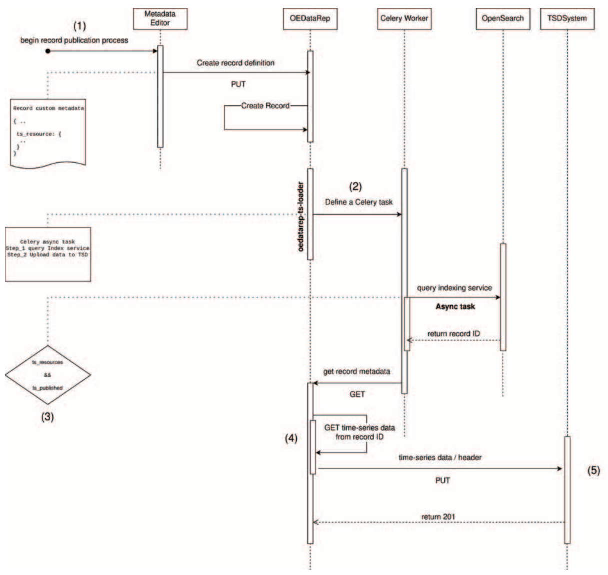
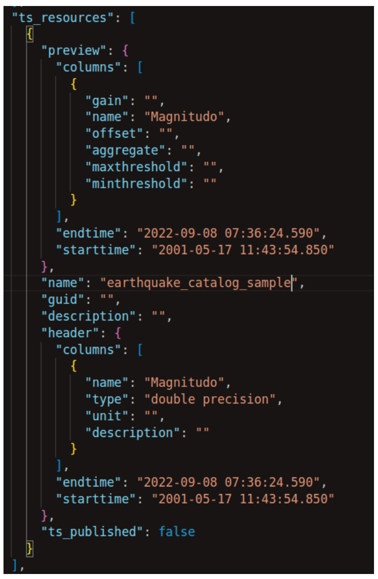


Figure 5 Activity Diagram describing the interaction between all the components of an OEDataRep deployment.

1.3 OEDataRep time-series resource metadata structure

Publishing a dataset containing a time-series to the OEDataRep repository, has involved the extension of the original InvenioRDM repository's record metadata schema [32]. This extension was required in order to properly store time-series resource metadata, so the standard InvenioRDM module invenio-rdm-record [33] was customized.
InvenioRDM's bibliographic records are stored as JSON documents in a structure that is compliant with the DataCite's Metadata Schema v4.x [DataCite WG, 2020] with some relevant additions. This metadata structure has been extended to include time-series resources.
Metadata are fundamental pillars in data repository and InvenioRDM has a powerful system to define and maintain new metadata. In order to integrate datasets that belong to the field of time-series, a new metadata has been defined and called ts_resources, which indicates the presence of a time-series dataset among the resources published in the record.
Following the InvenioRDM metadata definition convention, the ts_resources metadata cardinality is (0-n), which means it is not a mandatory metadata. Furthermore, its array type structure definition states that it could contain a set of time-series resources.
The ts_resource metadata structure, detailed in the following chapters, was defined by TSDSystem and OEDataRep repository developers in coordination with the INGV Data Management Office. As a general approach, the structure was left as general as possible, in order not to be tightly coupled with domain specific use cases. The ts_resources metadata schema defined for the current OEDatarep repository release is available on the public GitHub repository [34].
An example of ts_resources metadata is shown in (Figure 6).



```
"ts_resources": [
  {
    "preview": {
      "columns": [
        {
          "gain": "",
          "name": "Magnitudo",
          "offset": "",
          "aggregate": "",
          "maxthreshold": "",
          "minthreshold": ""
        }
      ],
      "endtime": "2022-09-08 07:36:24.590",
      "starttime": "2001-05-17 11:43:54.850"
    },
    "name": "earthquake_catalog_sample",
    "guid": "",
    "description": "",
    "header": {
      "columns": [
        {
          "name": "Magnitudo",
          "type": "double precision",
          "unit": "",
          "description": ""
        }
      ],
      "endtime": "2022-09-08 07:36:24.590",
      "starttime": "2001-05-17 11:43:54.850"
    },
    "ts_published": false
  }
],
```

Figure 6 ts_resource metadata sample.

Furthermore, additional minor changes were applied to enrich the set of metadata already available; all of the significant add-ons are then shown on the record landing page. As an example of add-ons:

- method (0-1) is a textual field, descriptive metadata, used to describe the way dataset data were collected;
- cover (0-1) is a textual field, describing the URL pointing to the INGV metadata catalog cover image resource, if it is available.

InvenioRDM currently uses Elasticsearch as its underlying search engine, and there is a plan for its replacement with its open source fork named Open search[11] in a future release. Since Elasticsearch is fully JSON-based, it fits well together with storing records internally in the database directly as JSON documents[12]. Each document is a collection of fields. Mapping is the process of defining how a document, and the fields it contains, are stored and indexed.
Following this concept, a mapping has been defined that indexes the ts_published metadata, which is then used by the oedatarep-ts-loader module during the process that automatically uploads time-series data on the TSDSystem (see chapter 2.2.2). This boolean field is used by the search indexing system to find those records that have not yet been published in the TSDSystem database but need to be. The mapping defined for the current OEDatarep repository release is also available on the corresponding public GitHub repository [37].

1.3.1 Metadata attributes and details

This chapter contains a detailed description of all metadata fields that were added during the customization of the InvenioRDM-record module.

1.3.1.1 ts_resources (0-n)

The "ts_resources" metadata describes the time-series resource defined in the record and that is published or that is going to be published in the TSDSystem; it has (0-n) cardinality. Table 2 shows the list of attributes, their cardinality, the type of data stored and brief description of them.

Field	Cardinality	Type	Description
`guid`	`0-1`	`String`	Time-series resource unique identifier in TSDSystem
`name`	`0-1`	`String`	Time-series resource name
`tsdws_url`	`0-1`	`String`	Time-series resource url in TSDSystem, used to query data
`ts_published`	`1`	`boolean`	Indicates if time-series was published on TSDSystem or not yet
`header`	`1`	`object`	Object to describe time-series specific characteristics
`preview`	`1`	`object`	Object to define value used to plot time-series preview in record landing page
`description`	`0-1`	`String`	A general description about time-series
`additional_info`	`0-1`	`object`	Further details stored in TSDSystem as JSON object

Table 2 ts_resouces metadata attributes.

[11] OpenSearch is a scalable, flexible, and extensible open-source software suite for search, analytics, and observability applications licensed under Apache 2.0 [35]

[12] JSON is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and arrays [36].

1.3.1.2 ts_resources.header (1)

This attribute is mandatory and is used to describe time-series specific characteristics; Table 3 shows its attributes.

Field	Cardinality	Type	Description
`starttime`	`1`	`datetime`	Time-series start time
`endtime`	`0-1`	`datetime`	Time-series end time, could be undefined for dynamic time-series
`columns`	`1-n`	`object`	Describe time-series columns
`sampling`	`0-1`	`integer`	Indicates time-series sampling interval (in seconds)

Table 3 ts_resouces.header metadata attributes.

1.3.1.3 ts_resources.header.columns (1)

Going down in the time-series metadata structure, the columns array attribute is used to describe columns in the time-series dataset, one object per each column. The object structure of the columns array is shown in Table 4.

Field	Cardinality	Type	Description
`name`	`1`	`string`	Time-series column name
`type`	`1`	`string`	String represents data format, could be one of: • `smallint` • `integer` • `double precision`
`unit`	`0-1`	`string`	String represents unit measure, as example: `m/s`, `kg/m`^{3}, `µrad`, …
`description`	`0-1`	`string`	Text field for long descriptive information

Table 4 ts_resouces.header.columns metadata attributes.

1.3.1.4 ts_resources.preview (1)

The preview metadata object, used to define values to plot time-series preview in the OEDataRep record landing page; its structure is shown in Table 5.

Field	Cardinality	Type	Description
`starttime`	1	`datetime`	Time-series preview start time
`endtime`	1	`datetime`	Time-series preview end time
`columns`	1-n	`object`	Time-series preview columns
`sampling`	0-1	`integer`	Indicates time-series preview sampling interval (in seconds)
`aggregation`	0-1	`string`	Aggregation function used to plot time-series preview data. Could be one of: • `AVG: average` • `MIN: min` • `MAX: max` • `SUM: sum` • `COUNT: count` • `MEDIAN: median` It is generally applied on all listed columns. If not specified, raw data will be plotted.

Table 5 ts_resouces.preview metadata attributes.

1.3.1.5 ts_resources.preview.columns (1)

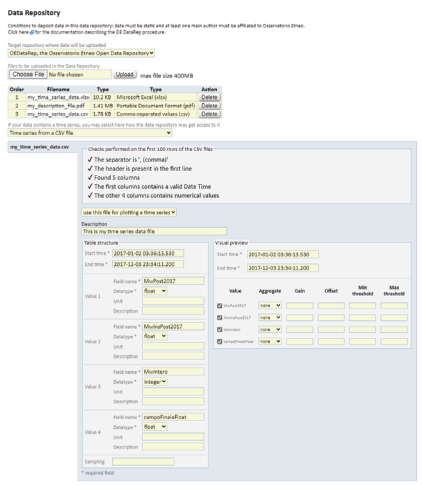
The objects in columns array, used to define the columns needed to the plot time-series preview in record landing page. It stores an object per each column in the time-series dataset to be plotted, the Table 6 shows columns attribute structure.

Field	Cardinality	Type	Description
`name`	1	`string`	Time-series column name
`aggregate`	0-1	`string`	Specific aggregate function used to plot time-series column values if any has been used. Could be one of: • `AVG: average` • `MIN: min` • `MAX: max` • `SUM: sum` • `COUNT: count` • `MEDIAN: median`
`offset`	0-1	`object`	Time-series preview offset value
`gain`	0-1	`numeric`	Time-series preview gain value
`minthreshold`	0-1	`numeric`	Time-series preview minimum threshold
`maxthreshold`	0-1	`numeric`	Time-series preview maximum threshold

Table 6 ts_resouces.preview.columns metadata attributes.

1.4 Changes to the INGV Metadata Editor

Switching to a new version of the OEDataRep required an update of the upload procedure implemented in the Metadata Editor associated with the institutional Data Registry.
In the previous version of the Metadata Editor and the DataRep repository, users were requested to upload their data into their personal Google Drive space, and publicly share them through a link, which was then inserted (copied) at the time of compiling the draft metadata record in the INGV Data Registry. Once the metadata record was successfully validated following the procedure defined in the INGV Data Policy [7], the Metadata Editor copies the data files from Google Drive to the data repository via APIs.
Thanks to the adoption of the newer InvenioRDM version as the underlying software for the new OEDataRep release, the new set of API related to the InvenioRDM record file management made a more robust and efficient integration possible. With the new release, the Metadata Editor can load data files uploaded by the user directly to the OEDataRep at the time of creating the draft record in the Data Registry, skipping the additional step of loading data into Google Drive first. The data files are stored in the OEDataRep, keeping only the metadata part in the Data Registry. The new procedure is certainly more straightforward, and only involves the use of the Metadata Editor and OEDataRep, exempting from using Google Drive space as an intermediary step.
In addition to a more direct file upload procedure, the web graphical user interface of the Metadata Editor had to be updated to consider the new metadata introduced to describe the presence of a time-series dataset. After selecting the OEDataRep as the target data repository to be used, and after uploading all their files, users may select a file to be used for loading a time-series dataset to the OEDataRep. A new specific web form is used to provide all parameters required for loading the time-series data file into the TSDSystem instance associated with OEDataRep (Figure 7). Once all parameters are provided, and the draft record is published to the OEDataRep data repository, the record landing page in the OEDataRep will show a preview of the time series (Figure 3).



Data Repository

Conditions to deposit data in this data repository: data must be static and at least one main author must be affiliated to Osservatorio Etneo.
Click here for the documentation describing the OE DataRep procedure.

Target repository where data will be uploaded
OEDataRep, the Osservatorio Etneo Open Data Repository

Files to be uploaded in the Data Repository
Choose File No file chosen Upload max file size 400MB

Order	Filename	Type	Type	Action
1	my_time_series_data.xlsx	10.2 KB	Microsoft Excel (xlsx)	Delete
2	my_description_file.pdf	1.41 MB	Portable Document Format (pdf)	Delete
3	my_time_series_data.csv	1.78 KB	Comma-separated values (csv)	Delete

If your data contains a time series, you may select here how the data repository may get access to it
Time series from a CSV file

my_time_series_data.csv

Checks performed on the first 100 rows of the CSV files
✓ The separator is ', (comma)'
✓ The header is present in the first line
✓ Found 5 columns
✓ The first columns contains a valid Date Time
✓ The other 4 columns contains numerical values

use this file for plotting a time series

Description
This is my time series data file

Table structure
Start time * 2017-01-02 03:36:13.530
End time * 2017-12-03 23:34:11.200

Value 1
Field name * MwPost2017
Datatype * float
Unit
Description

Value 2
Field name * MwInsPost2017
Datatype * float
Unit
Description

Value 3
Field name * MwIntero
Datatype * integer
Unit
Description

Value 4
Field name * campoFinaleFloat
Datatype * float
Unit
Description

Sampling

Visual preview
Start time * 2017-01-02 03:36:13.530
End time * 2017-12-03 23:34:11.200

Value	Aggregate	Gain	Offset	Min threshold	Max threshold
✓ MwPost2017	none				
✓ MwInsPost2017	none				
✓ MwIntero	none				
✓ campoFinaleFloat	none				

* required field

Figure 7 Screenshot of the "Data Repository" section in the Metadata Editor associated with the national INGV Data Registry. Example showing the web form describing a time series uploaded to OEDataRep.

2. The OEDataRep deployment strategies

As documented in chapter 1, the initial release of the Open Data Repository for the Osservatorio Etneo, has been redesigned for a new release - formally named OEDataRep - with the aim of improving its reliability, usability and software maintainability.

The redesign process involved both the components making up its software stack and the deployment strategy adopted to execute them. The term deployment strategy defines how to deliver software applications and in which environment (both hardware and software) these software execute.
This chapter details the evolution of the deployment strategy for the OEDataRep repository. In particular, the next subsection gives an overview of the hardware infrastructure located at the Osservatorio Etneo's computing data center. The subsequent sections elaborate the deployment strategies adopted to run the repository: starting from a brief overview of the initial prototype DataRep deployment to a detailed elaboration of the new OEDataRep deployment.

2.1 Hardware Infrastructure Level: The INGV-OE Data Center

All resources - both computational and data storage space - made available for the deployments described in this document are hosted in the data center located at the INGV-OE. However, despite being based on the same hardware, those allocated to the initial release of the DataRep repository reflected its prototypical nature, being generally lower than those of the newly redesigned release.
These resources are provided by an OE data center failover cluster[13] managed by the Microsoft Failover Clustering technology [40] [41] [42] to provide high availability (HA) of Hyper-V virtual machines (VMs) [43].
To enable the seamless migration of virtual machines between failover cluster nodes, a shared storage[14] infrastructure managed by the Failover Cluster service has been implemented. Physical storage is provided to the cluster by a SAN[15] through the iSCSI protocol [45].
Redesigning the entire repository was already scheduled in the "conclusion and future developments" chapter of the technical report of the DataRep initial release [Torrisi et al., 2022].
The DataRep repository prototype was hosted by a single High Availability (HA) Virtual Machine upon which the entire software stack was deployed.
Differently, OEDataRep repository is hosted on a set of HA Virtual Machines configured as clustered roles within the failover cluster [46], representing the host systems upon which to deploy the software stack services (see chapter 2). The OEDataRep clustered roles topology is made up of 5 virtual machines (VM):

- 2 stripped down HA VMs hosting the HAProxy load balancing service [47]. It serves as the entry point for public requests from the outside towards the services of the OEDataRep repository (the "web-api" and "web-ui" and "s3 object storage", see chapters 3.2.1, 3.2.2);
- 3 HA VMs hosting the whole OEDataRep software stack, distributed as docker containers and managed by the Docker Swarm orchestrator (see chapter 3.2).

All VMs are based on GNU Debian version 11.7 [48], boosted with Linux Kernel 5.10.179-1 [49]. Networking between the VMs is enabled by a dedicated VLAN declared on a virtual switch of the clustered infrastructure. All traffic between the containers is enabled to be isolated while

[13] Failover Cluster: A failover cluster is a set of computer servers that work together to provide either high availability (HA) [38] or continuous availability (CA) [39]. If one of the servers goes down, another node in the cluster can assume its workload with either minimum or no downtime through a process referred to as failover. Some failover clusters use physical servers only, whereas others involve virtual machines (VMs).

[14] Shared storage mechanisms usually store and consolidate files and related metadata information in a central resource that can be accessed (shared) among multiple users and systems simultaneously.

[15] SAN: Storage Area Network is a computer network which provides access to consolidated, block-level data storage. SANs are primarily used to access data storage devices, from servers so that the devices appear to the operating system as direct-attached storage [44].

reachability of the services from the outside is made possible by another DMZ VLAN configured on the HAProxy VMs.

2.2 Application services level: OEDataRep deployment

Looking at the InvenioRDM internals, the application adopts a microservices pattern: a set of independently deployable, loosely coupled, components that work together, communicating with each other at application level in a deployment environment.
Very often this kind of environment coincides with a software platform belonging to the class of virtualization known as "Operating System Virtualization" [Soltesz et al., 2007], capable of delivering software in packages called containers. Nowadays, among the most used platforms belonging to this category there is Docker [50].
In the initial prototype release, all of the DataRep application's services (databases, message-queues, caches, web-service APIs, etc.) were documented and configured through a single Docker Compose [52] YAML file[16] and deployed on a single VM host satisfying the multi-container nature of the application, as shown in (Figure 8).

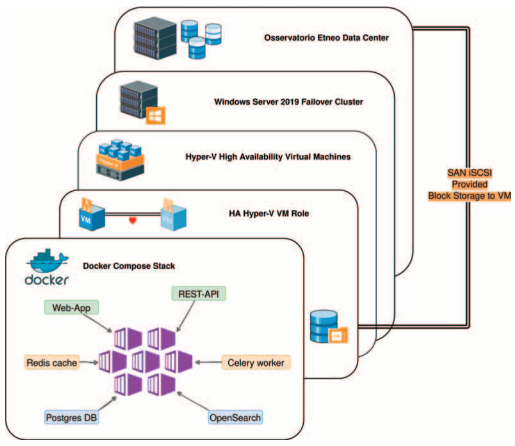

Osservatorio Etneo Data Center
Windows Server 2019 Failover Cluster
Hyper-V High Availability Virtual Machines
HA Hyper-V VM Role
VM
SAN iSCSI Provided Block Storage to VM
docker
Docker Compose Stack
Web-App
REST-API
Redis cache
Celery worker
Postgres DB
OpenSearch

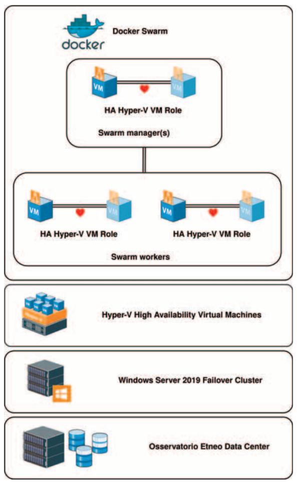
Figure 8 Docker Compose graphical representation of the deployment for the DataRep prototype. The storage identified by a block device provided by the SAN (through iSCSI protocol) is tied to the Virtual Machine that hosts all the Docker containers services stack.

The main benefit of using Compose is that, by using a single command, it is possible to create and start all services from aforementioned configurations, as well as the management of the whole application lifecycle. Thanks to these capabilities, Compose is cataloged as a container orchestrator, although it has traditionally been focused on development and testing workflows, due to its single-host deployment focus [53].
In order to achieve a production-grade deployment for the OEDataRep release, both hardware/virtualized resources (see chapter 3.1) and deployment tools/strategies have been improved. Concerning this last point, the first substantial difference regards the deployment tool: all of the InvenioRDM application software services (see chapter 2) are provided as Docker containers, and the cooperation between all these microservices is managed by the Docker Swarm tool [54], which belongs to the suite of components of the Docker platform (PaaS[17]).
As simply explained on the Docker documentation web page, Docker Swarm mode is an

[16] YAML is a human-readable data-serialization language. It is commonly used for configuration files and in applications where data is being stored or transmitted [53].

[17] PaaS: Docker is a set of platforms as a service (PaaS) products that use OS-level virtualization to deliver software in packages called containers. Platform as a service (PaaS) or application platform as a service (aPaaS) is a category of cloud computing services that allows to provide, instantiate, run, and manage a modular bundle of applications, without the complexity of building and maintaining the infrastructure [55].

advanced feature for managing a “cluster of Docker daemons” (engines) used in production runtime environments. A cluster of Docker engines is called a swarm.
More in general, a swarm consists of multiple Docker hosts which run in swarm mode and act as managers (to manage membership and delegation) and workers (which run swarm services) [56].
In the deployment design conceived for OEDataRep, the swarm is identified by the Docker engines, each one executing on the set of HA VMs mentioned in the previous chapter (see chapter 3.1). Each VM and its Docker engine act as a swarm node, and in particular: one VM identifies the Swarm Manager node while the remaining two VMs are worker nodes (Figure 9).



docker
Docker Swarm
VM
HA Hyper-V VM Role
Swarm manager(s)
HA Hyper-V VM Role
HA Hyper-V VM Role
Swarm workers
Hyper-V High Availability Virtual Machines
Windows Server 2019 Failover Cluster
Osservatorio Etneo Data Center

Figure 9 The production environment deployment stack: at the lowest level is the Osservatorio Etneo Data Center hardware, mainly composed of the Computing resources and Storage resources (which is provided by the SAN). All the computing resources are managed the Windows Server 2019 Failover Cluster, which also manages and instantiates Hyper-V virtual machines. At the top level, there is a Docker Swarm cluster based on several HA VMs.

The manager node dispatches units of work - called “tasks” - (representing the OEDataRep software stack components), to worker nodes according to the deployment policies documented in the swarm YAML file. From this the manager node reads the desired state of the swarm and performs the orchestration and cluster management functions required to achieve and maintain it (in the OEDataRep deployment scenario, the manager also acts as worker node).
The main intrinsic features made available by the Docker Swarm tool have been exploited for the deployment design of the new OEDataRep release. In particular, with an initial set of 3 VMs dedicated to the swarm, it was possible to obtain a high degree of load balancing of several services through the native feature integrated into the Docker Swarm. The Swarm Manager uses ingress load balancing to expose the services externally to the swarm (both the WEB and REST API interface). As previously described, an HAProxy load balancer acts as an external proxy, with the aim of forwarding requests from the outside world to the ingress load balancing service on the manager node which in turn routes the requests to the appropriate service containers distributed across the swarm nodes (Figure 10).

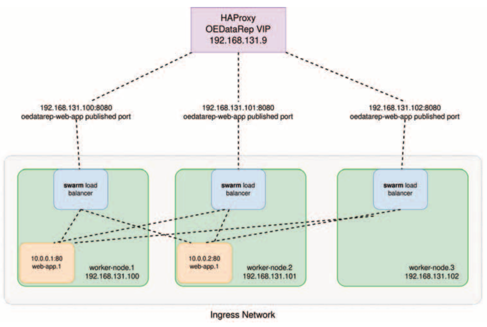


Figure 10 Graphical representation of how an SDN network infrastructure within the Swarm deployment is implemented: An HAProxy with a VirtualIP is the gateway for all requests that in turn will be redirected to the horizontally scaled instances of worker-nodes (hosted by Virtual Machines composing the Swarm cluster). Each worker-node will implement a swarm load balancer to manage requests across the services instances.

The OEDataRep swarm topology includes three worker nodes; in order to better distribute the software services across them, they were labeled according to the type of services hosted as Frontend and Backend nodes (or Tiers).
Among the software components making up the OEDataRep platform stack, some services strictly related to the role of web application, could be categorized as stateless, meaning that they fulfill the role of processing requests (coming both from the WEB or from the REST API interface), without maintaining a state of such requests, other than the response state of the same (successful or not). Due to this nature, these services were able to take advantage of the ability of the Docker Swarm tool to easily horizontally scale [57] themselves (up and down) on the available worker nodes. These services belong to the Frontend-tier, opposed to the Backend-tier which represents the group of services that are categorized as stateful and whose main role is to maintain states.
Following this logical distinction, it was possible to fully leverage the Swarm declarative service model approach, in order to define the desired state of the services in the application stack: on which nodes deploy a group of services and how many resources are assigned to them.
The native Multi-host networking mechanism implemented by the Docker Swarm has been the way through which OEDataRep services discover themselves; the Swarm Manager automatically assigns IP addresses to the containers (services) on the overlay[18] network when it initializes or updates the application. Furthermore, a native service discovery mechanism is implemented by the Swarm Manager: a unique DNS name is assigned to each service in the swarm. A DNS server[19] embedded in the swarm is in charge of distributing (load balancing) network traffic among swarm containers.
To summarize therefore, the final design of the OEDataRep deployment foresees: the declaration of a Docker Swarm composed of at least 3 nodes (VMs acting as hosts for the services); one of those nodes will cover the role of Swarm Manager (as well as worker), in charge of managing the execution of the tasks declared in the services.
Two logical groups, called the Frontend-tier and the Backend-tier, serve to divide the swarm nodes based on the type of service they will host, be it stateless or stateful.
Furthermore the VM, or swarm node that is labeled as the Backend-tier will be the one in charge of managing the storage space provisioned by the SAN. This latter will manage a highly reliable block-disk and will provide it to the Backend-tier labeled VM through an iSCSI protocol; it will be the task of this VM to drive the access to this block device at a software level by the "horizontally scaled" services of the OEDataRep stack.
The Swarm Manager node pursues the so-called "Desired state reconciliation" policy [58]: it constantly monitors the cluster state and reconciles any differences between the actual state and the expressed desired state. For example, in the case of OEDataRep, the services belonging to the Frontend-tier, e.g. both the web related and REST API interface services, at their first instantiation are configured to run in 3 containers replicas; the Swarm Manager assigns the replicas to workers that are running and available.
The services of the stack communicate with each other through an overlay network defined and managed by the Docker Swarm, while the services that must be exposed to the outside are managed by the internal load balancer of Docker Swarm. Finally, a High Available (HA) HAProxy service is used to route requests from the outside towards the Swarm load-balancing internal service and then to the stack services themselves.

[18] An overlay network is a computer network that is layered on top of another network. In this case it is the network associated with the Docker Swarm cluster, where each microservice container runs.

[19] The Domain Name System is a hierarchical and distributed naming system for computers, services, and other resources on the Internet or other Internet Protocol networks: it is a naming database in which internet domain names are located and translated into Internet Protocol (IP) addresses.

In the next subsections are detailed the components of the OEDataRep software stack, logically divided into Frontend and Backend tiers. A blueprint service diagram (Figure 12) summarizes all the interactions of the microservices deployed through containers within a Docker swarm.

2.2.1 OEDataRep application services: Frontend-tier

Swarm services belonging to the Frontend-tier are executed by the Swarm Manager on the nodes labeled in the same way within the cluster.
They represent those services that do not need to maintain a state over the execution flow, namely stateless applications.
All of them work together to allow OEDataRep accomplish its role as a public Open Data Repository, freely accessible and searchable by the community.
First two services described here are those providing both public WEB and REST API interfaces to the repository contents and resources, respectively. These two services are identified by the container names "web-ui" and "web-api" within the Docker Swarm stack. They are both based on the same Docker custom image[20] (hosted on a private Docker registry): a prebuilt image which includes all of the InvenioRDM frameworks needed libraries and the configuration files to properly start the services. Furthermore, these services are based on the execution of web application business-logic, based on the Python Flask microframework [59] and served by the uWSGI application microserver [60]; but, while the first one ("web-ui") serves the web application itself, the latter ("web-api") instead implements the access to the REST API interface, needed by internal services.
Example use cases of the exposed REST API resources are:

a. The Metadata Editor client interface that drives the publication flow of a record;
b. The custom TSDSystem integration module, oedatarep-timeseries-loader, which uses a subset of APIs to accomplish the time-series retrieval from the record associated files and PUT them to TSDSystem.

The other main stack's services included in the Frontend-tier are those named "worker" and "tsd-app". The first one is a core component of the InvenioRDM distribution, which implements a Celery worker node needed to execute scheduled jobs within the application logic. As previously detailed (see chapter 2), one of the custom jobs executed by the celery worker in OEDataRep is the one in charge of communicating with the TSDSystem REST APIs. These APIs are those made available by the tsd-app Docker service container, which contains and executes the core business logic implemented by the TSDSystem application (see chapter 2).
The other two services included in the Frontend-tier are the "kibana" web-application visualizer[21] and the Nginx "frontend" server to manage ssl certificates.

[20] InvenioRDM Docker images: https://inveniordm.docs.cern.ch/maintenance/docker-images/, Docker base image GitHub repository: https://github.com/inveniosoftware/docker-invenio

[21] Kibana is a dashboard software for data visualization used in an Elasticsearch (Opensearch) environment.

Swarm service	Docker image	Software	Initial scale	Description
web-ui	invenioRDM	Flask + uwsgi + invenioRDM	3	Web application
web-api	invenioRDM	Flask + uwsgi + invenioRDM	3	REST API resources
tsd-app	tsdsystem-app	php	3	REST API resources interface for TSD DB
worker	celery	celery	3	Asynchronous job queues
kibana	kibana	kibana	3	Data visualization dashboard software for Elasticsearch
frontend	nginx	nginx	1	Forwards web browser or REST API requests to web servers

Table 7 All microservices belonging to the Frontend-tier, described by the name the service will have for the deployment, the image name used, the software included in the instance container and the initial horizontal scale for the service.

2.2.2 OEDataRep application services: Backend-tier

These kinds of services are those intended to run on the Docker Swarm cluster nodes labeled as Backend-tier. They conceptually include all the services which maintain a **state** (stateful) and manage data.
As a standard InvenioRDM application/distribution, OEDataRep includes a database service based on a PostgreSQL Docker container image. In the specific case of OEDataRep, the instantiation of this container within the swarm has been customized in order to also host the TSDSystem time-series data. In particular, the instance is based on a Docker image that allows PostgreSQL database to enable the TimescaleDB extension (a set of functionalities aimed at optimizing the management of temporal data, time-series); this way enables having a single database deployment, hosting both the TSDSystem and OEDataRep schemas.
During a publishing process, data is uploaded to the repository by the Metadata Editor web interface, through the REST APIs exposed by the "web-api" containers (see chapter 3.2.1).
These data, which represent the core business of the repository, need to be stored in a secure and resilient way. This was the point of reference that drove the storage infrastructure design for the OEDataRep deployment. The raw storage space is provided by a SAN (see chapter 3.1), which specifically reserves a LUN space[22] for the repository's data.
Due to the nature of the deployment depicted which predicts the possibility to horizontally scale the containers belonging to the stateless Frontend-tier, storage needs to be accessible (reliable) to containers spread on several swarm worker nodes. To accomplish this constraint, the OEDataRep upload process has been configured to store the data not on local storage (as by default in InvenioRDM) but to an Object Storage backend [61], exploiting the InvenioRDM native feature that allows to interact with an S3 standard API [62]. For this purpose, specifically for the OEDataRep deployment, an S3 compliant Object Storage system has been set up in

[22] LUN: a logical unit number (LUN) is a slice or portion of a configured set of disks that is presentable to a host and mounted as a volume within the OS.

order to represent the distributed access point for the repository's data. It consists of a container running the MinIO high-performance, S3 compatible object store software [63].
A service for the MinIO Docker instance deployment is constrained upon the single swarm node labeled as Backend-tier and which relies on the VM that has been provided with the disk storage by the SAN (Figure 11).

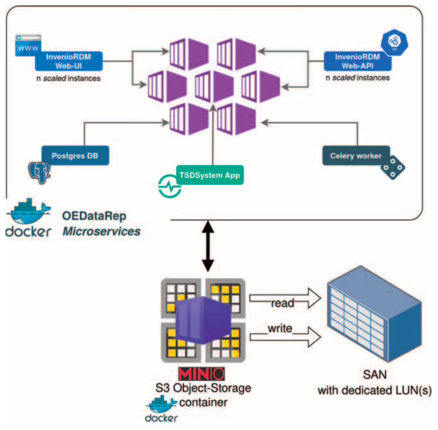


Figure 11 Graphical representation of storage implementation for the OEDataRep deployment: all OEDataRep microservices, deployed by Docker containers through Docker Swarm, will communicate with the MinIO Docker container (deployed on the Backend-tier Swarm node) that has access to the dedicated lock device provided by the SAN.

The other two main services that compose the Backend-tier are those running the ElasticSearch and the RabbitMQ software. The first is mostly used for indexing any text related information belonging to the records and the metadata associated, while the latter allows communication between the microservices at application level.
Another service hosting Redis software is used for caching purposes.

Swarm service	Software	Description
db	PostgreSQL +(TimescaleDB + PostGIS) extensions	Main database instance for the OEDataRep deployment; including both repository DB and TSDSystem time-series DB
minio	MinIO high-performance, S3 compatible object store	Backend S3 object-storage for the repository data, accessible from every frontend container instance
es	Elasticsearch suite for search, analytics, and observability	Implements the Repository system of indexing metadata
mq	RabbitMQ open-source message-broker software	Implements the way repository's microservices communicate with each other
cache	Redis in-memory data structure store	In memory cache for some relevant internal components of the repository
pgadmin	PGAdmin a PostgreSQL management web UI	Service to facilitate underlying database management operation

Table 8 All microservices belonging to the Backend-tier, described by the name the service will have for the Swarm deployment, the software included in the instance container and a description of the service.

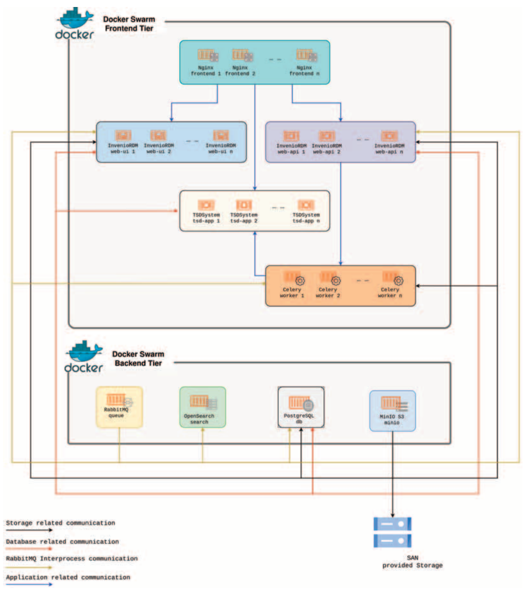


Figure 12 Scheme representation for several intercommunication types of all the microservices within the OEDataRep Docker Swarm deployment. All the services belonging to the Frontend-tier could be scaled horizontally in n-instances, while those belonging to the Backend-tier are single instance containers. Swarm will check the healthy status of each container and restart in case of failure. The Storage provided by the SAN is a block device associated with the Backend Virtual Machine and accessed by the MinIO S3 container instance. All the services will communicate with it to physically store data on the SAN.

3. Conclusions

This technical report illustrates the evolution of the second-generation INGV-OE data repository following the Open Science paradigm, and particularly, adopting the FAIR data principles. The new version is fully exploited by an updated version of the national INGV Metadata Editor, which significantly simplifies the upload procedure of the data.
The evolution was undertaken with the primary aim of strengthening the fundamental requirements of reliability and resilience for the stored data. At the same time, a comprehensive scalability strategy was implemented, leveraging the capabilities of the production-grade hardware infrastructure available at the INGV-OE Data Center.
All the code customization and deployment receipts can be found in the public GitHub repository at: https://github.com/ingv-oe-dev.
Furthermore, researchers and technologists belonging to the INGV-OE may declare in their scientific articles that their data is published in OEDataRep, citing it through its DOI (http://doi.org/10.17616/R31NJNEL). In fact, OEDataRep has been declared in the re3data registry [Strecker, 2023], the global registry that covers Research Data Repositories (RDR), and it has been described using their metadata schema [re3data, 2023].
OEDataRep is also listed in the Sherpa/OpenDOAR [64] public directory, the quality-assured, global Directory of Open Access Repositories (https://v2.sherpa.ac.uk/id/repository/10759), recognizing the conformity to an up-to-date Open Science practice. It is listed also in FAIRsharing.org (https://doi.org/10.25504/FAIRsharing.bd916e), a public curated, informative and educational resource on data and metadata standards.
In conclusion, the second-generation OEDataRep was successfully adopted as the official data repository of INGV-OE for the past 10 months. Currently, it hosts 24 datasets, for a total amount of approximately 6 GB of data. Being the first and only generalistic, non field-specific data repository operating at INGV, it is being used as a pioneering experiment and closely monitored in order to evaluate its potential. If successful, its use may be extended to users affiliated with other INGV sections, or, alternatively, new customized instances may be established in other INGV sections and, possibly, federated under a unique umbrella.

Authors contribution

As author and main developer of the TSDSystem framework, C.C. worked on the integration with the OEDataRep repository with the support of F.P and M.T.. F.P. and M.T. have redesigned the TSDSystem with a microservices architecture in mind. F.P. and M.T. worked on the deployment design of the whole system on the Docker Swarm platform, integrating OEDataRep and TSDSystem into container microservices and paying particular attention to the storage infrastructure. M.T. worked out almost all of the technical details of OEDataRep addons development with support of F.P. As main developer of the Metadata Editor, M.L. was in charge of updating it to interact with the new version of the OEDataRep APIs. C.C., M.T., M.L., F.P. and P.M. designed the metadata details for time-series resources. P.M. conceived the original idea and supervised the project. F.P. took the lead in writing the manuscript and designing figures. All authors provided critical feedback and helped shape the research, analysis and manuscript.

References

Caffaro J., Kaplun S. (2010). *Invenio: A Modern Digital Library for Grey Literature*. CERN-OPEN-2010-027. https://cds.cern.ch/record/1312678

Cassisi C., Montalto P., Aliotta M., Cannata A., Prestifilippo M. (2015). *TSDSystem: un database multidisciplinare per la gestione di serie temporali*. Rapp. Tec. INGV, 304: 34, https://doi.org/10.13127/RPT/304

DataCite Metadata Working Group (2021). *DataCite Metadata Schema for the Publication and Citation of Research Data and Other Research Outputs*. Version 4.4. DataCite e.V. https://doi.org/10.14454/fxws-0523

INGV Data Management Office (2020). *INGV Open Data Registry, the metadata catalogue of the Istituto Nazionale di Geofisica e Vulcanologia*. Istituto Nazionale di Geofisica e Vulcanologia (INGV). https://doi.org/10.13127/data-registry

Jaramillo D., Nguyen D.V., Smart R. (2016). *Leveraging microservices architecture by using Docker technology*. SoutheastCon 2016, Norfolk, VA, USA, 2016, pp. 1-5, https://doi.org/10.1109/SECON.2016.7506647

Morell T., Holmes K. (2019). *InvenioRDM: A Collaborative Next-Generation Research Data Management and Repository Solution*. Zenodo. https://doi.org/10.5281/ZENODO.3570807

Newman S. (2021). *Building microservices*. "O'Reilly Media, Inc."

Re3data (2023). *re3data Metadata Schema 4.0 XML Schema (4.0)*. https://doi.org/10.48440/RE3.015

Robinson N., Le Meur J-Y., Simko T. (2007). *Managing an Institutional Repository with CDS Invenio*. CERN-IT-Note-2007-008. https://cds.cern.ch/record/1055312

Rodrigues E., Bollini A., Cabezas A., Castelli D., Carr L., Chan L., Humphrey C., Johnson R., Knoth, P., Manghi P., Matizirofa L., Perakakis P., Schirrwagen J., Selematsela D., Shearer K., Walk P., Wilcox D., Yamaji K. (2017). *Next Generation Repositories: Behaviours And Technical Recommendations Of The Coar Next Generation Repositories Working Group*. Zenodo. https://doi.org/10.5281/ZENODO.1215014

Sheoran A., Fahmy S., Sharma P., Modi N. (2021). *Invenio: Communication Affinity Computation for Low-Latency Microservices*. Proceedings of the Symposium on Architectures for Networking and Communications Systems. https://doi.org/10.1145/3493425.3502750

Soltesz S., Pötzl H., Fiuczynski M.E., Bavier A. Peterson L. (2007). *Container-based operating system virtualization: a scalable, high-performance alternative to hypervisors*. SIGOPS Oper. Syst. Rev. 41, 3 (June 2007), 275–287. https://doi.org/10.1145/1272998.1273025

Strecker, D., Axtmann A., Bertelmann R., Cousijn H., Elger K., Ferguson L.M., Fichtmüller D., Jones

C., Lindenmann I., Neidiger C., Nguyen T.B., Pal J.K., Pampel H., Petras V., Schnepf E., Semrau A., Ulrich R., Upmeier A., Vierkant P., … Wright S.J. (2023). *Metadata Schema for the Description of Research Data Repositories : version 4.0*. Re3data. https://doi.org/10.48440/RE3.014

Torrisi M., Pistagna F., D'Agostino M., Locati M. e Montalto P., (2022). *Prototipo di un portale per la distribuzione dei dati dell'Osservatorio Etneo basato sulla piattaforma Zenodo*. Rapp. Tec. INGV, 457: 126, https://doi.org/10.13127/rpt/457

Wilkinson M.D., Dumontier M., Aalbersberg I.J., et al., (2016). *The FAIR Guiding Principles for scientific data management and stewardship*. Scientific Data, 3(1): 160018. https://doi.org/10.1038/SDATA.2016.18

Sitography

[1] Open data - Wikipedia: 2011. https://en.wikipedia.org/wiki/Open_data

[2] "Understanding open science". *unesdoc.unesco.org*. UNESCO.org. 2022. Retrieved 7 April 2023., https://unesdoc.unesco.org/ark:/48223/pf0000383323

[3] Software deployment - Wikipedia: 2008. https://en.wikipedia.org/wiki/Software_deployment

[4] List of field-specific data repository managed by INGV https://www.ingv.it/risorse-e-servizi/archivi-e-banche-dati

[5] Data INGV, ingv.it, https://data.ingv.it

[6] DataCite Metadata Schema 4.4: https://support.datacite.org/docs/datacite-metadata-schema-44

[7] https://data.ingv.it/docs/implementation/index.html#procedure-for-entering-elements

[8] GitHub - ingv-oe-dev/tsdsystem: https://github.com/ingv-oe-dev/tsdsystem.

[9] What is time-series Data? | Definition, Examples, Types & Uses: 2023. https://www.influxdata.com/what-is-time-series-data/

[10] "Microservices from theory to practices", *IBM Redbooks*, [online] Available: http://www.redbooks.ibm.com/redbooks/pdfs/sg248275.pdf

[11] Microservices, Martin Fowler, https://martinfowler.com/articles/microservices.html

[12] Zenodo - Research. Shared.: https://www.zenodo.org/

[13] InvenioFramework — inveniosoftware.org, https://invenio-software.org/products/framework/

[14] InvenioRDM "Turn-key research data management repository", https://inveniordm.docs.cern.ch/

[15] Governance — inveniosoftware.org, https://inveniosoftware.org/governance/

[16] GitHub - ingv-oe-dev/oedatarep-ts-loader, https://github.com/ingv-oe-dev/oedatarep-ts-loader

[17] What is Infrastructure as Code (IaC)?, *Redhat.com*, https://www.redhat.com/en/topics/automation/what-is-infrastructure-as-code-iac

[18] Timescale is PostgreSQL++ for time-series and event data, https://www.timescale.com/

[19] GitHub - timescale/timescaledb: An open-source time-series SQL database optimized for fast ingest and complex queries. Packaged as a PostgreSQL extension. https://github.com/timescale/timescaledb

[20] PostgreSQL Database, https://www.postgresql.org/

[20] Microservices vs. monolithic architecture | Atlassian: https://www.atlassian.com/microservices/microservices-architecture/microservices-vs-monolith

[21] What is a Container? | Docker: https://www.docker.com/resources/what-container/

[22] Welcome! - The Apache HTTP Server Project: https://httpd.apache.org/

[23] NGINX Reverse Proxy | NGINX Documentation: https://docs.nginx.com/nginx/admin-guide/web-server/reverse-proxy/

[24] pgAdmin - PostgreSQL Tools: https://www.pgadmin.org/

[25] Grafana, GitHub - grafana/grafana: The open and composable observability and data

visualization platform. Visualize metrics, logs, and traces from multiple sources like Prometheus, Loki, Elasticsearch, InfluxDB, Postgres and many more. https://github.com/grafana/grafana

[26] Building services - Turn-key research data management repository: https://inveniordm.docs.cern.ch/develop/topics/service/

[27] Software - Turn-key research data management repository: https://inveniordm.docs.cern.ch/develop/architecture/software/#service-layer

[28] Introduction to Celery, What is a task queue, https://docs.celeryq.dev/en/stable/getting-started/introduction.html#what-s-a-task-queue

[29] Asynchronous Tasks with Django and Celery - Real Python, https://realpython.com/asynchronous-tasks-with-django-and-celery/

[30] Elasticsearch: What is it and why it's important - AWS: https://aws.amazon.com/what-is/elasticsearch/

[31] Overview REST API - Turn-key research data management repository: https://inveniordm.docs.cern.ch/reference/rest_api_index/

[32] InvenioRDM Overview Metadata - Turn-key research data management repository: https://inveniordm.docs.cern.ch/reference/metadata/

[33] GitHub - inveniosoftware/invenio-rdm-records: DataCite-based data model for InvenioRDM flavour.: 2023. https://github.com/inveniosoftware/invenio-rdm-records

[34] Github.com ingv-oe-dev/invenio-rdm-records: https://github.com/ingv-oe-dev/invenio-rdm-records/blob/v0.2/invenio_rdm_records/records/jsonschemas/records/record-v5.0.0.json

[35] OpenSearch, https://opensearch.org/

[36] JSON - Introducing JSON, *json.org*, https://www.json.org/json-en.html

[37] GitHub - ingv-oe-dev/invenio-rdm-records: invenio-rdm-records/invenio_rdm_records/records/mappings/v7/rdmrecords/records/record-v5.0.0.json at v0.2: https://github.com/ingv-oe-dev/invenio-rdm-records/blob/v0.2/invenio_rdm_records/records/mappings/v7/rdmrecords/records/record-v5.0.0.json

[38] What is High Availability? - Cisco, https://www.cisco.com/c/en/us/solutions/hybrid-work/what-is-high-availability.html#~infrastructure-elements

[39] Continuous availability - Wikipedia: https://en.wikipedia.org/wiki/Continuous_availability

[40] Failover Clustering: 2022. https://learn.microsoft.com/en-us/windows-server/failover-clustering/failover-clustering-overview

[41] How to set up and manage a Hyper-V Failover Cluster, Step by step: https://www.altaro.com/hyper-v/failover-cluster-manager/

[42] What is Failover Clustering? | ServerWatch: 2019. https://www.serverwatch.com/servers/failover-cluster/

[43] Hyper-V Technology Overview: 2021. https://learn.microsoft.com/en-us/windows-server/virtualization/hyper-v/hyper-v-technology-overview

[44] What Is a Storage Area Network or SAN? - SAN vs. NAS | NetApp: https://www.netapp.com/data-storage/what-is-san-storage-area-network/

[45] iSCSI protocol, *wireshark.org*, https://wiki.wireshark.org/iSCSI

[46] Hyper-V Failover Cluster Setup: A Step-by-Step Guide: 2022. https://www.nakivo.com/blog/hyper-v-cluster-setup/

[47] An Introduction to HAProxy and Load Balancing Concepts | DigitalOcean: 2017. https://www.digitalocean.com/community/tutorials/an-introduction-to-haproxy-and-load-balancing-concepts

[48] Debian — News — Updated Debian 11: 11.7 released: https://www.debian.org/News/2023/20230429

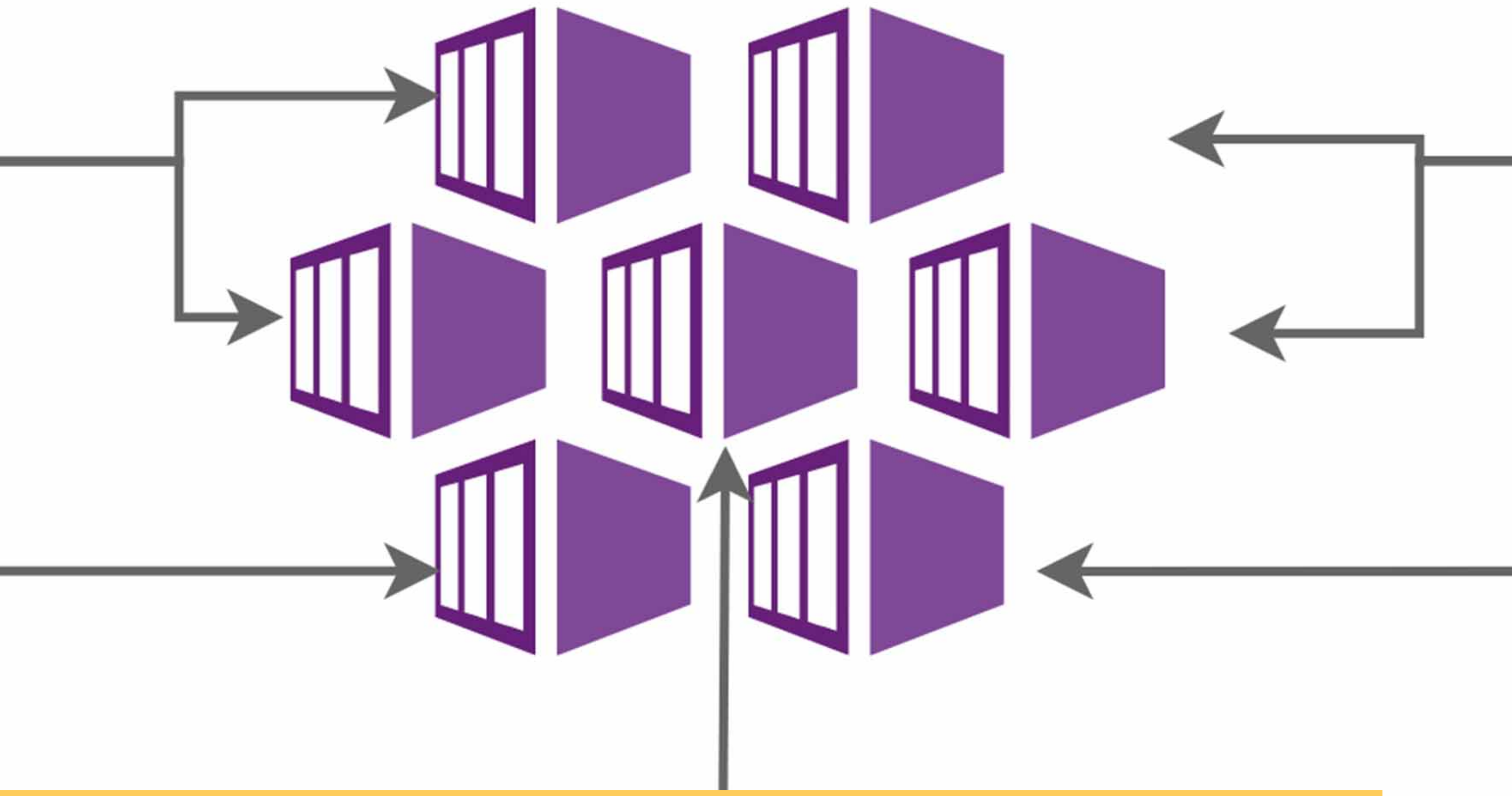
[49] kernel/git/stable/linux.git - Linux kernel stable tree:

https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux.git/diff/?id=v5.10.179&id2=v5.10.178&dt=2
[50] Docker overview: 2023. https://docs.docker.com/get-started/overview/
[51] Docker Compose overview: 2023. https://docs.docker.com/compose/
[52] YAML - Wikipedia: 2001. https://en.wikipedia.org/wiki/YAML
[53] Key features and use cases of Docker Compose: 2023. https://docs.docker.com/compose/features-uses/
[54] Swarm mode overview: 2023. https://docs.docker.com/engine/swarm/
[55] Platform as a service - Wikipedia: 2011. https://en.wikipedia.org/wiki/Platform_as_a_service
[56] Swarm mode key concepts: 2023. https://docs.docker.com/engine/swarm/key-concepts/
[57] Horizontal scaling - AWS Well-Architected Framework: https://wa.aws.amazon.com/wat.concept.horizontal-scaling.en.html
[58] Swarm mode overview: 2023. https://docs.docker.com/engine/swarm/
[59] Getting Started with Flask, a Python Microframework — SitePoint: https://www.sitepoint.com/flask-introduction/
[60] How To Serve Flask Applications with uWSGI and Nginx on Ubuntu 20.04 | DigitalOcean: 2020. https://www.digitalocean.com/community/tutorials/how-to-serve-flask-applications-with-uwsgi-and-nginx-on-ubuntu-20-04
[61] What is Object Storage? - Object Storage Explained - AWS: https://aws.amazon.com/what-is/object-storage/
[62] Amazon S3 REST API Introduction - Amazon Simple Storage Service: https://docs.aws.amazon.com/AmazonS3/latest/API/Welcome.html
[63] MinIO | High Performance, Kubernetes Native Object Storage: https://min.io
[64] Welcome to OpenDOAR - Sherpa Services: https://v2.sherpa.ac.uk/opendoar/

Coordinamento editoriale
Francesca DI STEFANO
Istituto Nazionale di Geofisica e Vulcanologia

Progetto grafico
Barbara ANGIONI
Istituto Nazionale di Geofisica e Vulcanologia

Impaginazione
Barbara ANGIONI
Patrizia PANTANI
Massimiliano CASCONE
Istituto Nazionale di Geofisica e Vulcanologia



ISTITUTO NAZIONALE DI GEOFISICA E VULCANOLOGIA