RE4DY

MANUFACTURING DATA NETWORKS

RE4DY TOOLKIT

Name of the Tool	F-UJI (FAIRsFAIR Research Data Object Assessment Service)
Tool Owner	Industry Commons Foundation
Version	1.0
Date	Nov 2025
Version	V1.0

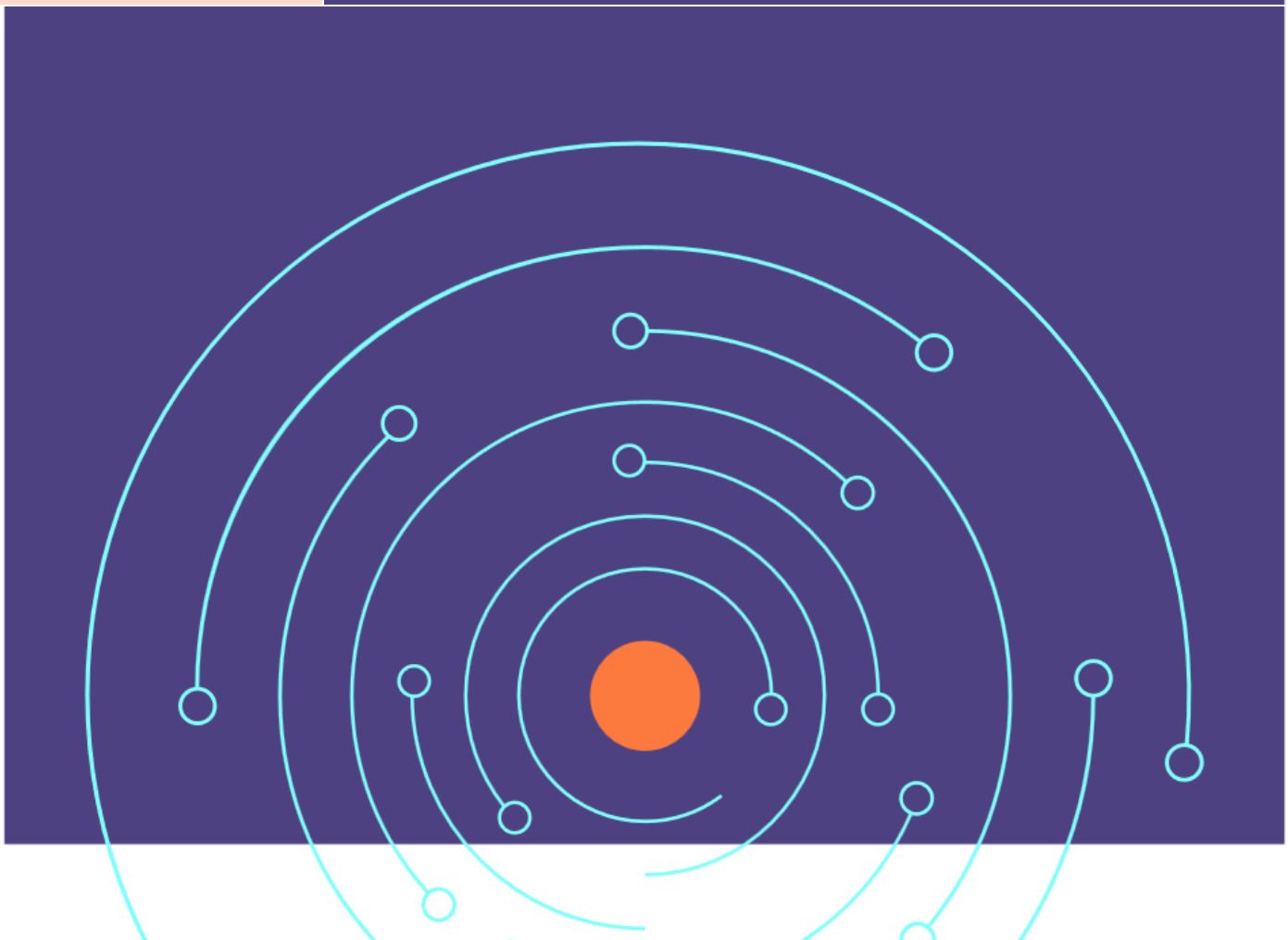

Table of contents

1. Component Description

F-UJI is a tool for data FAIRness assessment which provides the ability to specify a data set identifier (e.g. DOI, URL) in order to generate an evaluation based on the seventeen FAIRness principles released by GO FAIR. Optionally, a metadata service (OAI-PMH, SPARQL, CSW) endpoint URI may be provided as input to identify additional information about the data set. The assessment is automated, leveraging the metadata included with or embedded in the data set and providing a score as well as pass or fail statements for each metric. In the context of RE4DY, the tool will be utilized to programmatically assess the FAIRness of data sets as a means of compliance check and assurance. The functionality offered by F-UJI will be integrated with other tools and components.

2. Input

The F-UJI tool provides its FAIRness assessment services by implementing a REST API. Regarding the evaluation, the automated F-UJI tool requires an identifier of a data set, such as a DOI or a URL as input. It also provides an endpoint that accepts a metadata service (OAI-PMH, SPARQL, CSW) endpoint URI to fetch additional metadata of the data set to be assessed. The input may be provided either by an individual or another service in the RE4DY platform. Moreover, apart from the Python implementation of the tool, F-UJI has been integrated into a web demo which provides the evaluation output in a more human-friendly format.

3. Output

Based on the metadata extracted from the dataset, the FAIRness assessment results are returned as a JSON object and include pass or fail statements and the score earned for each FAIRness metric.

4. Information Flow

In order to evaluate the FAIRness of a dataset, it is possible to directly call the REST API offered by F-UJI using the URL of the dataset to be evaluated, as shown in the diagram below. The procedure is identical for the rest of the API endpoints offered by F-UJI.

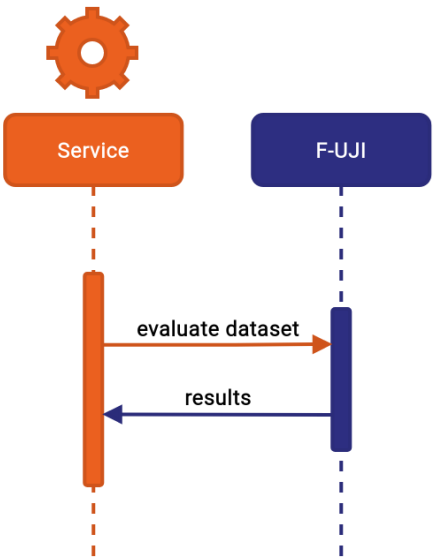

Figura 1 UML Diagram of the dataset evaluation process and the communication of a client service with the F-UJI component.

5. Internal Architecture

The F-UJI component utilizes a number of Python libraries to provide its functionality. Particularly, it uses `rdflib` and `urllib` to access network resources and `extruct`, `tika` and `lxml` to extract metadata from text of various file types. Finally, it uses `Flask` to implement and expose the API endpoints required for dataset evaluation and metadata harvesting.

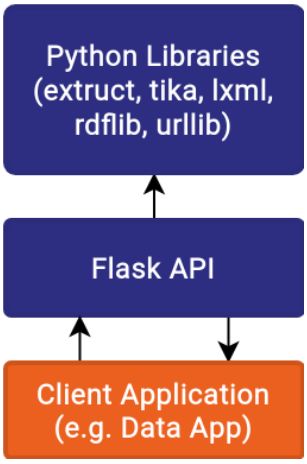

Figura 2 Internal architecture of the F-UJI component and related libraries and services

6. API

`POST /evaluate`

- **Input**

```
{
    "metadata_service_endpoint":
"http://ws.pangaea.de/oai/provider",
    "metadata_service_type": "oai_pmh",
    "metric_version": "metrics_v0.5",
    "object_identifier":
"https://doi.org/10.1594/PANGAEA.908011",
    "test_debug": true,
    "use_datacite": true
}
```

- **Output**

Some fields are truncated or omitted for brevity.

```
{
    "metric_specification":
"https://doi.org/10.5281/zenodo.6461229",
    "metric_version": "metrics_v0.5",
    "resolved_url":
"https://doi.pangaea.de/10.1594/PANGAEA.908011",
    "results": [
        {
            "id": 3,
            "maturity": 3,
            "metric_identifier": "FsF-F2-01M",
            "metric_name": "Metadata includes
descriptive...",
            "metric_tests": {
                "FsF-F2-01M-1": {
                    "metric_test_maturity": 1,
                    "metric_test_name": "Metadata has
been...",
                    "metric_test_score": 0.5,
                    "metric_test_status": "pass"
                },
                // ...
            },
            "score": {
                "earned": 2,
                "total": 2
            },
```

```
                "test_status": "pass"
            },
        ],
        "summary": {
            // ...
        },
        // ...
    }
```

- **Description**
  This endpoint evaluates the FAIRness of a data object. It outputs a JSON object which includes metadata about the dataset, the results of the assessment with scores and pass or fail statements for each metric, and finally a summary of the results. The endpoint supports basic authentication which requires a username and password to be provided. Users and their credentials are managed through a configuration file which is read by the tool upon request. It returns 400 if an invalid data object identifier is provided and 401 if the authentication information passed with the request headers is missing or invalid.

```
POST /harvest
```

- **Input**

```
{
    "object_identifier":
"https://doi.org/10.1594/PANGAEA.908011"
}
```

- **Output**
  Some fields are truncated or omitted for brevity.

```
{
    "metadata": [
        {
            "format": "application/ld+json",
            "metadata": {
                "access_free": true,
                "access_level": "unrestricted",
                "creator": "Robert Huber",
                "creator_first": "Robert",
                "creator_last": "Huber",
                "language": "en",
                "license": [

"https://creativecommons.org/licenses/by/4.0/"
                ],
```

```
                    // ...
                    "summary": "This data set contains...",
                    "title": "Maximum diameter of..."
                },
                "method": "SCHEMAORG_EMBEDDED",
                "namespaces": [
                    "http://schema.org/"
                ],
                "schema": "http://schema.org",
                "url":
"https://doi.pangaea.de/10.1594/PANGAEA.908011"
            },
            // ...
            ,
        ],
        "target_uri": "https://doi.org/10.1594/PANGAEA.908011"
}
```


– **Description**

This endpoint may be used to optionally harvest metadata given the DOI of a data set.

```
GET /metrics/{version}
```

– **Output**

Some fields are truncated or omitted for brevity.


```
{
    "metrics": [
        {
            "created_by": "FAIRsFAIR",
            "date_created": "2020-07-08",
            "date_updated": "2020-11-25",
            "description": "A data object may be assigned
with...",
            "evaluation_mechanism": "Identifier is
considered...",
            "fair_principle": "F1",
            "metric_identifier": "FsF-F1-01D",
            "metric_name": "Data is assigned a globally
unique...",
            "metric_number": 1,
            "metric_short_name": "Unique Identifier",
            "metric_tests": [
```


```
                ],
                "target": "Data",
                "test_scoring_mechanism": "alternative",
                "total_score": 1,
                "version": 0.5
            },
            // ...
        ],
        "total": 17
    }
```


– **Description**

When given a version of the FAIRness metrics (e.g. 0.5), it retrieves and returns the specified version of the list of metrics used to perform a data set's FAIRness assessment, their descriptions, scores, and additional metadata. This information is loaded directly from the `.yaml` files available in the tool's source code.

`GET /metrics/{version}/{metric}`

– **Output**


```
{
    "created_by": "FAIRsFAIR",
    "date_created": "2020-07-08",
    "date_updated": "2020-11-25",
    "description": "A data object may be assigned with a
globally unique identifier such that ...",
    "evaluation_mechanism": "Identifier is considered unique
if...",
    "fair_principle": "F1",
    "metric_identifier": "FsF-F1-01D",
    "metric_name": "Data is assigned a globally unique
identifier.",
    "metric_number": 1,
    "metric_short_name": "Unique Identifier",
    "metric_tests": [
    ],
    "target": "Data",
    "test_scoring_mechanism": "alternative",
    "total_score": 1,
    "version": 0.5
},
```


– **Description**

When given a version of the FAIRness metrics (e.g. 0.5) and a metric identifier (e.g. FsF-F1-01D), it retrieves and returns a single metric and its definition from the list of metrics used to perform a data set's FAIRness assessment.

The API documentation is also available in the Annex as a Swagger editor `.yaml` file.

7. Implementation Technology

F-UJI is written in the Python programming language and internally utilizes Flask to provide the necessary API endpoints that implement the data FAIRness evaluation. The documentation presents two methods of deploying the tool, either by installing the dependencies and directly executing the F-UJI server or by using the provided Docker image. Since we have adopted a container-based deployment architecture based on Docker, we chose to deploy F-UJI in our premises using the Docker image.

8. Comments

This component has been set up, tested, and deployed. In particular, we have configured the service to accept RE4DY-specific credentials for the API authentication process and we have added a new service which starts the F-UJI container in our deployment scripts. It is currently operational and ready to use in a standalone manner and be integrated with the rest of the RE4DY components.