International FAIR Digital Objects Implementation Summit 2024

Full Papers

https://doi.org/10.52825/ocp.v5i.1415

Published: 18 Mar. 2025

An FDO-Based Implementation for a Standardized Data Exchange

Yudong Zhang[1,*], Claus-Peter Klas[2], Daniel Schiffner[3], Fidan Limani[4], Roland Johannes[3], Alexander Mühlbauer[1], and Peter Mutschke[1]

[1]Leibniz-Institut für Sozialwissenschaften (Gesis), Germany

[2]Till end of 2023, Leibniz-Institut für Sozialwissenschaften (Gesis), Germany

[3]Leibniz-Institut für Bildungsforschung und Bildungsinformation (DIPF), Germany

[4]Leibniz Information Centre for Economic (ZBW), Germany

*Correspondence: Yudong Zhang, Yudong.zhang@gesis.org

Abstract. In this paper, we present an FDO[1] implementation focusing on standardized scientific research data exchange. Using DOIP[2] and Cordra[3] and based on a concrete FDO model with assumed layers and compositions, we implement some basic use-cases of exchanging/handling scientific research data based on FDOs, and introduce FDO services to handle research data based on FDOs. We introduce workflows to orchestrate the application of FDO services. These define and foster an FDO-based ecosystem research data. Our work was funded by the German Research Foundation(DFG) within the KonsortSWD, the consortium for the social, behavioral, educational and economic sciences, is part of the German National Research Data Infrastructure (NFDI) initiative.

Keywords: FDO, DOIP, Scientific Research, Data Exchange, Cordra, FDO-Based Ecosystem, KonsortSWD, NDFI, DGF

1. Introduction

Scientific research data comes from and is used in various domains, each having different attributes, such as formats, schemas and also physical locations. Therefore, effectively modelling, sharing, and using it poses a great challenge for scientific communities. In this context, we see the encapsulation of scientific research data as FDOs of great importance. However, FDOs, that are self-sufficient, referenceable digital entities, including principles to be better located and reused, are still facing several challenges. While we cannot solve all of them, we present our approach to address them.

FAIR Digital Objects (FDOs)[1] represent an approach for modelling and sharing scientific research data. Relying on this model is an effective way to model data so that it is self-sufficient. That means, it is sufficient to understand the data, and it includes all the necessary information, such as metadata to provide information of the data and the defined operations for using/handling the data. This, of course, is only possible within an FDO-based ecosystem, including registries, services, and other components required to realize these objectives. They also include the FAIR principles[4] that, ultimately, improve their FAIRness.

Currently, as the results of 3 years discussion, there are FDO specifications[5], The specifications are on purpose minimal and for all variant implementations. Therefore, the commu-

nities are adopting different competing implementation options[6]. Therefore, all of those options need to be considered for any case or context. Finally, different services within an ecosystem are necessary, which in turn require a common definition of the FDO model.

KonsortSWD, the consortium for the social, behavioural, educational and economic sciences, is part of the German National Research Data Infrastructure (NFDI) initiative. One purpose of KonsortSWD is to develop a research data infrastructure for social, behavioral, educational, and economic sciences[7]. The Task Area 5 (TA5) Measure 3 (M3) of KonsortSWD[8] aims to develop an API for standardized data exchange based on FDOs to overcome the current practice of using heterogeneous, proprietary solutions for data exchange. KonsortSWD is funded by the German Research Foundation(DFG) within NFDI with project number 442494171.

2. FDO Services

FDO services are user self-defined FDO applications that are accessible, documented and published. These services can be viewed, accessed, invoked or executed using the FDO service API provided in our implementation. These have been implemented for linking, getting the raw data of study variables in CSV format (refined data), doing Signposting[9] checks on PIDs, etc. By using FDO services, we promote creating, sharing and reusing FDO applications. Some advantages of our FDO service approach are listed below:

1. Dynamic; it can be used to provide services as needed such as service to get raw data or service for checking data such as doing a Signposting check.

2. Standardized; having standard Restful API for registering/accessing/invoking.

3. Reusable; reusing existing FDO services as needed.

4. Enable; enable using existing services to create new services such as create FDO service C by combining FDO service A and B.

5. Fostering, contributing to an FDO Service ecosystem to foster exchanging and handling of research data via FDOs.

The diagram below describes the general concept of how FDO services work.

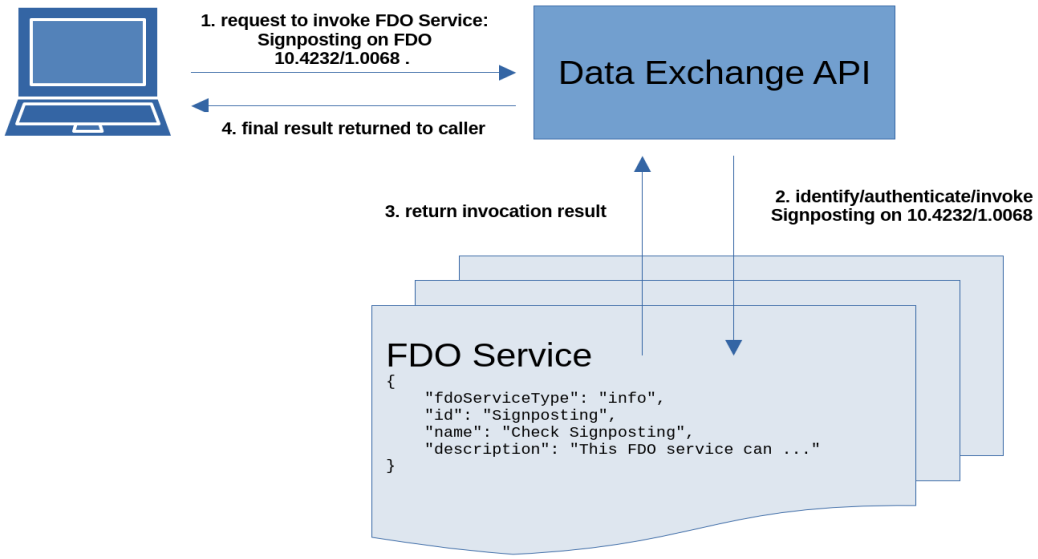

Figure 1. *FDO Services*

FDO services are uniquely identified by their ID. Besides the unique ID, FDO services have properties such as type, name or description. The type attribute enables the user to effectively identify the type of the service offered. When invoking an FDO Service on an FDO, the user needs to provide the FDO either by providing the FDO itself or by providing its PID.

3. FDO Service Registry

To manage available FDO services, we also created an FDO Service Registry that is effectively a central repository of available FDO services. The Registry has a common RESTful API for registering/managing/discovering FDO services. For users of FDO services, it enables them to discover and therefore to reuse existing FDO services. For providers of FDO services, it enables them to register, publish and manage their FDO services. The diagram below illustrates the concept of an FDO Service Registry.

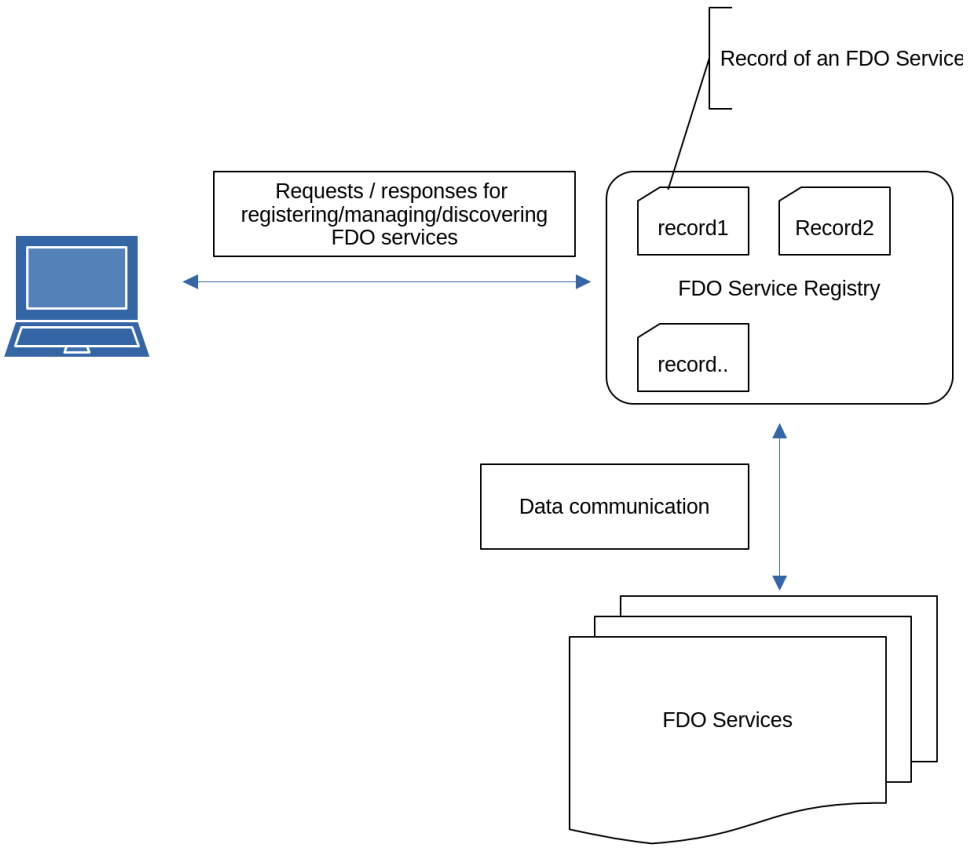

Figure 2. *FDO Services Registry*

Using the common API, new FDO services can be added to the FDO Services Registry. After registration, the FDO services keep their record in the FDO Services Registry. The Registry communicates with the FDO services for keeping the both in sync and up-to-date.

4. FDO Workflows

FDO workflows extend FDO interaction into a predefined process to orchestrate the application of operations on FDOs, for example retrieving data, applying a statistical analysis on the data, or preparing the data for re-publication as a new entity. In each operation the FDOs are processed by an FDO service.

An FDO workflow consists of pipelines. These pipelines are executed one after another according to the defined order. When a pipeline is executed, the FDO service is invoked and the result of the invocation is obtained and then used as input of the next pipeline. This process continues until the last pipeline is executed and the final result is obtained. Below is a diagram to illustrate the concept of applying FDO Workflows with 3 FDO services.

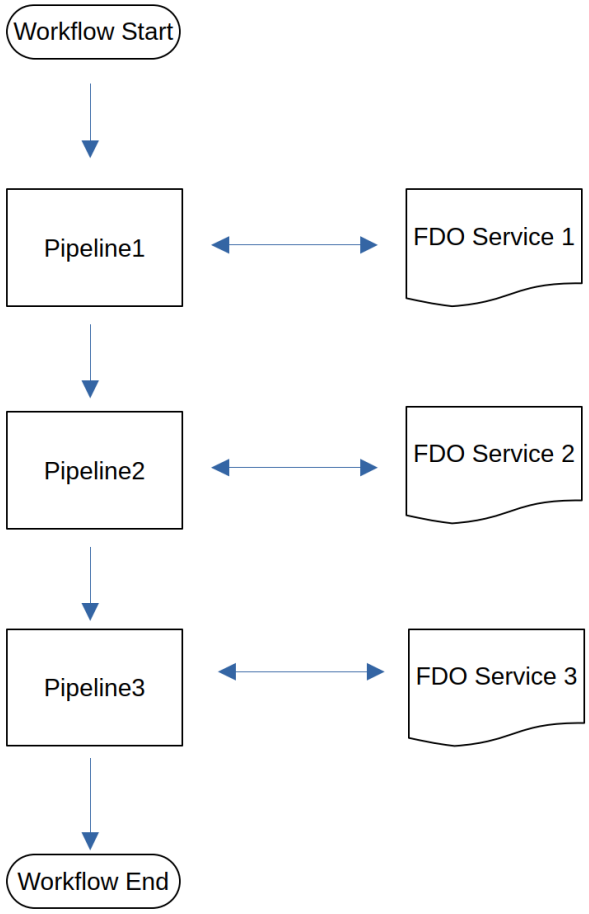

***Figure 3**. FDO Workflows*

In the above workflow, 3 pipelines are defined with 3 FDO services respectively. Starting from executing pipeline1, then pipeline2 and finally pipeline3. The final result is delivered after executing the final pipeline, i.e. the result of the consecutive application of all pipelines in the defined order.

5. Security - Authentication and Authorization

Finally, as authentication and authorization is very critical for research data sharing, we implemented authentication via Single Sign-on (SSO). With SSO, a user using a single set of credentials can authenticate themselves to different research data centers that share their data with each other. This improves both security and the end user experience[10].

6. Technologies

We made use of the following key technologies that are currently available to handle FDOs: Digital Object Interface Protocol (DOIP)[2], a protocol to enable client and server interacting with each other via FDOs and Cordra[3] for managing FDOs with resolvable identifiers. For implementation of authentication with SSO, we used Keycloak[11], a widely used, open source

and easy-to-use application. To implement the FDO workflow, we used the open source orchestration platform Apache Hop[12].

7. Implementation

Our exemplary implementation includes some of the identified common research data exchange use cases including retrieving, searching, linking and registering FDOs. The implementation of these use cases provides a basic feature of data exchanges via FDOs.

7.1 FDO Services

By combining several aspects, our implementation enables dynamically registering and creating FDO Services for user self-defined functionalities, such as getting raw data of an FDO instance or counting the number of videos associated with an FDO instance. Furthermore it is also possible to rely on existing FDOs services from user-defined/community-defined (such as a service to Link FDOs) so that a user is able to reuse existing FDO services from the FDO ecosystem to provide their own service(s). For example, a Link FDO service, that can link different FDOs together to create a linked FDO, can use a Register FDO service to register the created linked FDO. By adopting the FDO model and using predefined FDO services, it is possible to exchange and handle research data without even knowing the research data itself. Also as mentioned above, it is also possible to reuse existing FDO services to create new FDO services. These foster the exchange and handling of research data.

Below we show the implementation of a real FDO service: Signposting Check. As a demonstrative purpose, we do the following simplifications.

1. The response of the service is in simple Json. However, because our FDO Services are flexible to support different response specifications. Therefore, the response of an FDO service can also be implemented to follow different Specifications such as the DOIP specifications[2].

2. Furthermore, in the example, the service ID is a simple hardcoded key string. However, each ServiceID (e.g. FDS_SIGNPOSTING_CHECK) could also be registered as a PID, which points to its definition. The REST API could then take the PID as a parameter instead of using a hardcoded key string.

The Signposting Check service can check/validate signposting on DOIs and landing pages. It has the following record in the FDO Service Registry:

```
{
    "fdoServiceType": "INFO",
    "id": "FDS_SIGNPOSTING_CHECK",
    "name": "FDO SIGNPOSTING CHECK",
    "endPoint": "https://multiweb.gesis.org/doip/api/fdoservices",
    "description": "This service checks signposting links in DOIs and landing pages."
}
```

Figure 4. *Example record in the FDO Service registry*

1. This service is of type INFO meaning that it can provide information.

2. The endpoint is given by the service provider when registering the service in the FDO Service Registry. Therefore, the FDO service engine knows where to call the service. However, the endpoint is not needed by the user of the FDO service.

3. The user of the service only needs to know the ID of the service to invoke the service. In this example, the ID is FDS_SIGNPOSTING_CHECK. That is the ID used for invoking the service.

4. Name and description provide extra information about the service.

The service can be invoked with an example DOI: 10.4232/1.0068 with the request: https://multiweb.gesis.org/doip/api/fdoservices/invoke/FDS_SIGNPOSTING_CHECK?pid=10.4232/1.0068

In the above, the path of the end point is /api/fdoservices/invoke/{fdo-service-id}?pid={pids}. In our example the ID of the service is "FDS_SIGNPOSTING_CHECK" and the PIDs is 10.4232/1.0068. If more than one PID, they need to be separated by a comma. The request returned by calling this service with the PID is as follows.


```
{
    "responseCode": 1,
    "type": "Signposting Check",
    "content": [
        {
            "signpostingCheck": {
                "PID": "https://doi.org/10.4232/1.0068",
                "signpostingHtmlLinks": {
                    "1": "describedby: <https://search.gesis.org/getSchemaOrg.php?id=ZA0068>",
                    "2": "type: <https://schema.org/Dataset>",
                    "3": "type: <https://schema.org/AboutPage>"
                },
                "signpostingHttpLinks": {}
            }
        }
    ]
}
```


Figure 5. *Example response of calling the FDS_SIGNPOSTING_CHECK service.*

PIDs is 10.4232/1.0068. If more than one PID, they need to be separated by a comma. The request returned by calling this service with the PID is as follows:

1. The responseCode is 1 meaning everything is OK.
2. The type is the type of the performed action. It is a Signposting Check in this example.
3. The checking result is contained in the content.
4. In our example the result is in JSON. However, FDO services are user defined applications. Therefore, the result of calling a FDO service can be of different content types and different schema depending on the implementation of the FDO service.

7.2 FDO Workflows

FDO workflows orchestrate the process of handling FDOs in a predefined manner. In the example below, a FDO workflow is given (c.f. Figure 6) that counts associated videos of an FDO and then registers the result as a new FDO. This workflow automates the handling of FDO in a step-by-step manner. Therefore, it provides reproducible operations of research data.

Figure 6. *Example workflow to process FDOs*

A summary of the workflow process depicted in Figure 6 above is as follows:

1. Pipeline 1: Provide PID, PID is submitted.
2. Pipeline 2: Retrieve FDO, the FDO of the given PID is retrieved.

3. Pipeline 3: Count videos, the retrieved FDO is given to the count video FDO Service to count the number of videos associated with the FDO.
4. Pipeline 4: Register FDO, the result returned from the count video service is registered as a new FDO.

The workflow above demonstrates the ability of applying FDOs in a workflow manner via invoking different FDO services in different pipelines to process FDOs in a predefined manner. Apache Hop is used as a workflow engine for orchestrating the process.

Also important to mention is that the countVideoFDOSerivce does not rely on the operation layer of the FDO, but the FDO service itself to determine what operation itself can be performed on FDOs. This is to show the generalized approach of FDO service without having to rely on the operation layer of FDOs. On the other hand, it is also possible to use the FDO operation layer to provide further operational information when needed.

8. Architecture

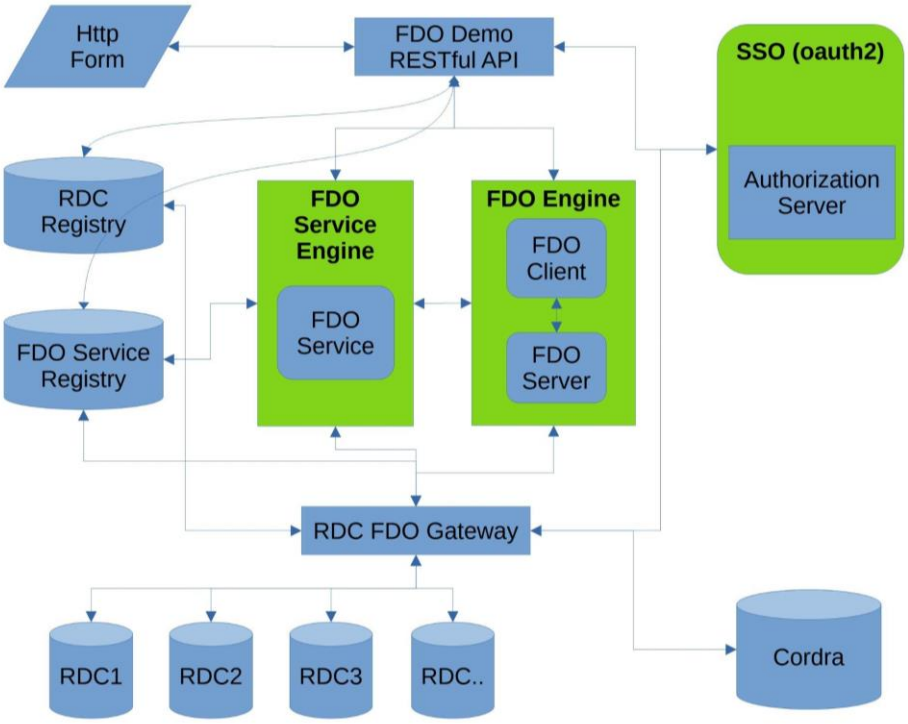

Figure 7. *Architecture diagram of the FDO DEMO implementation.*

The used architecture is shown in Figure 7 above. The RESTful API provides generic access to the features of the FDO DEMO such as retrieving/searching FDOs, executing registered FDO services, enquiring RDC gateway/FDO service registries etc. The FDO Service Engine instantiates and executes FDO services. While the FDO Engine prepares internal requests, harmonizes the data exchange, handles the actual operations of modeling data into FDOs etc, the RDC FDO Gateway distributes the internal request after authorization to the corresponding Research Data Centers (RDC), that deliver the actual data based on the assumed layers of our FDO model. We use Cordra to manage and store the newly registered FDOs. The RDC Registry stores the needed information of the RDCs such as their data access endpoint and ID and enables the RDC FDO Gateway to acquire data from the RDCs. Similarly, the FDO Service Registry stores the needed information of the registered FDO Services so as to enable the FDO service engine to execute the FDO services.

While the API provides a set of basic common features to satisfy the most common needs, FDO services, which can be created/registered dynamically, enable the user of the API to define whatever feature they need for handling their FDOs. While the basic common features are operated by the FDO Engine/RDC FDO Gateway etc. internally, the operation of the FDOs service can be executed externally or internally. This is subject to the implementation of the FDO services. The internal communication only utilizes the layers of the FDO, allowing for a data agnostic exchange i.e. data exchange without knowing what the data actually are. The FDO Service Registry uses metadata to identify services that are able to perform different features.

In general, with our approach, the user can use the basic common features of the API to satisfy their needs for some common use cases such as retrieving FDOs. When the common features are not sufficient, they can look up the FDO Service Registry to see if there are services that suit their specific needs. If they find the services, they can reuse them. Finally, they can always define/register/invoke their self-defined FDO services to realize the specific features they need.

We use Keycloak to provide SSO. The user needs to authenticate themselves before getting access to the services, API and workflows. The granularity of the access is handled independently by each service provider. Each RDC keeps full authority over the data that which possess and maintain and can limit data access based on the identity in the incoming request.

9. Conclusion and Next Steps

We present an FDO implementation focusing on standardized scientific research data exchange. Using DOIP and Cordra and based on a concrete FDO model with assumed layers and compositions, we implement some basic use-cases of exchanging/handling scientific research data based on FDOs, and introduce FDO services to handle research data based on FDOs. We introduce workflows to orchestrate the application of FDO services. These define and foster an FDO-based ecosystem research data.

We will continue to identify new use cases in the KonsortSWD community and provide corresponding integrations based on the proposed design. Furthermore, we plan to adopt the data exchange interface API into existing protocols (DOIP and Cordra) to handle FDO instances.

Data availability statement

This submission is not based on data.

Author contributions

Yudong Zhang: Conceptualization, Methodology, Software, Writing - Original draft preparation, Writing – Reviewing and Editing;

Claus-Peter Klas: Supervision, Conceptualization, Methodology, Software;

Daniel Schiffner: Supervision, Conceptualization, Writing – Reviewing and editing;

Fidan Limani: Conceptualization, Writing – Reviewing and editing;

Roland Johannes: Conceptualization, Writing – Reviewing and editing;

Alexander, Mühlbauer: Methodology, Software;

Peter Mutschke: Supervision, Conceptualization, Methodology, Writing – Reviewing and editing;

Competing interests

The authors declare that they have no competing interests.

Funding

KonsortSWD is funded by the German Research Foundation(DFG) within the framework of the NFDI – project number 442494171.

References

[1] FAIR Digital Objects, https://fairdo.org/1316-2. Accessed 08 Mar. 2024.
[2] Digital Object Interface Protocol Specification V2.0, https://www.dona.net/sites/default/files/2018-11/DOIPv2Spec_1.pdf accessed 08.Mar. 2024
[3] Cordra, Highly Configurable Software for Managing Digital Objects at Scale. https://www.cordra.org/ Accessed 08 Mar. 2024.
[4] Wilkinson, M., Dumontier, M., Aalbersberg, I. et al. The FAIR Guiding Principles for scientific data management and stewardship. Sci Data 3, 160018 (2016). https://doi.org/10.1038/sdata.2016.18
[5] Ivonne, A., Christophe, B., Daan, B., Maggie, H., Sharif, I., Thomas, J., Larry, L., Karsten, P.-. von G., Robert, Q., Alexander, S., Ulrich, S., Stian, S.-R., Strawn, G., Dieter, . van U., Claus, W., Peter, W., & Carlo, Z. (2023). FDO Forum FDO Requirement Specifications. https://doi.org/10.5281/zenodo.7782262
[6] Larry, L., Karsten, P.-. von G., Ivonne, A., Andreas, P., Alexander, S., Zach, T., & Peter, W. (2022). FDO Forum FDO Configuration Types. https://doi.org/10.5281/zenodo.7825703
[7] KonsortSWD, the consortium for the social, behavioral, educational and economic sciences. https://www.konsortswd.de/en/. Accessed, 05.07.2024
[8] Limani, F., Johannes, R., Zhang, Y., & Schiffner, D. (2022). KonsortSWD Task Area 5 Measure 3: Milestone 1 & 2 Report (0.1). Zenodo. https://doi.org/10.5281/zenodo.6497190

[9] Signposting the Scholarly Web, https://signposting.org/, Accessed, 05.07.2024
[10] Sign-On (SSO): Single Sign-On (SSO): Advantages and Disadvantages | Gartner Peer Community, Accessed 31.05.2024
[11] Keycloak, Open Source Identity and Access Management. https://www.keycloak.org/. Accessed 08 Mar. 2024
[12] Apache Hop, the Hop Orchestration Platform, https://hop.apache.org/. Accessed 08 Mar. 2024