RESEARCH PAPER

FAIR Data Point: A FAIR-Oriented Approach for Metadata Publication

Luiz Olavo Bonino da Silva Santos[1,2†], Kees Burger[2], Rajaram Kaliyaperumal[2], Mark D. Wilkinson[3]

[1]Services and Cybersecurity group, Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente - Enschede, 7513 GB, the Netherlands

[2]Biosemantics group, Department of Human Genetics, Leiden University Medical Center - Leiden, NL 2333 ZC, the Netherlands

[3]Escuela Técnica Superior de Ingeniería Agronómica, Alimentaria y de Biosistemas; Centro de Biotecnología y Genómica de Plantas UPM – INIA; Universidad Politécnica de Madrid (UPM) - Instituto Nacional de Investigación y Tecnología Agraria y Alimentaria (INIA), Madrid, ES 28223, Spain

Keywords: FAIR; FAIR data point; FAIR principles; Metadata; Interoperability; Linked data; Semantic interoperability

Citation: da Silva Santos, L.O.B, Burger, K., Kaliyaperumal, R., et al.: FAIR Data Point: A FAIR-Oriented approach for metadata publication. Data Intelligence (5) 2023 doi: 10.1162/dint_a_00160

ABSTRACT

Metadata, data about other digital objects, play an important role in FAIR with a direct relation to all FAIR principles. In this paper we present and discuss the FAIR Data Point (FDP), a software architecture aiming to define a common approach to publish semantically-rich and machine-actionable metadata according to the FAIR principles. We present the core components and features of the FDP, its approach to metadata provision, the criteria to evaluate whether an application adheres to the FDP specifications and the service to register, index and allow users to search for metadata content of available FDPs.

1. INTRODUCTION

The FAIR principles [1] quickly gained significant attention around the world as they address and condense a set of long-lasting concerns about how we treat data and other types of digital objects. The principles define a set of behaviours expected from the digital infrastructure to make digital objects more

† Corresponding author: Luiz Olavo Bonino da Silva Santos (Email: l.o.boninodasilvasantos@utwente.nl; ORCID: 0000-0002-1164-1351).

findable, accessible, interoperable and reusable. The principles are intended to be used, therefore, as guidelines to help developers implement these expected behaviours.

One particular aspect that is highly relevant in the FAIR principles is metadata. The principles make a clear distinction between metadata and the digital object they describe. When the FAIR principle F2 states that "*data are described with rich metadata*", it assigns a description role for metadata. The reusability principle and its sub-principles elaborate on this role, requesting that metadata include information about usage license, provenance and compliance with relevant standards.

Additionally, the FAIR principles aim at machine-actionability, i.e., the ability of machines to act based on information encountered during their autonomous exploration of the digital environment. This is a step forward from machine-readability, which reflects the ability of machines to parse documents. Machine-actionability encompasses machine-readability, expanding it to include the capacity to interpret the read (or parsed) content. Part of the interpretation depends on recovering the meaning (semantics) originally attributed to the document's content.

Focusing on the importance of machine-actionable metadata, this paper reports on the FAIR Data Point (FDP)—an approach to exposing semantically-rich metadata for a wide range of digital objects in a FAIR manner. The FDP allows digital object owners/publishers to expose their metadata in a FAIR manner and, allows digital objects' consumers to discover information (metadata) about the resources offered. Frequently, FAIR Data Points are used to expose metadata of datasets, but metadata of other types of digital resources can also be exposed, such as ontologies, repositories, workflows, analysis algorithms, websites, and even physical entities such as biobanks, people and organizations.

The main goal of the FDP is to establish a common method for metadata provisioning and accessing that is compliant with the FAIR principles. A direct consequence is that client applications have a predictable way of accessing and interacting with metadata content. To fulfil this goal, we have created two types of artefacts. A set of specifications to help developers extend the functionality of their current and new applications so that they behave also as FAIR Data Points and a reference implementation for those who would like to have the FDP functionality in a stand-alone web application that is ready to be deployed. In [2] we presented the initial ideas and preliminary prototype. In the subsequent 6 years the work on the FDP progressed significantly. This paper extends and updates that early report, discusses the architectural design of the contemporary FDP, its metadata provisioning approach, and reports on the current status of the reference implementation, supporting services, and upcoming developments.

This paper is structured as follows: section 2 discusses the FDP architecture with its components and main features. Section 3 elaborates on the metadata provision approach taken by the FDP. Section 4 presents the criteria to evaluate whether an application behaves as a FDP. Section 5 presents the service to index the metadata of available FDPs. Section 6 reports on the progress of the reference implementation. Section 7 discusses the next steps of the design and development, and provides some final remarks.

2. FDP ARCHITECTURE

The FDP as a FAIR-compliant metadata provider has the following goals:

- Allow owners/creators/publishers to expose the metadata of their digital objects in a way that follows the FAIR principles;
- Allow consumers/users to discover information about digital objects of interest;
- Provide this metadata in a machine-actionable way.

Based on these goals, Figure 1 depicts the general architecture of the FDP using the Archimate [3] architecture modeling language. The FDP exposes its functionality through an application programming interface (API). The figure depicts the following elements:

- **FDP API**—The FDP API is the way client applications interact with the FDP. All functionality offered by the FDP should be available through its API.
- **Metadata Provider Service**—The main service of the FDP and is responsible for serving the stored metadata records required by client through the FDP API.
- **Access Control Function**—The Metadata Provider Service should support access control over the available metadata. Minimally, the access control function manages who can add or edit the metadata records. The access control may also restrict access to (parts of) metadata. For instance, some metadata content may be too sensitive for unrestricted access and, therefore, only a selection of stakeholders may have access to them.
- **Metadata Storage**—The service that stores the metadata records made available through the FDP. The metadata storage may be an internal component of the FDP or may be an external and independent component. However, in the latter case, the FDP must have access to this storage facility to store new metadata records and retrieve them whenever necessary.
- **Metadata Record**—The metadata record is the artefact that contains properties and relations that are used to describe a related digital object. In the FDP the metadata record is presented as RDF. Therefore, the Metadata Storage service is, normally, an RDF store. A given implementation of the FDP may store the metadata records in other formats; however, when those metadata records are requested from the FDP's API, the content must be transformed into RDF.
- **Metadata Schema Storage**—The service storing a number of different metadata schemas that define the structure and semantics of the acceptable metadata records for that given FDP.
- **Metadata Schema**—The metadata schema is an artefact that defines the structure and semantics of its related metadata records. For instance, a metadata schema of a dataset defines the properties that should be present in metadata records of datasets.

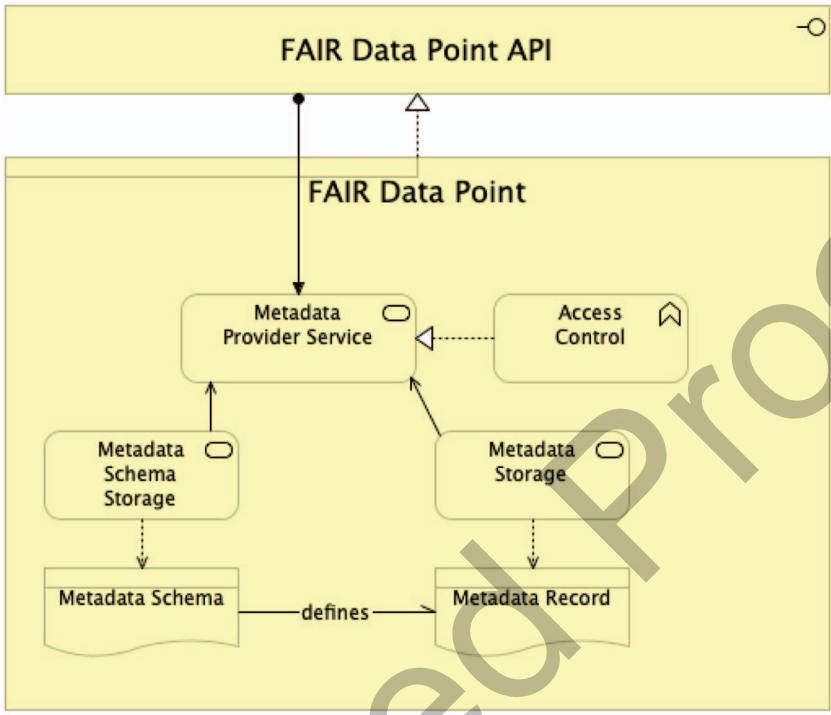


Figure 1. General architecture of the FDP.

The FDP architecture is intended to be used as a guideline for developing new and adapting existing applications to expose metadata of themselves and their content following the FAIR principles. Therefore, a number of applications can implement the FDP architecture and underlying metadata publication approach and behave as a FDP. This will make these applications interoperable with others also following the FDP architecture. Naturally, besides presenting their metadata according to the FDP architecture, these FDP-compliant applications may have their own functionality and purpose. For this reason, the FDP architecture only covers the functionality of metadata provisioning.

The FDP has been designed to address a number of the FAIR principles related to metadata, as well as issues commonly faced by consumers and producers of digital objects. Below we list the main concerns, the FAIR principles related to each concern, what the consequences are and how the FDP addresses them. In all cases, the word *data* is used to refer to any type of digital object.

2.1 Clear Separation Between Metadata and Data

Several of the FAIR principles refer to "*(meta)data*". This means that the related principle applies to both metadata and data. The first FAIR principle states that

F1. (meta)data are assigned a globally unique and persistent identifier

From this principle we have that both metadata and data should have their own identifiers. The FDP supports this requirement by providing to each metadata record its own identifier. The FDP also allows the inclusions of the data's identifier in the metadata records.

The FAIR principle F3 states that

F3. metadata clearly and explicitly include the identifier of the data it describes

Some approaches embed the metadata in the same file of its described digital object. Examples of such approaches are a number of image and video file formats such as JPG, DICOM and MPEG. In these examples, the metadata is encoded in the header of the file format and, therefore, the accessibility to the metadata is only possible when accessing the whole media file. Other approaches, such as common identification systems, couple both metadata and the described digital object with the same identifier. The main drawback of approaches that do not differentiate metadata from the digital objects they describe is that, if the metadata are gathered and stored in a different location than their associated data (e.g. after being indexed by a search engine), the connection between metadata and data is lost. In these situations, we can no longer determine the unique identity of the object described by the metadata.

To address this issue, FAIR principle F3 requires that, within a given metadata record, we can find the identifier of the digital object it describes. This covers the requirement of *explicitly* including the identifier of the described object in the metadata. However, a metadata record may contain a potentially large number of other identifiers, e.g., identifiers of properties, types and values of these properties. In this case, how can we make sure that a metadata consumer will be able to identify which of these identifiers refer to the described object? To answer this question principle F3 also requires that the identifier of the described object is *clearly* included in the metadata. In order to properly follow this part of the principle we defined in the FDP ontology [4] the property *isMetadataOf*. An example of the usage of this property (in RDF Turtle) is:

```
<MetadataIdentifier> fdp-o:isMetadataOf <ObjectIdentifier>
```

Additionally, the FDP ontology defines the *metadataIdentifier* property to more clearly indicate that a given identifier refers to the metadata and not to the object this metadata describes.

2.2 Common Metadata Access Mechanism

One of the most successful initiatives for supporting a common mechanism for metadata provisioning is the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) [5]. The protocol defines, among others, an HTTP-based Application Programming Interface (API) that needs to be implemented by metadata repositories and their clients (called Harvesters by OAI-PMH) in order to have allow the harvesting of the metadata content.

The FDP architecture also follows this approach of providing an HTTP-based API for metadata harvesting. The main difference is that, instead of requiring the client applications to "*know*" beforehand the API, the

FDP adopts a "*follow your nose*" approach. In this approach the client application starts the interaction with the FDP in two situations: (1) from the root API of the FDP or (2) by having the URI of a metadata record provided by the FDP. In either way, the FDP will provide a metadata record[①] containing information on how to navigate up and down the FDP's metadata structure. This approach follows the guideline stated in the FAIR principles paper [1] where we should enable "*machines to be capable of autonomously and appropriately acting when faced with the wide range of types, formats, and access-mechanisms/protocols that will be encountered during their self-guided exploration of the global data ecosystem*".

3. METADATA

The FAIR principles give special attention to metadata. In fact, all principles relate to metadata in at least one aspect. Metadata can be defined as data that provides information about other data. Here we extend this notion to define metadata as information about other digital objects. This information can include a variety of different descriptions such as origin, structure, provenance, rights and obligations or other characteristics of digital objects. The FAIR Data Point first provides metadata about the FDP itself. When a client interacts with a service, it should be capable of determining its features. Therefore, the FDP provides metadata about itself and, from that point on, the client can navigate its metadata content to discover the other metadata records.

The FDP uses the W3C's Data Catalog Vocabulaire (DCAT) version 2 [6] model as the basis for its metadata content. Figure 2 depicts the FDP extensions to the DCAT model.

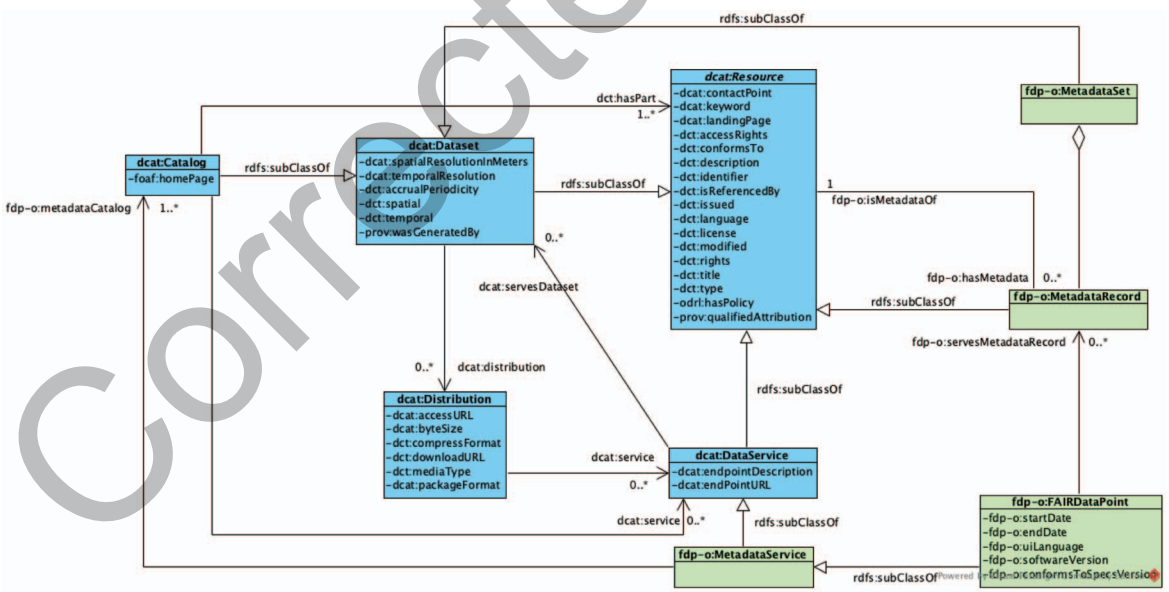


Figure 2. FDP extensions to the DCAT 2 model.

[①] The metadata record from the root API of the FDP is the metadata record of the FDP itself as a data repository.

In DCAT, the class Resource represents entities that can be described by a metadata record. Since it has been defined as an abstract class, it is not intended to be used directly. We should use one of its subclasses, such as Dataset or Data Service instead. Dataset represents a collection of data while Data Service represents services accessible through an interface (API) that serve datasets. Catalog, a subclass of Dataset, represents aggregations of metadata records about digital objects. For instance, a Catalog may contain references to the metadata records of Datasets.

The FDP extendes the DCAT model by adding the concept of FAIRDataPoint as a specific type of data service that serves metadata catalogs and metadata records. The DCAT extensions and other FDP-specific concepts and relations are defined in the FDP ontology (using the namespace prefix fdp-o in the figure). In the FDP ontology, the FAIR Data Point is represented by a sub-class of the concept of MetadataService. Figure 2 only depicts the properties that are not already inherited from Data Service and Resource.

With the definition of the FDP as a type of metadata service that serves metadata records grouped in catalogs, the relation between the Metadata Service and dcat:Catalog is represented by the predicate fdp-o:metadataCatalog. Following the DCAT approach of providing qualified relations between resources, the fdp-o:metadataCatalog is defined having fdp-o:MetadataService as its domain and dcat:Catalog as its range.

3.1 Navigation Information

Since the FDP supports the provisioning of metadata about different types of digital objects and the relations among these metadata records can be customized, each FDP installation may have a different structure. The FDP specifications [7] define that, minimally, we have the metadata of the FDP itself and at least one catalog is mandatory. As defined by DCAT, catalogs are arbitrary collections of metadata records about different types of digital objects (called resources in DCAT specifications). Accordingly, in the FDP, catalogs are used to group and organise metadata records. For instance, one can define a catalog of datasets to group the metadata about different datasets and a catalog of ontologies to group the metadata about a number of ontologies.

From the catalog on, the metadata structure of the FDP will vary from deployment to deployment. For instance, the FDP reference implementation is pre-loaded with the structure of metadata about FAIR Data Point → Catalog → Dataset → Distribution. Another FDP could have a different metadata structure, e.g., FAIR Data Point → Catalog → Semantic Artefact. With this flexibility, a client application would not be able to know how to navigate the FDP metadata content unless it follows all URIs in the metadata, which may be extensive. To tackle this issue, the FDP informs its own navigation structure by using the Linked Data Platform (LDP) [8] containment predicates ldp:contains or ldp:hasMemberRelation. This information is present in every metadata record that may lead to other metadata records.

In Listing 1, the RDF Turtle code shows an example of Metadata Service metadata record with navigation information (lines 18-27).

```
@prefix dct:    <http://purl.org/dc/terms/> .
@prefix fdp-o: <http://purl.org/fdp/fdp-o#> .
@prefix ldp:    <http://www.w3.org/ns/ldp#> .
<https://fairdatapoint.org/app> a fdp-o:FAIRDataPoint
   dct:title "Demonstration FAIR Data Point";
   dct:conformsTo <https://www.purl.org/fairtools/fdp/schema/0.1/fdpMetadata>;
   dct:description "This FAIR Data Point deployment is used for demonstration of
       the application and to allow the navigation through its metadata content.
       The metadata presented here is also for demonstration purposes only and not
        necessarily describe properly their related resources.";
   dct:hasVersion "1.0" ;
   dct:license <http://rdflicense.appspot.com/rdflicense/cc-by-nc-nd4.0>;
   dct:publisher <https://purl.org/fairdatapoint/app#publisher>;
   fdp-o:metadataCatalog
     <https://purl.org/fairdatapoint/app/catalog/3dde263d-0a7c-4f2a>,
     <https://purl.org/fairdatapoint/app/catalog/508d3c96-8106-49d2> ,
     <https://purl.org/fairdatapoint/app/catalog/6e43c568-c6fa-41c8> ,
     <https://purl.org/fairdatapoint/app/catalog/a46aa445-3be0-4db8> ,
     <https://purl.org/fairdatapoint/app/catalog/a91d3db7-fe83-4de1> .

<https://purl.org/fairdatapoint/app/catalog/> a ldp:DirectContainer ;
   dct:title "Catalogs" ;
   ldp:membershipResource <https://fairdatapoint.org/app> ;
   ldp:hasMemberRelation fdp-o:metadataCatalog ;
   ldp:contains
     <https://purl.org/fairdatapoint/app/catalog/3dde263d-0a7c-4f2a> ,
     <https://purl.org/fairdatapoint/app/catalog/508d3c96-8106-49d2> ,
     <https://purl.org/fairdatapoint/app/catalog/6e43c568-c6fa-41c8> ,
     <https://purl.org/fairdatapoint/app/catalog/a46aa445-3be0-4db8> ,
     <https://purl.org/fairdatapoint/app/catalog/a91d3db7-fe83-4de1> .
```

Listing 1. FDP metadata example with navigation information.

In this metadata record example, the FDP is identified by the URI *https://fairdatapoint.org/app* and is declared as being an instance of the class *fdp-o:FAIRDataPoint* (line 4). On lines 11-16 it is declared that the FDP is related to a number of catalogs using the predicate *fdp-o:metadataCatalog*. This predicate represents the parent-child relation that should be followed by a client that wants to navigate the FDP metadata structure from the FDP (as metadata repository) metadata record to the metadata records of its catalogs. To explicitly inform the navigation structure, we have the code segment from line 18 to 27. There we declare that we have a container identified as *https://purl.org/fairdatapoint/app/catalog/*, which is a container of type *ldp:DirectContainer* (line 18). This container is responsible for relating the *https://purl.org/fairdatapoint/app* as the object of the *ldp:membershipResource* (line 20) with is members using the custom membership relation *fdp-o:metadataCatalog*. This information alone is enough for the client application to know that it can follow the links with the relation *fdp-o:metadataCatalog* in the metadata record (lines 12 to 16) to go to the FDP's catalogs. However, for completion, the LDP segment (lines 18 to 27) repeats the URIs of the catalogs using the LDP's generic *ldp:contains* relation (line 22). The client application can follow either the *fdp-o:metadataCatalog* or the *ldp:contains* predicates to navigate down the metadata structure.

3.2 Metadata Schemas

As discussed in the last section, different FDP installations may provide metadata about different types of digital objects. Once again the FDP follows the DCAT model where different types of digital objects are defined as sub classes of DCAT Resource. DCAT already provides the definitions for datasets and data services. When needed, the data steward responsible for the organisation of the FDP can define the DCAT extensions and then define the metadata schema for the specific type of digital object. For instance, one may want to provide metadata about semantic artefacts such as ontologies, vocabularies, taxonomies, etc. Once the class Semantic Artefact has been defined as a sub class of DCAT Resource, the data steward can proceed to create the metadata schema about semantic artefacts.

It is a good practice that the DCAT extensions are explicitly defined and, therefore, the artifact containing these definitions should be published and reachable. Since the FDP provides metadata about itself as a metadata repository service and introduces some other classes and relations such as *isMetadataOf* and *servesMetadata*, we have created the so-called FDP Ontology (FDP-O) [4] to include these definitions. The listing 2 shows an excerpt of the FDP-O, expressed in OWL [9], with the declarations of the class FAIRDataPoint (lines 8-12) as a sub-class of MetadataService, which is defined (lines 1-6) as a sub-class of DCAT's DataService, as depicted in Figure 2. With this we have declared our DCAT extension and then we can define metadata schemas having these classes as subject targets.

In the FDP, the metadata schemas are expressed using the Shapes Constraint Language (SHACL) [10]. The metadata schema in SHACL are used in the FDP to inform that a given metadata record conforms to a given metadata schema, and to allow the FDP to validate a metadata record that has been submitted via its API to be published.

```
<owl:Class rdf:about="https://w3id.org/fdp/fdp-o#MetadataService">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/ns/dcat#DataService"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string">In the
        context of FAIR Data Point these two classes can be treated as exactMatch
        . Since class Repository has been used in version 1 of FDP specs</
        rdfs:comment>
    <rdfs:label>Metadata service</rdfs:label>
    <skos:closeMatch rdf:resource="http://www.re3data.org/schema/3-0#Repository"
        />
</owl:Class>

<owl:Class rdf:about="https://w3id.org/fdp/fdp-o#FAIRDataPoint">
    <rdfs:subClassOf rdf:resource="https://w3id.org/fdp/fdp-o#MetadataService"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">FAIR Data
         Point</rdfs:label>
    <skos:closeMatch rdf:resource="http://www.re3data.org/schema/3-0#Repository"
        />
</owl:Class>
```

Listing 2. Fragment of the FAIR Data Point Ontology.

The following SHACL listing 3 shows as an example a fragment of the metadata schema definition for catalog.

The metadata schema used by a given metadata record is informed using the Dublin Core Terms' *conformsTo* predicate. According to the DCAT guidelines, the *dct:conformsTo* predicate should point to a profile defined using the W3C's Profile Vocabulary [11], which includes the reference to the metadata schema. Listing 4 provides an example of profile the FDP metadata schema. In this profile we defined that we have an artefact (line 11), which is serialised in RDF Turtle (line 10) using the SHACL vocabulary (line 12) and is used for validation (line 13).

4. FDP COMPLIANCE CRITERIA

The main motivation for the design and development of the FDP was to explore how a metadata publication service would behave in light of the FAIR principles. The original idea was to define a set of expected behaviors and metadata presentation approaches to guide developers in developing their

```
@prefix      : <http://fairdatapoint.org/> .
@prefix dcat: <http://www.w3.org/ns/dcat#> .
@prefix  dct: <http://purl.org/dc/terms/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix   sh: <http://www.w3.org/ns/shacl#> .
@prefix  xsd: <http://www.w3.org/2001/XMLSchema#> .

:CatalogShape a sh:NodeShape ;
  sh:targetClass dcat:Catalog ;
  sh:property [
    sh:path dct:title ;
    sh:nodeKind sh:Literal ;
    sh:minCount 1 ;
  ], [
    sh:path dct:hasVersion ;
    sh:nodeKind sh:Literal ;
    sh:maxCount ;
  ], [
    sh:path dct:description ;
    sh:nodeKind sh:Literal ;
  ].
```

Listing 3. Example of metadata schema definition in SHACL.

FAIR-compliant metadata services or extending their existing applications to expose metadata in a FAIR way. Assuming the existence of a number of different implementations of the FDP specifications, it is useful to define a set of compliance criteria such that we can guarantee interoperability among them.

For this purpose we have defined the following characteristics that must be present in any application that intends to behave as a FDP:

1. **The root API URL must provide the Metadata Service metadata.** The application's root API URL must serve the metadata record of the FDP itself, properly typed using the FDP-O class of *fdp-o:FAIRDataPoint* which is a DCAT extension.

```
@prefix dct: <http://purl.org/dc/terms/> .
@prefix prof: <http://www.w3.org/ns/dx/prof/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix fdp: <https://purl.org/fairdatapoint/app/> .

fdp:profile/fdpprofileexample a prof:Profile;
  rdfs:label "FAIR Data Point Profile";
  prof:isProfileOf fdp:profile/core;
  prof:hasResource [ a prof:#ResourceDescriptor;
      dct:format <https://w3id.org/mediatype/text/turtle>;
      prof:hasArtifact fdp:shapes/fdpshaclexample
      dct:conformsTo <https://www.w3.org/TR/shacl/>;
      prof:hasRole prof:role/Validation;
    ] .
```

Listing 4. Example of profile with reference to the metadata schema.

2. **The metadata records must be presented in, at least, RDF Turtle and JSON-LD syntaxes.** Although potentially controversial, this requirement aims at guaranteeing a common presentation format. Currently, it is difficult to develop client applications that can consume metadata given the wide variety of metadata formats available. The choice in the FDP for RDF-based serialisation format is justified by its capability of embedding semantic references together with the metadata property values. Naturally, applications are free to internally store the metadata content in any format so long as the FDP interfaces return the content in, at least, these two RDF syntaxes. Moreover, developers are also free to offer additional serialisation formats through content negotiation.
3. **Each metadata record provides a reference to its own schema.** As discussed in section 3.2, besides providing provenance information about the metadata record, the information regarding to which schema a given record conforms is used by the FDP to validate entries. Since the schema is also used for validating metadata records expressed in RDF, it must be expressed using SHACL.
4. **Metadata records describe sub-classes of DCAT Resource.** In order to accomplish this, the metadata schema for any give metadata record should have as its target class a sub-class of DCAT Resource.
5. **Metadata records should provide navigation information.** To allow client applications that did not have any prior information about the FDP's metadata structure to navigate seamlessly through its content, the FDP must provide the navigation information using the Linked Data Platform containment structure discussed in section 3.1.

With these objective compliance criteria, we can build checkers and evaluators to automatically assess the compliance of any given application to the FDP specifications. Additionally, more domain or application-specific criteria can be added. For instance, for a particular application, it may be agreed that the involved FDPs would require the metadata records to adhere to a specific schema.

5. FDP INDEX SERVICE

The FAIR principle F4 states that

F4. (meta)data are registered or indexed in a searchable resource

The FAIR Data Point follows this principle to the extent that it supports the publication of rich metadata that has the potential to be registered or indexed in a search engine. However, this potential is only going to be realized if and when the metadata content of the FDP has been effectively registered. In order to address this issue, we have developed an FDP Index service.

The FDP Index provides an API call where an FDP can request to be registered via its URL. The FDP Index registers follows this URL and ensures that the URL represents an FDP. This check is done based on the FDP compliance criteria presented in section 4. Once verified, the FDP Index proceeds to harvest the metadata content of the newly registered FDP. The harvested metadata content is then indexed in the FDP Index and is available for search. Figure 3 depicts the home page of the demonstration deployment of the FDP Index that can be seen at https://home.fairdatapoint.org.

In order to keep the indexed metadata as up to date as possible, the FDP Index visits the registered FDPs periodically (the default frequency is once a week). Moreover, the FDPs can also notify the FDP Index of changes in their metadata content, which triggers the FDP Index to schedule a new metadata harvest on that FDP.

The FDP Index has been designed with the same decentralized spirit as the FDP. In this way a variety of different topological configurations can be defined. Figure 4 depicts some examples of topologies for FDP Indexes. In this example, we have a number of domain-specific indexes. There, each of these indexes would index the metadata content of the FDPs in those domains. This keeps the indexes focused on the content of their respective domains. However, to have a cross-domain findability, these indexes could be indexed themselves in, for instance, a research-specific metadata search. In another situation, organisations may not want to have the metadata of their digital objects discoverable by third-parties. In this case, the organisation can establish an FDP Index behind its firewall. For the metadata records that are allowed to be seen more publicly, they may have an additional FDP Index outside the organisation's firewall. And finally they may have a larger FDP index that indexes other existing indexes such as the public corporate and research FDP Indexes. Moreover, a given FDP can be connected to multiple FDP Indexes, which increases even further the flexibility of the metadata indexing and improves the possibilities for the digital object described by a given metadata record to be discovered.

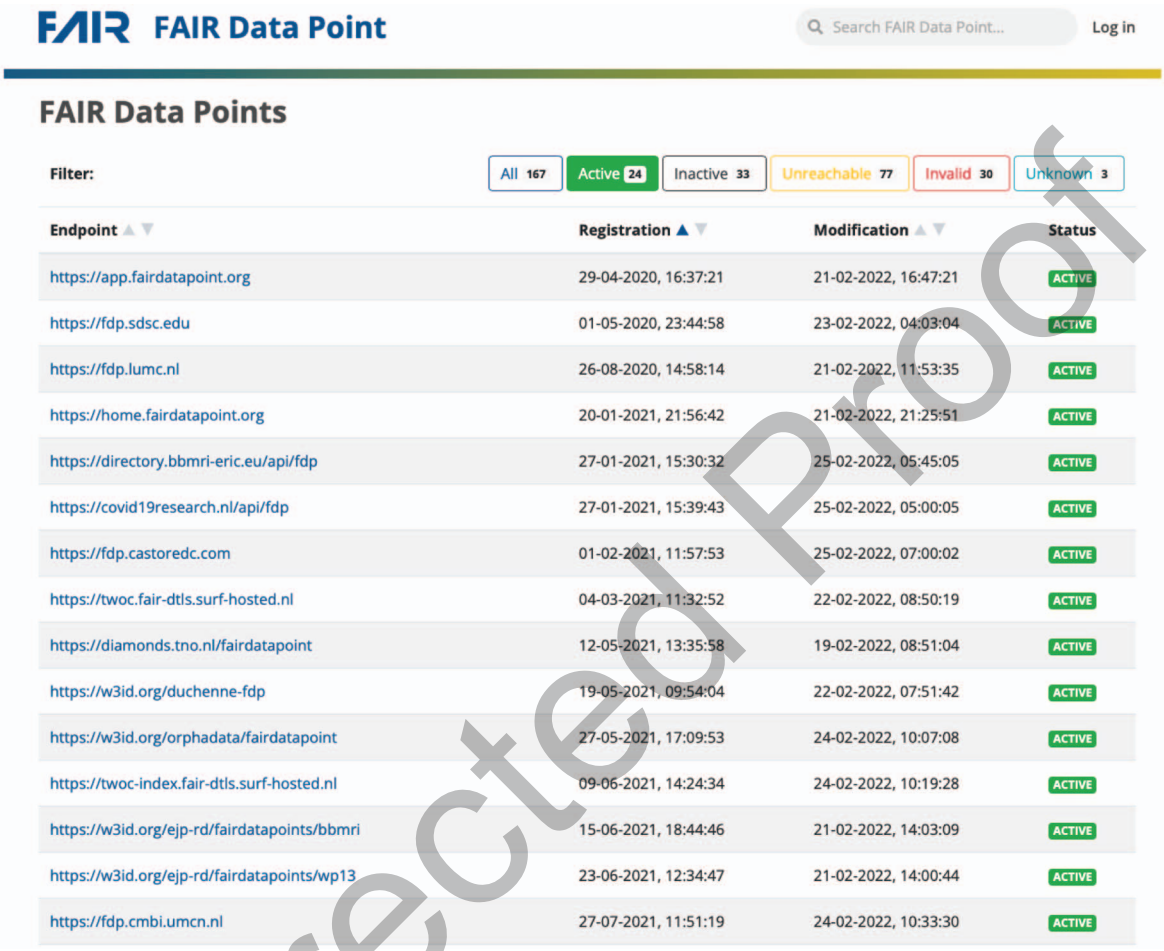


Figure 3. The FDP Index demonstration deployment.

6. REFERENCE IMPLEMENTATION

The motivations for working on the development of the FDP reference implementation were: *(i)* have a concrete implementation of the FDP specifications so that people could more easily understand the proposed approach, *(ii)* offer an exemplar implementation to help guiding developers in their implementations and, *(iii)* allow us to validate the specifications against a concrete implementation and, whenever necessary, adjust the design.

Figure 5 depicts the general architecture of our FDP reference implementation. We have divided it in two main components, namely, the FDP Server [12] and the FDPWeb Client [13]. In the FDP Server we have the HATEOAS REST API that the FDP Client and any other third-party client application can use to interact with the FDP. When most applications interact with the FDP through its URL, the API is used,

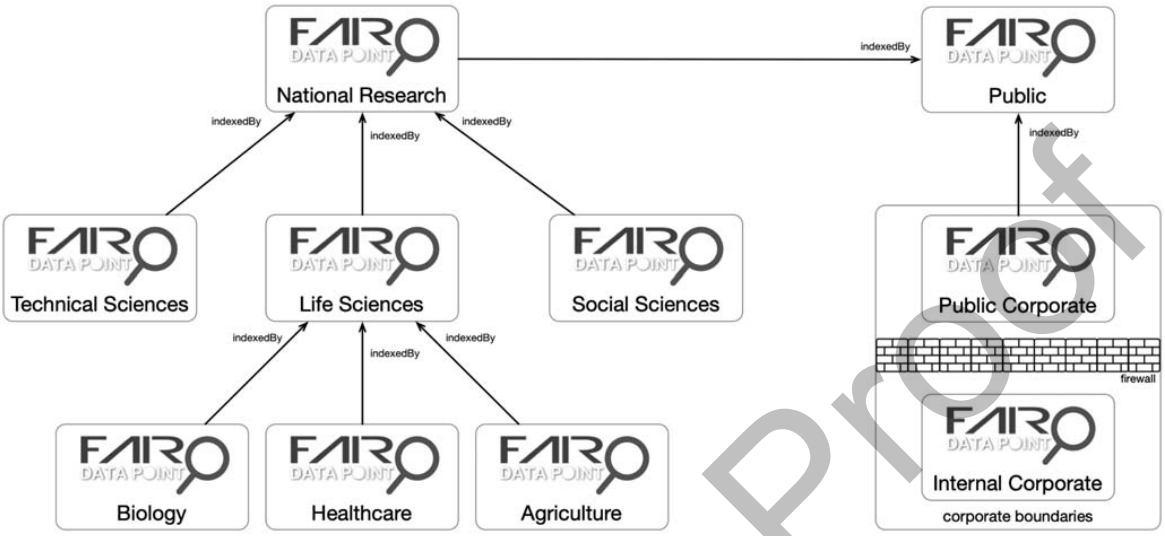


Figure 4. Example configurations for FDP Indexes.

except when a Web browser is used. In this case the FDP Web Client takes over the interaction and provides a human-oriented graphical user interface.

The FDP reference implementation offers support for activities such as add, edit and read metadata records and metadata schemata, user management and search. By default, if an user or a client application is not authenticated, the FDP content is read-only. We have defined two types of users, *administrator* and *regular user*. Once authenticated, a regular user is able to add and edit metadata content in the FDP. This user type is intended to be used by people responsible for populating a given FDP with metadata content. The administrator has other capabilities, including creating new users and configuring the FDP. Figure 6 presents a screenshot of the FDP Web Client initial page showing also the menu available for an authenticated administrator user. The screenshot also shows the + *Create* button, allowing the user to add new metadata records, in this case, of catalogs.

The reference implementation is constantly evolving with, normally, monthly new releases bringing new features, bug fixes, stability and usability improvements.

7. CONCLUSIONS

In this paper we reported the latest status of the design and development of the FAIR Data Point, a solution to facilitate the publication of machine-actionable metadata following the FAIR principles. Regarding the FAIR principles on findability, the FDP supports the attribution of globally unique and persistent identifiers for both metadata records and the digital objects they describe (principle F1) and the publication of rich metadata describing digital objects (principle F2). The FDP ontology defines the predicate *isMetadataOf* that allows metadata records to clearly and explicitly include the identifier of the digital

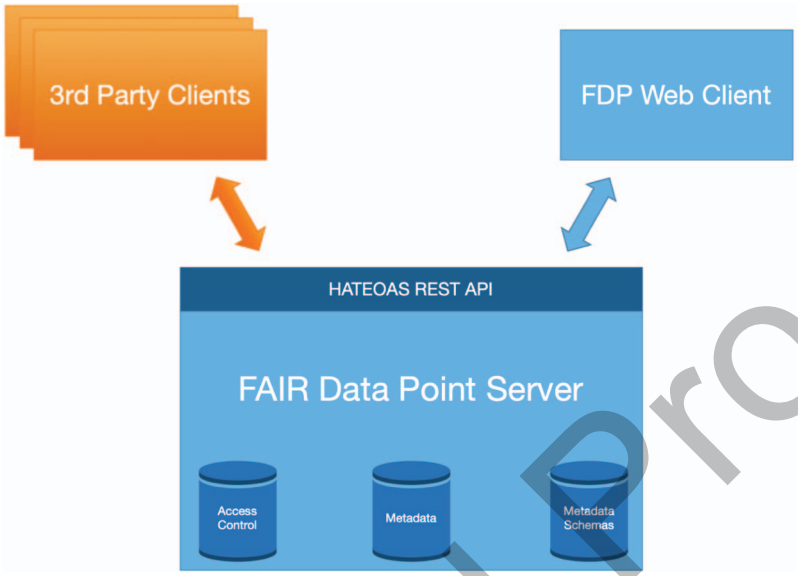


Figure 5. General architecture of the FDP reference implementation.

objects they describe (principle F3). The FDP Index service registers and indexes the metadata content of FDPs, satisfying the FAIR principle F4.

A given FDP answers to requests on identifiers of the metadata records published through it using the standardised HTTP protocol (principle A1). Moreover, the metadata consumer can navigate through the FDP's metadata structure following information contained in each metadata record using the standardised W3C's Linked Data Platform. The FAIR principle A2, which asks for the metadata to be accessible even when the data are no longer available is also supported by the FDP as the data steward responsible for the curation of the FDP content can keep metadata records of digital objects that are non-longer available.

The interoperability principles are also supported for metadata by the FDP as the metadata records are presented as RDF (principle I1), and widely used vocabularies make-up the metadata such as DCAT, FOAF and Dublin Core Terms (principle I2); moreover, the metadata records has qualified references to other metadata records (e.g., the navigation instructions using LDP),and to ontological annotations using external ontologies. Finally, they also explicitly point to the digital objects they describe (e.g., the *isMetadataOf* predicate) (principle I3).

Regarding the FAIR reusability principles, FDPs offer a degree of support by including usage license (principle R1.1) and authorship (principle R1.2) in the mandatory metadata schemas for Metadata Service and Catalog. The metadata records are based on DCAT and some related vocabularies (e.g., FOAF and DC Terms), which can be considered standards in the metadata community (principle R1.3). However, additional adherence to the FAIR principles, e.g., richer provenance information and community standards, rely on the data stewards and metadata providers.

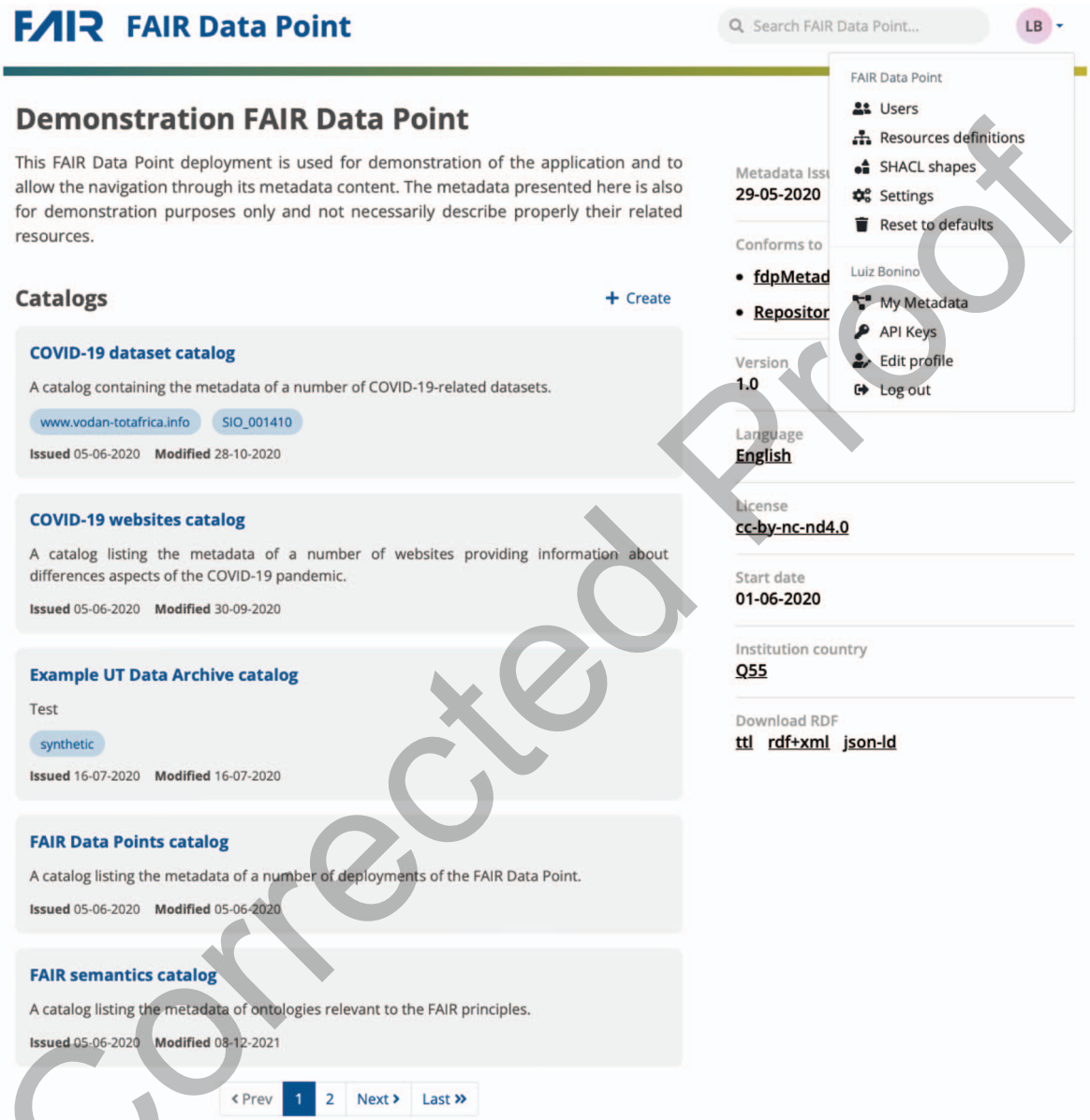


Figure 6. FDP Web Client screenshot with authenticated user.

This focus of the work on the FDP has been on the improvements on the metadata provisioning mechanism. This led to the design of the metadata structure and navigation. The FDP supports flexibility in the metadata structure, allowing users to define their own metadata schemata. However, in the current implementation, this requires a knowledge of SHACL that may not be widely available. To alleviate this specific knowledge requirement, in the FDP reference implementation we are currently working on a metadata schema editor that allows users to define their schemata in a more visual way and the tool automatically generates the SHACL descriptions. Another upcoming feature of the reference implementation is the possibility to publish the metadata of metadata schemas and allow users to import metadata schemas published in other FDPs. The goal of this feature is to foster the reuse of metadata schemas, specially the ones agreed upon by communities.

Another aspect of the FDP reference implementation that requires significant improvements is the access control. Data security is a permanent concern in many domains and applications. Although in most cases metadata is considered less sensitive, in some situations access restrictions may need to be imposed also to metadata. In order to tackle this issue, metadata access control is in our roadmap of new features for the FDP reference implementation.

ACKNOWLEDGMENTS

The work on design, architecture and implementation of the FDP has been conducted in the scope of a variety of projects in the last 7 years. Currently, LOBSS, RK and KB are partially funded from the EU's Horizon 2020 project FAIRsFAIR (grant number 831558). MDW and RK are funded via the EU's Horizon 2020 research and innovation programme under the EJP RD COFUND-EJP #825575.

AUTHOR CONTRIBUTIONS

Luiz Olavo Bonino da Silva Santos (l.o.boninodasilvasantos@utwente.nl) is the FDP lead architect and designer. Kees Burger (c.a.burger@lumc.nl) is the development lead of the FDP reference implementation and contributed in its design and architecture. Rajaram Kaliyaperumal (r.kaliyaperumal@lumc.nl) contributed in the design, architecture, semantic modeling and the FDP ontology. Mark D Wilkinson (mark.wilkinson@upm.es) authored early prototype versions of the DCAT+LDP combination of technologies, and contributed to the writing and editing of the manuscript.

DATA AVAILABILITY

The source code of the FDP reference implementation is openly available on the open-access GitHub repository, https://github.com/FAIRDataTeam/FAIRDataPoint, and is released under the MIT license. The FDP specifications are also openly and freely available at https://specs.fairdatapoint.org.

REFERENCES

[1] Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J., et al.: The fair guiding principles for scientific data management and stewardship. Scientific Data 3(1), 160018 (2016)

[2] Bonino Da Silva Santos, L.O., Wilkinson, M.D., Kuzniar, A., et al.: FAIR Data Points Supporting Big Data Interoperability, pp. 270–279. iSTE Press (2016)

[3] Archimate 3.1 specification. The open group standard, The Open Group, November 2019. Available at: https://publications.opengroup.org/standards/archimate

[4] Bonino da Silva Santos, L.O., Burger, K., Kaliyaperumal, R.: Fair data point ontology. Technical report (2022). Available at: https://w3id.org/fdp/fdp-o

[5] Lagoze, C., Van de Sompel, H., Nelson, M., Warner, S.: The open archives initiative protocol for metadata harvesting—version 2.0. Technical report, The Open Archives (January 2015). Available at: http://www.openarchives.org/OAI/openarchivesprotocol.html

[6] Albertoni, R., Browning, D., Cox, S., et al.: Data catalog vocabulary (dcat)—version 2. W3C recommendation, W3C (February 2020). Available at: https://www.w3.org/TR/vocab-dcat-2/

[7] Bonino da Silva Santos, L.O., Burger, K., Kaliyaperumal, R.: Fair data point specifications. Technical report (August 2021). Available at: https://specs.fairdatapoint.org

[8] Speicher, S., Arwe, J., Malhotra, A.: Linked data platform 1.0. W3C recommendation, W3C (February 2015). Available at: https://www.w3.org/TR/ldp/

[9] W3C OWL Working Group. Owl 2 web ontology language document overview. W3C recommendation, W3C (December 2012). Available at: https://www.w3.org/TR/owl2-overview/

[10] Knublauch, H., Kontokostas, D.: Shapes constraint language (shacl). W3C recommendation, W3C (July 2017). Available at: https://www.w3.org/TR/shacl/

[11] Atkinson, R., Car, N.J.: The profiles vocabulary—w3c working group note. W3C technical report, W3C (December 2019). Available at: https://www.w3.org/TR/dx-prof/

[12] Fair data point server (2022). Available at: https://github.com/FAIRDataTeam/FAIRDataPoint

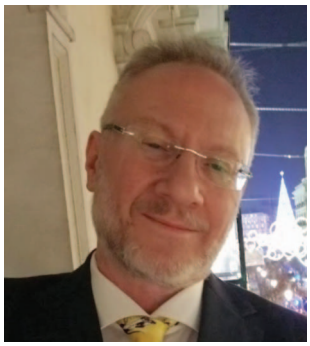
[13] Fair data point client (2022). Available at: https://github.com/FAIRDataTeam/FAIRDataPoint-client

AUTHOR BIOGRAPHY

Luiz Olavo Bonino da Silva Santos, PhD, is Associate Professor of the Services and CyberSecurity group at the University of Twente and Associate Professor of the BioSemantics group at the Leiden University Medical Centre. Luiz has a background in ontology-driven conceptual modeling, semantic interoperability, service-oriented computing, requirements engineering and context-aware computing. In the past 8 years Luiz has worked on designing and developing technologies, methods and tools to support making, publishing, indexing, searching and annotating FAIR (meta)data.
ORCID: 0000-0002-1164-1351

Kees Burger is a software engineer at the Leiden University Medical Center. He is leading the development of FAIR infrastructure reference implementations, and has been involved with FAIR developments since the inception of FAIR.
ORCID: 0000-0002-5437-779X

Rajaram Kaliyaperumal was born in Pondicherry, India. He received a B. Tech degree in Biomedical Engineering from Pondicherry University, India in 2008 and an M.Sc degree in Biomedical Engineering from Linköping University, Sweden in 2011. In 2012 he joined the department of Computer and Information Science, Linköping University as a software engineer. During this time, he developed methods and tools to align and repair ontologies. In 2013 he joined the Biosemantics group, Leiden, in the Netherlands as a software developer. His current research activities include investigating the use of semantic web technology in the context of FAIR data and developing prototypes to demonstrate generating and the use of FAIR data.
ORCID: 0000-0002-1215-167X



Mark D. Wilkinson, Ph.D., is Issac Peral Senior Investigator at the Center for Plant Biotechnology and Genomics of the Universidad Politécnica de Madrid, Spain. Mark was co-author on the original article describing the FAIR Data Principles, and continues to lead a research laboratory focused on the implementation of FAIR, and maximizing the exploitation of FAIR data.
ORCID: 0000-0001-6960-357X