Evaluating FAIR Digital Object and Linked Data as distributed object systems

Soiland-Reyes, S.; Goble, C.; Groth, P.

DOI
10.48550/arXiv.2306.07436

Publication date
2023

Document Version
Other version

License
CC BY

Citation for published version (APA):

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
Evaluating FAIR Digital Object and Linked Data as distributed object systems

Stian Soiland-Reyes¹,², Carole Goble¹, and Paul Groth²

¹Department of Computer Science, The University of Manchester, UK
²Informatics Institute, Faculty of Science, University of Amsterdam, NL

ABSTRACT

FAIR Digital Object (FDO) is an emerging concept that is highlighted by European Open Science Cloud (EOSC) as a potential candidate for building an ecosystem of machine-actionable research outputs. In this work we systematically evaluate FDO and its implementations as a global distributed object system, by using five different conceptual frameworks that cover interoperability, middleware, FAIR principles, EOSC requirements and FDO guidelines themselves. We compare the FDO approach with established Linked Data practices and the existing Web architecture, and provide a brief history of the Semantic Web while discussing why these technologies may have been difficult to adopt for FDO purposes. We conclude with recommendations for both Linked Data and FDO communities to further their adaptation and alignment.

INTRODUCTION

The FAIR principles (Mark D. Wilkinson et al. 2016) encourage sharing of scientific data with machine-readable metadata and the use of interoperable formats, and are being adapted by a wide range of research infrastructures. They have been recognised by the research community and policy makers as a goal to strive for (European Commission 2016). In particular, the European Open Science Cloud (EOSC) has promoted adaptation of FAIR data sharing of data resources across electronic research infrastructures (Mons et al. 2017). The EOSC Interoperability Framework (Corcho, Eriksson, et al. 2021) puts particular emphasis on how interoperability can be achieved technically, semantically, organisationally, and legally – laying out a vision of how data, publication, software and services can work together to form an ecosystem of rich digital objects.

Specifically, the EOSC Interoperability framework highlights the emerging FAIR Digital Object (FDO) concept (Schultes and Wittenburg 2019) as a possible foundation for building a semantically interoperable ecosystem to fully realise the FAIR principles beyond individual repositories and infrastructures. The FDO approach has great potential, as it proposes strong requirements for identifiers, types, access and formalises interactive operations on objects.

In other discourse, Linked Data (Bizer et al. 2009) has been seen as an established set of principles based on Semantic Web technologies that can achieve the vision of the FAIR principles (Bonino Da Silva Santos et al. 2016; Hasnain and Rebholz-Schuhmann 2018). Yet regular researchers and developers of emerging platforms for computation and data management are reluctant to adapt such a “FAIR Linked Data approach” fully (Verborgh and Vander Sande 2020), opting instead for custom in-house models and JSON-derived formats from RESTful Web services (Meroño-Peñuela et al. 2021a; Neumann et al. 2021). While such focus on simplicity allows for rapid development and highly specialised services, it raises wider concerns about interoperability (Turcoane 2014; S. R. Wilkinson et al. 2022).

One challenge that may, perhaps counter-intuitively, steer developers towards a not-invented-here mentality (Stefi 2015; Stefi and Hess 2015) when exposing their data on the Web is the heterogeneity and apparent complexity of Semantic Web approaches themselves (Meroño-Peñuela et al. 2021b). These approaches – FDO and Linked Data – thus, form two of the major avenues for allowing developers and the wider research community to achieve the goal of FAIR data. Given their importance,
in this article, we aim to compare FAIR Digital Objects with Linked Data and the Web architecture in the context of the discourse around FAIR data.

Concretely, the contribution of this paper is a **systematic comparison between FDO and Linked Data using 5 different conceptual frameworks** that capture different perspectives on interoperability and readiness for implementation.

**BACKGROUND AND RELATED WORK**

In the following, we discuss the related work with respect to FAIR Digital Objects and Linked Data. We do so by looking through the lens of development of these technologies over time, including future directions.

**FAIR Digital Object**

The concept of **FAIR Digital Objects** (Schultes and Wittenburg 2019) has been introduced as a way to expose research data as active objects that conform to the FAIR principles (Mark D. Wilkinson et al. 2016). This builds on the **Digital Object** (DO) concept (Kahn and Wilensky 2006), first introduced by Kahn and Wilensky (1995) as a system of **repositories** containing **digital objects** identified by **handles** and described by **metadata** which may have references to other handles. DO was the inspiration for the **ITU-T X.1255** (2013) framework which introduced an abstract **Digital Entity Interface Protocol** for managing such objects programmatically, first realised by the Digital Object Interface Protocol (DOIP) (Reilly 2009).

In brief, the structure of a FAIR Digital Object (FDO) is to, given a **persistent identifier** (PID) such as a DOI, resolve to a **PID Record** that gives the object a type along with a mechanism to retrieve its **bit sequences**, **metadata** and references to further programmatic **operations**. The type of an FDO (itself an FDO) defines attributes to semantically describe and relate such FDOs to other concepts (typically other FDOs referenced by PIDs). The premise of systematically building an ecosystem of such digital objects is to give researchers a way to organise complex digital entities, associated with identifiers, metadata, and supporting automated processing (Wittenburg, Strawn, et al. 2019).

Recently, FDOs have been recognised by the European Open Science Cloud (EOSC) as a suggested part of its Interoperability Framework (Corcho, Eriksson, et al. 2021), in particular for deploying active and interoperable FAIR resources that are **machine actionable**. Development of the FDO concept continued within Research Data Alliance (RDA) groups and EOSC projects like GO-FAIR, concluding with a set of guidelines for implementing FDO (Bonino et al. 2019). The FAIR Digital Objects Forum has since taken over the maturing of FDO through focused working groups which have currently drafted several more detailed specification documents (see **Next steps for FDO** on the following page).

**FDO approaches**

FDO is an evolving concept. A set of FDO Demonstrators (Wittenburg, Anders, et al. 2022) highlight how current adapters are approaching implementations of FDO from different angles:

- Building on the Digital Object concept, using the simplified **DOIPv2.0** (2018) specification, which detail how to exchange JSON objects through a text-based protocol\(^1\) (usually TCP/IP over TLS). The main DOIP operations are retrieving, creating and updating digital objects. These are mostly realised using the reference implementation Cordra (Tupelo-Schneck, Robert and Lannom 2022). FDO types are registered in the local Cordra instance, where they are specified using JSON Schema (Wright et al. 2022) and PIDs are assigned using the Handle system. Several type registries have been established.
- Following a Linked Data approach, but using the DOIP protocol, e.g. using JSON-LD and schema.org within DOIP in Material Sciences archives (Riccardi et al. 2022).
- Approaching the FDO principles from existing Linked Data practices on the Web, e.g. Work-flowHub use of RO-Crate and schema.org (Soiland-Reyes, Sefton, et al. 2022).

From this it becomes apparent that there is a potentially large overlap between the goals and approaches of FAIR Digital Objects and Linked Data, which we will cover on page 4.

---

\(^1\)For a brief introduction to DOIP 2.0, see CNRI (2023a)
**Next steps for FDO**

The FAIR Digital Object Forum (FDO 2022) working groups have prepared detailed requirement documents (FDO Specs 2022) setting out the path for realising FDOs, named FDO Recommendations. As of 2023-06-17, most of these documents are open for public review, while some are still in draft stages for internal review. As these documents clarify the future aims and focus of FAIR Digital Objects (Lannom, Schwardmann, Christophe Blanchi, et al. 2022), we provide a brief summary of each:

**FAIR Digital Object Overview and Specifications** (FDO-Overview-PEN-2.0) is a comprehensive overview of FAIR Digital Object specifications listed below. It serves as a primer that introduces FDO concepts and the remaining documents. It is accompanied by an FDO Glossary (Broeder and Wittenburg 2022).

The **FDO Forum Document Standards** (WD-DocProcessStd-1.1) documents the recommendation process within the forum, starting at Working Draft (WD) status within the closed working group and later within the open forum, then Proposed Recommendation (PR) published for public review, finalised as FDO Forum Recommendation (REC) following any revisions. In addition, the forum may choose to endorse existing third-party notes and specifications.

The **FDO Requirement Specifications** (PR-RequirementSpec-3.0) is an update of (Bonino et al. 2019) as the foundational definition of FDO. This sets the criteria for classifying an digital entity as a FAIR Digital Object, allowing for multiple implementations. The requirements shown in Table 3 on page 11 are largely equivalent, but in this specification clarified with references to other FDO documents.

The **Machine actionability** (PR-MachineActionDef-2.2) sets out to define what is meant by *machine actionability* for FDOs. *Machine readable* is defined as elements of bit-sequences defined by structural specification, *machine interpretable* elements that can be identified and related with semantic artefacts, while *machine actionable* are elements with a type with operations in a symbolic grammar. The document largely describes requirements for resolving an FDO to metadata, and how types should be related to possible operations.

**Configuration Types** (PR-ConfigurationTypes-2.1) classifies different granularities for organising FDOs in terms of PIDs, PID Records, Metadata and bit sequences, e.g. as a single FDO or several daisy-chained FDOs. Different patterns used by current DOIP deployments are considered, as well as FAIR Signposting (Van de Sompel, Klein, et al. 2022).

**PID Profiles & Attributes** (PR-PIDProfileAttributes-2.1) specifies that PIDs must be formally associated with a *PID Profile*, a separate FDO that defines attributes required and recommended by FDOs following said profile. This forms the *kernel attributes*, building on recommendations from RDA’s *PID Information Types* working group (Weigel et al. 2018). This document makes a clear distinction between a minimal set of attributes needed for PID resolution and FDO navigation, which needs to be part of the *PID Record* (Islam 2023), compared with a richer set of more specific attributes as part of the *metadata* for an FDO, possibly represented as a separate FDO.

**Kernel Attributes & Metadata** (PR-KernelAttributes-2.0) elaborates on categories of FDO Mandatory, FDO Optional and Community Attributes, recommending kernel attributes like *dateCreated*, *ScientificDomain*, *PersistencePolicy*, *digitalObjectMutability*, etc. This document expands on RDA Recommendation on PID Kernel Information (Weigel et al. 2018). It is worth noting that both documents are relatively abstract and do not establish PIDs or namespaces for the kernel attributes.

**Granularity, Versioning, Mutability** (PR-Granularity-2.2) considers how granularity decisions for forming FDOs must be agreed by different communities depending on their pragmatic usage requirements. The affect on versioning, mutability and changes to PIDs are considered, based on use cases and existing PID practices.

**DOIP Endorsement Request** (PED-DOIPEndorsement-1.1) is an endorsement of the DOIP v2.0 (DOIPV2.0 2018) specification as a potential FDO implementation, as it has been applied by several institutions (Wittenburg, Anders, et al. 2022). The document proposes that DOIP shall be assessed for completeness against FDO – in this initial draft this is justified as “we can state that DOIP is compliant with the FDO specification documents in process” (the documents listed above).

**Upload of FDO** (PEN-FDO-Upload) illustrates the operations for uploading an FDO to a repository, what checks it should do (for instance conformance with the PID Profile, if PIDs resolve). ResourceSync (ANSI Z39.99 2017) is suggested as one type of service to list FDOs. This document highlights potential practices by repositories and their clients, without adding any particular requirements.
Typing FAIR Digital Objects (PR-TypingFDOs-2.0) defines what type means for FDOs, primarily to enable machine actionability and to define an FDO’s purpose. This document lays out requirements for how FDO Types should themselves be specified as FDOs, and how an FDO Type Framework allows organising and locating types. Operations applicable to an FDO is not predefined for a type, however operations naturally will require certain FDO types to work. How to define such FDO operations is not specified.

Implementation of Attributes, Types, Profiles and Registries (WD-ImplAttributesTypesProfiles) details how to establish FDO registries for types and FDO profiles, with their association with PID systems. This document suggest policies and governance structures, together with guidelines for implementations, but without mandating any explicit technology choices. Differences in use of attributes are examplified using FDO PIDs for scientific instruments, and the proto-FDO approach of DARIAH-DE (Schwardmann and Kálmán 2022).

See bibliography on page 39 for the citation per document above. It is worth pointing out that, except for the DOIP endorsement, all of these documents are conceptual, in the sense that they permit any technical implementation of FDO, if used according to the recommendations. Existing FDO implementations (Wittenburg, Anders, et al. 2022) are thus not fully consolidated in choices such as protocols, type systems and serialisations – this divergence and corresponding additional technical requirements mean that FDOs are not yet in a single ecosystem.

From the Semantic Web to Linked Data
In order to describe Linked Data as it is used today, we’ll start with an (opinionated) description of the evolution of its foundation, the Semantic Web.

A brief history of the Semantic Web
The Semantic Web was developed as a vision by Tim Berners-Lee (Berners-Lee and Fischetti 1999), at a time that the Web had already become widely established for information exchange, being a global set of hypermedia documents which are cross-related using universal links in the form of URLs. The foundations of the Web (e.g. URLs, HTTP, SSL/TLS, HTML, CSS, ECMAScript/JavaScript, media types) were standardised by W3C, Ecma, IETF and later WHATWG. The goal of Semantic Web was to further develop the machine-readable aspects of the Web, in particular adding meaning (or semantics) to not just the link relations, but also to the resources that the URLs identified, and for machines thus being able to meaningfully navigate across such resources, e.g. to answer a particular query.

Through W3C, the Semantic Web was realised with the Resource Description Framework (RDF) (Schreiber and Raimond 2014) that used triples of subject-predicate-object statements, with its initial serialisation format (Lassila and Swick 1999) being RDF/XML (XML was at the time seen as a natural data-focused evolution from the document-centric SGML and HTML).

While triple-based knowledge representations were not new (Stanczyk 1987), the main innovation of RDF was the use of global identifiers in the form of URIs2 as the primary identifier of the subject (what the statement is about), predicate (relation/attribute of the subject) and object (what is pointed to). By using URIs not just for documents3, the Semantic Web builds a self-described system of types and properties, where the meaning of a relation can be resolved by following its hyperlink to the definition within a vocabulary. By applying these principles as well to any kind of resource that could be described at a URL, this then forms a global distributed Semantic Web.

The early days of the Semantic Web saw fairly lightweight approaches with the establishment of vocabularies such as FOAF (to describe people and their affiliations) and Dublin Core (for bibliographic data). Vocabularies themselves were formalised using RDFs or simply as human-readable HTML web pages defining each term. The main approach of this Web of Data was that a URI identified a resource (e.g. an author) with a HTML representation for human readers, along with a RDF representation for

---

2URIs (Berners-Lee, Roy T. Fielding, et al. 2005) are generalised forms of URLs that include locator-less identifiers such as ISBN book numbers (URNs). The distinction between locator-full and locator-less identifiers have weakened in recent years (OCLC 2010), for instance DOI identifiers now are commonly expressed with the prefix https://doi.org/ rather than as URNs with info:doi: given that the URL/URN gap has been bridged by HTTP resolvers and the use of Persistent Identifiers (PIDs) (Juty et al. 2011). RDF 1.1 formats use Unicode to support IRIs (Dürst and Suignard 2005), which extends URIs to include international characters and domain names.

3URIs can also identify non-information resources for any kind of physical object (e.g. people), such identifiers can resolve with 303 See Other redirections to a corresponding information resources (Sauermann et al. 2008).

4/39
machine-readable data of the same resource. By using content negotiation in HTTP, the same identifier could be used in both views, avoiding index.html vs index.rdf exposure in the URLs. The concept of namespaces gave a way to give a group of RDF resources with the same URI base from a Semantic Web-aware service a common prefix, avoiding repeated long URLs.

The mid-2000s saw large academic interest and growth of the Semantic Web, with the development of more formal representation system for ontologies, such as OWL (W3C OWL Working Group 2012), allowing complex class hierarchies and logic inference rules following open world paradigm. More human-readable syntaxes for RDF such as Turtle evolved at this time, and conferences such as ISWC (Horrocks and Hendler 2002) gained traction, with a large interest in knowledge representation and logic systems based on Semantic Web technologies evolving at the same time.

Established Semantic Web services and standards include: SPARQL (SPARQL WG 2013) (pattern-based triple queries), named graphs (Wood et al. 2014) (triples expanded to quads to indicate statement source or represent conflicting views), triple/quad stores (graph databases such as OpenLink Virtuoso, GraphDB, 4Store), mature RDF libraries (including Redland RDF, Apache Jena, Eclipse RDF4J, RDFLib, RDF.rb, rdflib.js), and graph visualisation.

RDF is one way to implement knowledge graphs, a system of named edges and nodes (Nurdiati and Hoede 2008), which when used to represent a sufficiently detailed model of the world, can then be queried and processed to answer detailed research questions. The creation of RDF-based knowledge graphs grew particularly in fields like bioinformatics, e.g. for describing genomes and proteins (Goble and Stevens 2008; Williams et al. 2012). In theory, the use of RDF by the life sciences would enable interoperability between the many data repositories and support combined views of the many aspects of bio-entities – however in practice most institutions ended up making their own ontologies and identifiers, for what to the untrained eye would mean roughly the same. One can argue that the toll of adding the semantic logic system of rich ontologies meant that small, but fundamental, differences in opinion (e.g. should a gene identifier signify just the particular DNA sequence letters, or those letters as they appear in a particular position on a human chromosome?) lead to large differences in representational granularity, and thus needed different identifiers.

Facing these challenges, thanks to the use of universal identifiers in the form of URIs, mappings could retrospectively be developed not just between resources, but also across vocabularies. Such mappings can be expressed themselves using lightweight and flexible RDF vocabularies such as SKOS (Isaac and Summers 2009) (e.g. dct:title skos:closeMatch schema:name to indicate near equivalence of two properties). Exemplifying the need for such cross-references, automated ontology mappings have identified large potential overlaps like 372 definitions of Person (Hu et al. 2011).

The move towards Open Science data sharing practices did from the late 2000s encourage knowledge providers to distribute collections of RDF descriptions as downloadable datasets, so that their clients can avoid thousands of HTTP requests for individual resources. This enabled local processing, mapping and data integration across datasets (e.g. Open PHACTS (Groth et al. 2014)), rather than relying on the providers’ RDF and SPARQL endpoints (which could become overloaded when handling many concurrent, complex queries).

With these trends, an emerging problem was that adopters of the Semantic Web primarily utilised it as a set of graph technologies, with little consideration to existing Web resources. This meant that links stayed mainly within a single information system, with little URI reuse even with large term overlaps (Kamdar et al. 2017). Just like link rot affect regular Web pages and their citations from scholarly communication (Klein et al. 2014), for a majority of described RDF resources in the Linked Open Data (LOD) Cloud’s gathering of more than thousand datasets, unfortunately do not actually link to (still) downloadable (dereferenceable) Linked Data (Polleres et al. 2020). Another challenge facing potential adopters is the plethora of choices, not just to navigate, understand and select to reuse the many possible vocabularies and ontologies (Carriero et al. 2020), but also technological choices on RDF serialisation (at least 7 formats), type system (RDFS (Guha and Brickley 2014), OWL (W3C OWL Working Group 2012), OBO (Tirimizi et al. 2011), SKOS (Isaac and Summers 2009)), and deployment challenges (Sauer-mann et al. 2008) (e.g. hash vs slash in namespaces, HTTP status codes and PID redirection strategies).

---

5In RDF, each triple represent an edge that is named using its property URI, and the nodes are subject/object as URIs, blank nodes or (for objects) typed literal values (Schreiber and Raimond 2014).
6Datasets that distribute RDF graphs should not be confused with RDF Datasets used for partitioning named graphs.
Linked Data: Rebuilding the Web of Data

The Linked Data (LD) concept (Bizer et al. 2009) was kickstarted as a set of best practices (Berners-Lee 2006) to bring the Web aspect of the Semantic Web back into focus. Crucial to Linked Data is the reuse of existing URIs, rather than making new identifiers. This means a loosening of the semantic restrictions previously applied, and an emphasis on building navigable data resources, rather than elaborate graph representations.

Vocabularies like schema.org evolved not long after, intended for lightweight semantic markup of existing Web pages, primarily to improve search engines' understanding of types and embedded data. In addition to several such embedded microformats (OGP 2022; Sporny, Herman, et al. 2015; Microdata 2023), we find JSON-LD (Sporny, Longley, et al. 2020) as a Web-focused RDF serialisation that aims for improved programmatic generation and consumption, including from Web applications. JSON-LD is as of 2023-05-18 used by 45% of the top 10 million websites (W3Techs 2023).

Recently there has been a renewed emphasis to improve the Developer Experience (Verborgh 2018) for consumption of Linked Data, for instance RDF Shapes – expressed in SHACL (Kontokostas and Knublauch 2017) or ShEx (Baker and Prud’hommeaux 2019) – can be used to validate RDF Data (Gayo et al. 2017; Thornton et al. 2019) before consuming it programatically, or reshaping data to fit other models. While a varied set of tools for Linked Data consumptions have been identified, most of them still require developers to gain significant knowledge of the underlying Semantic Web technologies, which hampers adaption by non-LD experts (Klimek et al. 2019), which then tend to prefer non-semantic two-dimensional formats such as CSV files.

A valid concern is that the Semantic Web research community has still not fully embraced the Web, and that the “final 20%” engineering effort is frequently overlooked in favour of chasing new trends such as Big Data and AI, rather than making powerful Linked Data technologies available to the wider groups of Web developers (Verborgh and Vander Sande 2020). One bridging gap here by the Linked Data movement has been “Linked Data by stealth” approaches such as structured data entry spreadsheets powered by ontologies (Katy Wolstencroft et al. 2011), the use of Linked Data as part of REST Web APIs (Page et al. 2011), and as shown by the big uptake by publishers to annotate the Web using schema.org (Bernstein et al. 2016), with vocabulary use patterns documented by copy-pastable JSON-LD examples, rather than by formalised ontologies or developer requirements to understand the full Semantic Web stack.

Linked Data provides technologies that have evolved over time to satisfy its primary purpose of data interoperability. The needs to embrace the Web and developer experience have been central lessons learned. In contrast, FDO is a new approach with many different potential paths forward, and having a partial overlap with the aims of Linked Data.

METHOD

Our main motivation for this article is to investigate how FAIR Digital Objects may differ from the learnt experiences of Linked Data and the Web. We also aim to reflect back from FDO’s motivation of machine-actionability to consider the Web as a distributed computational system.

To better understand the relationship between the FDO framework and other existing approaches, we use the following for analysis:

1. An Interoperability Framework and Distributed Platform for Fast Data Applications (Delgado 2016), which proposes quality measurements for comparing how frameworks support interoperability, particularly from a service architectural view.
2. The FAIR Digital Object guidelines (Bonino et al. 2019), validated against its current implementations for completeness.
3. A Comparison Framework for Middleware Infrastructures (Zarras 2004), which suggest dimensions like openness, performance and transparency, mainly focused on remote computational methods.
4. Cross-checks against RDA’s FAIR Data Maturity Model (Bahim et al. 2020) to find how the FAIR principles are achieved in FDO, in particular considering access, sharing and openness.

---

Presumably this large uptake of JSON-LD is mainly for the purpose of Search Engine Optimisation (SEO), with typically small amounts of metadata which may not constitute Linked Data as introduced above, however this deployment nevertheless constitute machine-actionable structured data.
5. EOSC Interoperability Framework (Corcho, Eriksson, et al. 2021) which gives recommendations for technical, semantic, organisational and legal interoperability, particularly from a metadata perspective.

The reason for this wide-ranging comparison is to exercise the different dimensions that together form FAIR Digital Objects: Data, Metadata, Service, Access, Operations, Computation. We have left out further considerations on type systems, persistent identifiers and social aspects as principles and practices within these dimensions are still taking form within the FDO community (as detailed on page 3).

Some of these frameworks invite a comparison on a conceptual level, while others relate better to implementations and current practices. For conceptual comparisons we consider FAIR Digital Objects and the Web broadly. For implementations, we contrast the main FDO realisation using the DOIPv2 protocol (DOIPv2.0 2018) against Linked Data as implemented in general practice.8

**Considering FDO/Web as interoperability framework for Fast Data**

The Interoperability Framework for Fast Data Applications (Delgado 2016) categorises interoperability between applications along 6 strands, covering different architectural levels: from *symbiotic* (agreement to cooperate) and *pragmatic* (ability to choreograph processes), through *semantic* (common understanding) and *syntactic* (common message formats), to low-level *connective* (transport-level) and *environmental* (deployment practices).

We have chosen to investigate using this framework as it covers the higher levels of the OSI Model (Stallings 1990) better with regards to automated machine-to-machine interaction (and thus interoperability), which is a crucial aspect of the FAIR principles. In Table 1 on the next page we use the interoperability framework to compare the current FAIR Digital Object approach against the Web and its Linked Data practices.

Based on the analysis shown in Table 1, we draw the following conclusions:

The Web has already showed us how one can compose workflows of heterogeneous Web Services (Katherine Wolstencroft et al. 2013). However, this is mostly done via developer or human interaction (Lamprecht et al. 2021). Similarly, FDO does not enable automatic composition because operation semantics are not well defined. There is a question as to whether the extensive documentation and broad developer usage that is available for Web APIs could potentially be utilised for FDO.

A difference between Web technologies and FDO is the stringency of the requirements for both syntax and semantics. Whereas the Web allows many different syntactic formats (e.g. from HTML to XML, PDFs), FDO realised with DOIP requires JSON. On the semantic front, FDO mandates that every object have a well-defined type and structured form. This is clearly not the case on the Web.

In terms of connectivity and the deployment of applications, the Web has a plethora of software, services, and protocols that are widely deployed. These have shown interoperability. The Web standards bodies (e.g. IETF and W3C) follow the OpenStand principles (OpenStand 2017) to embrace openness, transparency, and broad consensus. In contrast, FDO has a small number of implementations and corresponding protocols, although with a growing community, as evidenced at the first international FDO conference (Loo 2022). This is not to say that it is not worth developing further Handle+DOIP implementations in the future, but we note that the current FDO functionality can easily be implemented using Web technologies, even as DOIP-over-HTTP (CNRI 2023b).

It is also a question as to whether a highly constrained protocol revolving around persistent identifiers is in fact necessary. For example, DOIs are mostly resolved on the web using HTTP redirects with the common https://doi.org/ prefix, hiding their Handle nature as an implementation detail (DOI 2019).

---

7For further background on FDO implemented with Linked Data see (Bonino da Silva Santos et al. 2022; Soiland-Reyes, Castro, et al. 2022)
Table 1. Considering FDO and Web according to the quality levels of the Interoperability Framework for Fast Data (Delgado 2016).

<table>
<thead>
<tr>
<th>Quality</th>
<th>FDO w/ DOIP</th>
<th>Web w/ Linked Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbiotic: to what extent multiple applications can agree to interact, align, collaborate or cooperate</td>
<td>The purpose of FDO is to enable federated machine actionable digital objects for scholarly purposes, in practice this also requires agreement of compatibility between FDO types. FDO encourages research communities to develop common type registries to be shared across instances. In current DOIP practice, each service have their own types, attributes and operations. The wider symbiosis is consistent use of PIDs.</td>
<td>“Follow your nose” crawler navigation, which may lead to frequent dead ends. Operational composition, typically within a single API provider, documented by OpenAPI 3 (D. Miller et al. 2021), schema.org Actions (Schema.Org Actions 2022), WSDL/SOAP (Liu and Booth 2007), but frequently just as human-readable developer documentation with examples.</td>
</tr>
<tr>
<td>Pragmatic: using interaction contracts so processes can be choreographed in workflows</td>
<td>FDO types and operations enable detailed choreography (Canonical Workflows; CWFR Group (2021)). 0.TYPE/DOIPOperation has lightweight definition of operation. 0.DOIP/Request or 0.DOIP/Response may give FDO Type or any other kind of “specifics” (incl. human readable docs). Semantics/purpose of operations not formalised (similar operations can be grouped with 0.DOIP/OperationReference).</td>
<td>Lightweight HTTP semantics for authenticity/navigation. Semantic Type not commonly expressed on PID/header level, may be declared within Linked Data metadata. Semantic of type implied by Linked Data formats (e.g. OWL2, RDFS), although choice of type system may not be explicit.</td>
</tr>
<tr>
<td>Semantic: ensuring consistent understanding of messages, interoperability of rules, knowledge and ontologies</td>
<td>FDO semantic enable navigation and typing. Every FDO has a type. Types maintained in FDO Type registries, which may add additional semantics, e.g. the ePIC PID-InfoType for Model. No single type semantic, Type FDOs can link to existing vocabularies &amp; ontologies. JSON-LD used within some FDO objects (e.g. DISSCO Digital Specimen, NIST Material Science schema) (Wittenburg, Anders, et al. 2022)</td>
<td>Textual HTTP headers (including any signposting), single byte stream of any media type, e.g. Linked Data formats (JSON-LD, Turtle, RDF/XML) or embedded in document (HTML with RDFa, JSON-LD or Microdata). XML was previously the main syntax used by APIs, JSON is now dominant.</td>
</tr>
<tr>
<td>Syntactic: serialising messages for digital exchange, structure representation</td>
<td>DOIP serialise FDOs as JSON, metadata commonly use JSON, typed with JSON Schema. Multiple byte stream attachments of any media type.</td>
<td></td>
</tr>
</tbody>
</table>
Quality

**Connective:**
Transferring messages to another application, e.g. wrapping in other protocols

**Environmental:**
How applications are deployed and affected by its environment, portability

---

**DOIPV2.0** (2018) is transport-independent, commonly TLS TCP/IP port 9000, DOIP over HTTP (CNRI 2023b)

Main DOIP implementation is *Cordra*, which can be single-instance or distributed. Cordra storage backends include file system, S3, MongoDB (itself scalable). Unique DOIP protocol can be hard to add to existing Web application frameworks, although proxy services have been developed (e.g. B2SHARE adapter).

**Web w/ Linked Data**

HTTP/1.1, TCP/IP port 80 (Roy T. Fielding, Gettys, et al. 1999); HTTP/1.1+TLS, TCP/IP 443 (Rescorla 2000); HTTP/2, as HTTP/1* but binary (Belshe et al. 2015); HTTP/3, like HTTP/2+TLS but UDP (Bishop 2022)

---

**Mapping of Metamodel concepts**

The Interoperability Framework for Fast Data also provides a brief *metamodel* which we use in Table 2 to map and exemplify corresponding concepts in FDO’s DOIP realization and the Web using HTTP semantics (Roy T. Fielding, Nottingham, et al. 2022).

From this mapping we can identify the conceptual similarities between DOIP and HTTP, often with common terminology. Notable are that neither DOIP or HTTP have strong support for transactions (explored further on page 14), as well that HTTP has poor direct support for processes, as the Web is primarily stateless by design.

**Table 2.** Mapping the Metamodel concepts from the Interoperability Framework for Fast Data (Delgado 2016) to equivalent concepts for FDO and Web.

<table>
<thead>
<tr>
<th>Metamodel concept</th>
<th>FDO/DOIP concept</th>
<th>Web/HTTP concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource</td>
<td>FDO/DO</td>
<td>Resource</td>
</tr>
<tr>
<td>Service</td>
<td>DOIP service</td>
<td>Server/endpoint</td>
</tr>
<tr>
<td>Transaction</td>
<td>(not supported)</td>
<td>Conditional requests, 409 Conflict (primarily stateless), 100 Continue, 202 Accepted</td>
</tr>
<tr>
<td>Process</td>
<td>Extended operations</td>
<td>Method, query parameters</td>
</tr>
<tr>
<td>Operation</td>
<td>DOIP Operation</td>
<td>Request</td>
</tr>
<tr>
<td>Request</td>
<td>DOIP Request</td>
<td>Response</td>
</tr>
<tr>
<td>Response</td>
<td>DOIP Response</td>
<td>Message, Representation</td>
</tr>
<tr>
<td>Message</td>
<td>Segment, requestId</td>
<td>TCP/IP, TLS, UDP</td>
</tr>
<tr>
<td>Channel</td>
<td>DOIP Transport protocol (e.g. TCP/IP, TLS). JSWS signatures</td>
<td>HTTP/1.1, HTTP/2, HTTP/3</td>
</tr>
<tr>
<td>Protocol</td>
<td>DOIP 2.0, ++</td>
<td>URL</td>
</tr>
<tr>
<td>Link</td>
<td>PID/Handle</td>
<td></td>
</tr>
</tbody>
</table>
Assessing FDO implementations

The FAIR Digital Object guidelines (Bonino et al. 2019) set out recommendations for FDO implementations. Note that the proposed update to FDO specification (PR-RequirementSpec-3.0) clarifies these definitions with equivalent identifiers\(^9\) and relates them to further FDO requirements such as FDO Data Type Registries.

In Table 3 on the following page we evaluate completeness of the guidelines in two current FDO realizations, using DOIPv2 (DOIPv2.0 2018) and using Linked Data Platform (Speicher et al. 2015), as proposed by Bonino da Silva Santos et al. (2022).

A key observation from this is that simply using DOIPv does not achieve many of the FDO guidelines. Rather the guidelines set out how a protocol like DOIPv should be used to achieve FAIR Digital Object goals. The DOIP Endorsement (PED-DOIPEndorsement-1.1) set out that to comply, DOIPv must be used according to the set of FDO requirement documents (details on page 3), and notes Achieving FDO compliance requires more than DOIPv and full compliance is thus left to system designers. Likewise, a Linked Data approach will need to follow the same requirements to actually comply as an FDO implementation.

From our evaluation, we can observe:

- G1 and G2 call for stability and trustworthiness. While the foundations of both DOIPv and Linked Data approaches are now well established – the FDO requirements and in particular how they can be implemented are still taking shape and subject to change.
- Machine actionability (G4, G6) is a core feature of both FDOs and Linked Data. Conceptually they differ in the way types and operations are discovered, with FDO seemingly more rigorous. In practice, however, we see that DOIPv also relies on dynamic discovery of operations and that operation expectations for types (FDOF7) have not yet been defined.
- FDO proposes that types can have additional operations beyond CRUD (FDOF5, FDOF6), while Linked Data mainly achieves this with RESTful patterns using CRUD on additional resources, e.g. order/152/items. These are mainly stylistics but affect the architectural view – FDOs have more of an object-oriented approach.
- FDO puts strong emphasis on the use of PIDs (FDOF1, FDOF2, FDOF3, FDOF5), but in current practice DOIPv use local types, local extended operations (FDOF5) and attributes (FDOF4) that are not bound to any global namespace.
- Linked Data have a strong emphasis on semantics (FDOF8), and metadata schemas developed by community agreements (FDOF10). FDO types need to be made reusable across servers.
- While FDO recommends nested metadata FDOs (FDOF8, FDOF9), in practice this is not found (or linked with custom keys), particularly due to lack of namespaces and the favouring of local types rather than type/property re-use. Linked Data frequently have multiple representations, but often not sufficiently linked (link relation alternate (Nottingham 2017)) or related (prov:specializationOf from Lebo, McGuinness, et al. (2013)).
- FDO collections are not yet defined for DOIPv, while Linked Data seemingly have too many alternatives. LDP has specific native support for containers.
- Tombstones for deleted resources are not well supported, nor specified, for either approach, although the continued availability of metadata when data is removed is a requirement for FAIR principles (see RDA-A2-01M in Table 5 on page 21).
- DOIPv supports multiple chunks of data for an object (FDOF3), while Linked Data can support content-negotiation. In either case it can be unclear to clients what is the meaning or equivalence of any additional chunks.

\(^9\)Newer (PR-RequirementSpec-3.0) renames FDOF\(^*\) to FDOR\(^*\) but follows same ordering.
<table>
<thead>
<tr>
<th>FDO Guideline</th>
<th>DOIP 2.0</th>
<th>FDO suggestions</th>
<th>Linked Data Platform</th>
<th>LDP suggestion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Ensure FDO types will not be protocol-bound as DOIP might be updated/replaced</td>
<td>HTTP stable for 30 years, Semantic Web HTTP stable for 30 years, http:// URI mostly</td>
<td>Keep flexibility of RDF serialisation formats which may change more frequently</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Further promote DOIP and justify its benefits. Build tutorials and OSI open</td>
<td>replaced by https://.</td>
<td>Ensure simplicity for end developers, rather than semantic overengineering.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>source implementations. Standardise DOIP-over-HTTP</td>
<td></td>
<td>Example-driven documentation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>alternative.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>G1: <strong>invest for many decades</strong></td>
<td>Handle system stable for 20 years, DOIP 2.0 since 2017.</td>
<td>Ensure FDO types will not be extended/replaced</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G2: <strong>trustworthiness</strong></td>
<td>DOI/Handle trusted by all major academic publishers and data repositories.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G3: <strong>followers FAIR principles</strong></td>
<td>See Table 5 on page 19</td>
<td>Ensure all FAIR principles are covered, build complete examples.</td>
<td>HTTP CRUD operations, Open API (see Table 4 on page 15)</td>
<td>Add explicit expression to show each FAIR principle fulfilled.</td>
</tr>
<tr>
<td>G4: <strong>machine actionability</strong></td>
<td>CRUD and extension operations dynamically listed (see Table 4 on page 15)</td>
<td>Specify which operations should work for a given type, to reduce need for dynamic</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>lookup. Specify input/output expectations formally (e.g. JSON Schema).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>G5: <strong>abstraction principle</strong></td>
<td>Handle PIDs as abstraction base. DOIP operations can use any transport</td>
<td>Document transport protocols as FDOs, recommend which protocol to use.</td>
<td>URI as abstraction base. Does not specify PID requirements.</td>
<td>Give stronger deployment recommendations.</td>
</tr>
<tr>
<td></td>
<td>protocol.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G6: **stable binding between</td>
<td>Machine-navigation through PIDs and operations specified per type.</td>
<td>Make datatype of fields explicit to support navigation.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>entities**</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G7: <strong>encapsulation</strong></td>
<td>Operations discovered at runtime (0.DOIP/Op.ListOperations).</td>
<td>Allow method discovery by type FDOs in advance</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(PR-TypingFDOs-2.0).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>G8: <strong>technology independence</strong></td>
<td>In theory independent, in reality depends on single implementations of</td>
<td>Encourage open source DOIP testbeds and lighter reference implementations</td>
<td>Multiple HTTP implementations, multiple LDP implementations. No FDOF</td>
<td>Develop demonstrator of FDOF usage based on existing LDP server.</td>
</tr>
<tr>
<td></td>
<td>Handle system and DOIP</td>
<td></td>
<td>implementations.</td>
<td></td>
</tr>
<tr>
<td>FDO Guideline</td>
<td>DOIP 2.0</td>
<td>FDO suggestions</td>
<td>Linked Data Platform</td>
<td>LDP suggestion</td>
</tr>
<tr>
<td>---------------</td>
<td>----------</td>
<td>-----------------</td>
<td>---------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><strong>G9: standard compliance</strong></td>
<td>Handle (Sun, Lannom, et al. 2003), DOIP (DOIPV2.0 2018). FDO requirements not standardised yet.</td>
<td>Formalise standard process of FDO requirements (WD-DocProcessStd-1.1)</td>
<td>HTTP, LDP. However FDOF is not yet standardised.</td>
<td>Formalise FDOF from FDOF-SEM working group.</td>
</tr>
<tr>
<td><strong>FDOF1: PID as basis</strong></td>
<td>Extensive use of Handle system.</td>
<td>Clarify how local testing handles can be used during development, how “temporary” FDOs should evolve (PR-PIDProfileAttributes-2.1). Register 0.DOIP/* and 0.FDO/* as actual PIDs.</td>
<td>HTTP URLs as basis for identifiers, but they are frequently not persistent.</td>
<td>Add strong guidance for PID services like w3id and persistence policies (McMurry et al. 2017).</td>
</tr>
<tr>
<td><strong>FDOF2: PID record w/ type</strong></td>
<td>Unspecified how to resolve from Handle to DOIP Service (!), in practice 10320/loc, 0.TYPE/DOIPService, URL, URL_REPLICA</td>
<td>Document requirements for PID Record</td>
<td>w3id/purl PIDs redirect without giving any metadata. Datacite DOIs content-negotiate to give registered metadata.</td>
<td>Add FAIR Signposting (Van de Sompel, Klein, et al. 2022) at PID provider for minimal PID record</td>
</tr>
<tr>
<td><strong>FDOF3: PID resolvable to bytestream &amp; metadata</strong></td>
<td>Byte stream resolvable (0.DOIP/Retrieve), includeElementData option can retrieve bytestream or full object structure. No method/attribute defined for separate metadata, only directly in PID Record. Unclear meaning of multiple items and bytestream chunks.</td>
<td>Clarify expectations for multiple items. Recommend chunks to not be used.</td>
<td>URIs resolvable by default. Multiple ways to resolve metadata, unclear preference.</td>
<td>Add FAIR Signposting and preference order.</td>
</tr>
<tr>
<td><strong>FDOF4: Additional attributes</strong></td>
<td>Freetext attribute keys. Attributes should be defined for FDO type.</td>
<td>Require that attribute keys should be PIDs (or have predefined mapping to PIDs). Explicitly allow attributes not already defined in type.</td>
<td>All attributes individually identified. Any Linked Data attributes can be used by URI or with mapped prefix.</td>
<td>Clarify type expectations of required/recommended/optional attributes.</td>
</tr>
<tr>
<td><strong>FDOF5: Interface: operation by PID</strong></td>
<td>Extended operations use PID, but “pid-like” DOIP operations/types are not registered as handles.</td>
<td>Register 0.DOIP/* and 0.FDO/* as PIDs. Clarify that operations can be mapped to protocol directly.</td>
<td>CRUD operations used directly in HTTP (e.g. PUT). Unclear how to provide PID for additional operations.</td>
<td>Specify how additional operations should be called over HTTP.</td>
</tr>
<tr>
<td>FDO Guideline</td>
<td>DOIP 2.0</td>
<td>FDO suggestions</td>
<td>Linked Data Platform</td>
<td>LDP suggestion</td>
</tr>
<tr>
<td>---------------</td>
<td>----------</td>
<td>-----------------</td>
<td>----------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><strong>FDOF7: FDO Types related to operations</strong></td>
<td>Not yet formalised, by DOIP discoverable on a given FDO rather than type. PR-TypingFDOs leaves this open.</td>
<td>Add explicit relation between type and operations</td>
<td>OPTIONS per LDP Resource, but not by type. Common types (ldp:Resource, ldp:Container) indicate LDP support, but are not required.</td>
<td>Always make LDP types explicit in FDO profile.</td>
</tr>
<tr>
<td><strong>FDOF8: Metadata as FDO, semantic assertions</strong></td>
<td>DOIP includes all metadata in PID Record. Separate Metadata FDO need custom property.</td>
<td>Specify a 0.FDO/metadata or similar to point to Metadata FDOs.</td>
<td>Assertions are always with semantics, using RDF vocabularies. Unspecified how to find additional metadata resources, rdfs:seeAlso is common.</td>
<td>Use FAIR Signposting described by link relation to additional metadata PIDs</td>
</tr>
<tr>
<td><strong>FDOF9: Different metadata levels</strong></td>
<td>Defines open-ended “Response Attributes” without namespaces, but mandated as “None” for all CRUD operations. Metadata would need to be bundled within custom FDO types or attributes. Unclear how levels are separated within single FDO representation (may need FDOF8).</td>
<td>Declare which metadata are expected within response attribute or within FDO object. Require PIDs for custom attributes. Define how alternate metadata levels can be represented separately.</td>
<td>Undefined how to handle multiple metadata granularities or domains, alternative LDP containers can present different views on same stored objects.</td>
<td>Define how to navigate to alternate views and their semantic implications, e.g. owl:sameAs</td>
</tr>
<tr>
<td><strong>FDOF10: Metadata schemas by community</strong></td>
<td>Metadata schemas are in practice managed on single Cordra server as local types, using JSON Schema.</td>
<td>Require types to be FDOs with registered PIDs, implement shared types.</td>
<td>Plethora of existing RDF vocabularies/ontologies managed by larger communities, e.g. OBO Foundry (Smith et al. 2007)</td>
<td>Rather document better how individual ad-hoc schemas can be started for prototypes.</td>
</tr>
<tr>
<td><strong>FDOF11: FDO collections w/ semantic relations</strong></td>
<td>Collection type undefined by DOIP. Informal use of HAS_PARTS Handle attribute (e.g. (Semmler et al. 2022)).</td>
<td></td>
<td>LDP Containers required by specification, also user-created (eg. BasicContainer).</td>
<td>Clarify relation to other collections like DCAT 3 (Dataset Exchange Working Group 2023), Schema.org Dataset, OAI-ORE (Lagoze et al. 2008)</td>
</tr>
<tr>
<td><strong>FDOF12: Deleted FDO preserve PID w/ tombstone</strong></td>
<td>Tombstone for deleted resource undefined by DOIP. 0.DOIP/Status.104 status code does not distinguish “Not Found” or “Gone”</td>
<td>Formalise tombstone requirements with new FDO type</td>
<td>410 Gone recommended, but 404 Not Found common. No requirement for tombstone serialisation</td>
<td>Formalise tombstone requirements and serialisation</td>
</tr>
</tbody>
</table>
Comparing FDO and Web as middleware infrastructures

In this section, we take the perspective that FDO principles are in effect proposing a global infrastructure of machine-actionable digital objects. As such we can consider implementations of FDO as middleware infrastructures for programmatic usage, and can evaluate them based on expectations for client and server developers.

We argue that the Web, with its now ubiquitous use of REST API (Roy Thomas Fielding 2000), can be compared as a similar global middleware. Note that while early moves for developing Semantic Web Services (Fensel et al. 2011) attempted to merge the Web Service and RDF aspects, we are here considering mainly the current programmatic Web and its mostly light-weight use of 3 out of possible 5 stars Linked Data (Michael Hausenblas et al. 2012).

For this purpose, we here utilise the Comparison Framework for Middleware Infrastructures (Zarras 2004) that formalise multiple dimensions of openness, scalability, transparency, as well as characteristics known from Object-oriented programming such as modularity, encapsulation and inheritance.

Based on the analysis in Table 4 on the next page, we make the following observations:

- With respect to the aspect of **Performance**, it is interesting to note that while the first version of DOIP (Reilly 2009) supported multiplexed channels similar to HTTP/2 (allowing concurrent transfer of several digital objects). Multiplexing was removed for the much simplified DOIP 2.0 (DOIPV2.0 2018). Unlike DOIP 1.0, DOIP 2.0 will require a DO response to be sent back completely, as a series of segments (which again can be split the bytes of each binary *element* into sized *chunks*), before transmission of another DO response can start on the transport channel. It is unclear what is the purpose of splitting a binary into chunks on a channel which no longer can be multiplexed and the only property of a chunk is its size\(^{10}\).

- HTTP has strong support for scalability and caching, but this mostly assumes read-operations from static resources. FDO has no view on immutability or validity of retrieved objects, but this should be taken into consideration to support large-scale usage.

- HTTP optimisations for performance (e.g. HTTP/2, multiplexing) is largely used for commercial media distribution (e.g. Netflix), and not commonly used by providers of FAIR data

- Cloud deployment of Web applications give many middleware benefits (Scalability, Distribution, Access transparancy, Location transparancy) – it is unclear how DOIP as a custom protocol would perform in a cloud setting as most of this infrastructure assumes HTTP as the protocol.

- Programmatically the Web is rather unstructured as middleware, as there are many implementation choices. Usually it is undeclared what to expect for a given URL/service, and programmers follow documented examples for a particular service rather than automated programmatic exploration across providers. This mean one can consider the Web as an ecosystem of smaller middlewares with commonalities.

- Many providers of FAIR Linked Data also provide programmatic REST API endpoints, e.g. UNIPROT, ChEMBL, but keeping the FAIR aspects such as retrieving metadata in such a scenario may require combining different services using multiple formats and identifier conventions.

\(^{10}\)Although it is possible with 0.DOIP/Op.Retrieve to request only particular individual elements of an DO (e.g. one file), unlike HTTP’s Range request, it is not possible to select individual chunks of an element’s bytestream.
<table>
<thead>
<tr>
<th>Quality</th>
<th>FDO w/ DOIP</th>
<th>Web w/ Linked Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Openness:</strong> enable extension of applications</td>
<td>FDOs can be cross-linked using PIDs, pointing to multiple FDO endpoints. Custom DOIP operations can be exposed, although it is unclear if these can be outside the FDO server. PID minting requires Handle.net prefix subscription, or use of services like Datacite, B2Handle.</td>
<td>The Web is inherently open and made by cross-linked URLs. Participation requires DNS domain purchase (many free alternatives also exists). PID minting can be free using PURL/ARK services, or can use DOI/Handle with HTTP redirects.</td>
</tr>
<tr>
<td><strong>Scalability:</strong> application should be effective at many different scales</td>
<td>No defined methods for caching or mirroring, although this could be handled by backend, depending on exposed FDO operations (e.g. Cordra can scale to multiple backend nodes)</td>
<td>Cache control headers reduce repeated transfer and assist explicit and transparent proxies for speed-up. HTTP GET can be scaled to world-population-wide with Content-Delivery Networks (CDNs), while write-access scalability is typically managed by backend.</td>
</tr>
<tr>
<td><strong>Performance:</strong> efficient and predictable execution</td>
<td>DOIP has been shown moderately scalable to 100 millions of objects, create operation at 900 requests/second. DOIP protocol is reusable for many operations, multiple requests may be answered out of order (by requestId). Multiple connections possible. Setup is typically through TCP and TLS which adds latency.</td>
<td>HTTP traffic is about 10% of global Internet traffic, excluding video and social networks (Sandvine 2022). HTTP 1 connections are serial and reusable, and concurrent connections is common. HTTP/2 adds asynchronous responses and multiplexed streams (Belshe et al. 2015) but still has TCP+TLS startup costs. For reduced latency, HTTP/3 (Bishop 2022) use QUIC (Iyengar and Thomson 2021) rather than TCP, already adapted heavily (30% of EMEA traffic) of which Instagram &amp; Facebook video is the majority of traffic (Joras and Chi 2020).</td>
</tr>
<tr>
<td><strong>Distribution transparency:</strong> application perceived as a consistent whole rather than independent elements.</td>
<td>Each FDO is accessed separately along with its components (typically from the same endpoint). FDOs should provide the mandatory kernel metadata fields. FDOs of the same declared type typically share additional attributes (although that schema may not be declared). DOIP does not enforce metadata typing constraints, this need to be established as FDO conventions.</td>
<td>Each URL accessed separately. Common HTTP headers provide basic metadata, although it is often not reliable. A multitude of schemas and serializations for metadata exists, conventions might be implied by a declared profile or certain media types. Metadata is not always machine findable, may need pre-agreed API URI Templates (Gregorio, Roy T. Fielding, et al. 2012), content-negotiation (MDN 2023) or FAIR Signposting (Van de Sompel, Klein, et al. 2022).</td>
</tr>
<tr>
<td><strong>Access transparency:</strong> local/remote elements accessed similarly</td>
<td>FDOs should be accessed through PID indirection, this means difficult to make private test setup. Commonly a fixed DOIP server is used directly, which permits local non-PID identifiers.</td>
<td>Global HTTP protocol frequently used locally and behind firewalls, but at risk of non-global URIs (e.g. <a href="http://localhost/object/1">http://localhost/object/1</a>) and SSL issues (e.g. self-signed certificates, local CAs)</td>
</tr>
<tr>
<td><strong>Location transparency:</strong> elements accessed without knowledge of physical location</td>
<td>FDOs always accessed through PIDs. Multiple locations possible in Handle system, can expose geo-info.</td>
<td>PIDs and URL redirects. DNS aliases and IP routing can hide location. Geo-localised servers common for large cloud deployments.</td>
</tr>
<tr>
<td>Quality</td>
<td>FDO w/ DOIP</td>
<td>Web w/ Linked Data</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Concurrency transparency:</strong></td>
<td>No explicit concurrency measures. FDO kernel metadata can include checksum and date.</td>
<td>HTTP operations are classified as being stateless/idempotent or not (e.g. PUT changes state, but can be repeated on failure), although these constraints are occasionally violated by Web applications. Cache control, ETag (e.g. checksum) and modification date in HTTP headers allows detection of concurrent changes on a single resource.</td>
</tr>
<tr>
<td></td>
<td>DOIP status codes, e.g. 0.DOIP/Status.104, additional codes can be added as custom attributes</td>
<td>HTTP status codes e.g. 404 Not Found, specific meaning of standard codes can be documented in Open API. Custom codes uncommon.</td>
</tr>
<tr>
<td><strong>Failure transparency:</strong></td>
<td>Update of PID record URLs, indirection through 0.TYPE/DOIPSERVICEINFO (not always used consistently). No redirection from DOIP service.</td>
<td>HTTP 30x status codes provide temporary or permanent redirections, commonly used for PURLs but also by endpoints.</td>
</tr>
<tr>
<td></td>
<td>FDO requires use of PIDs for object persistence, including a tombstone response for deleted objects. There is no guarantee that an FDO is immutable or will even stay the same type (note: CORDRA extends DOIP with version tracking).</td>
<td></td>
</tr>
<tr>
<td><strong>Persistence transparency:</strong></td>
<td>No transaction capabilities declared by FDO or DOIP. Internal synchronisation possible in backend for Extended operations.</td>
<td>URLs are not required to persist, although encouraged (Berners-Lee 1998). Persistence requires convention to use PIDs/PURLs and HTTP 410 Gone. An URL may change its content, change in type may sometimes force new URLs if exposing extensions like .json. Memento (Van de Sompel, Nelson, et al. 2013) expose versioned snapshots. WebDAV VERSION-CONTROL method (Clemm et al. 2002) (used by SVN).</td>
</tr>
<tr>
<td><strong>Modularity:</strong></td>
<td>FDOs are inherently modular using global PID spaces and their cross-references. In practice, FDOs of a given type are exposed through a single server shared within a particular community/institution.</td>
<td>Limited transaction capabilities (e.g. If-Unmodified-Since) on same resource. WebDAV locking mechanisms (Dusseault 2007) with LOCK and UNLOCK methods.</td>
</tr>
<tr>
<td><strong>Encapsulation:</strong></td>
<td>Indirection by PID gives separation. FDO principles are protocol independent, although it may be unclear which protocol to use for which FDO (although 0.DOIP/Transport can be specified after already contacting DOIP). Cordra supports native DOIP, DOIP over HTTP and Cordra REST API)</td>
<td>The Web is inherently modular in that distributed objects are cross-referenced within a global URI space. In practice, an API’s set of resources will be exposed through a single HTTP service, but modularity enables fine-grained scalability in backend.</td>
</tr>
<tr>
<td></td>
<td>HTTP/1.1 semantics can seamlessly upgrade to HTTP/2 and HTTP/3. http vs https URIs exposes encryption detail(^\text{11}). Implementation details may leak into URIs (e.g. search.aspx), countered by deliberate design of URI patterns (Berners-Lee 1998)) and PIDs via Persistent URIs (PURL).</td>
<td></td>
</tr>
</tbody>
</table>

\(^{11}\) The http protocol (port 80) can in theory also upgrade (Khare and Lawrence 2000) to TLS encryption, as commonly used by Internet Printing Protocol for ipp URIs, but on the Web, best practice is explicit
Inheritance: Deriving specialised interface from another type

DOIP types nested with parents, implying shared FDO structures (unclear if operations are inherited). FDO establishes need for multiple Data Type Registries (e.g. managed by a community for a particular domain). Semantics of type system currently undefined for FDO and DOIP; syntactic types can also piggyback of FDO type’s schema (e.g. CORDRA $ref use of JSON Schema references (Wright et al. 2022)).

Syntactically Media Type with multiple suffixes (Sporny and Guy 2023) (mainly used with +json), declaration of subtypes as profiles (RFC6906) The 'profile' Link Relation Type. In metadata, semantic type systems (RDFS (Guha and Brickley 2014)), OWL2 (W3C OWL Working Group 2012), SKOS (Isaac and Summers 2009)), OpenAPI 3 (D. Miller et al. 2021) inheritance and Polymorphism. XML xsd:schemaLocation or xsd:type (Thompson et al. 2012), JSON $schema (Wright et al. 2022)), JSON-LD @context (Sporny, Longley, et al. 2020). Large number of domain-specific and general ontologies define semantic types, but finding and selecting remains a challenge.

Signal interfaces: asynchronous handling of messages

DOIP 2.0 is synchronous, in FDO async operations undefined. Could be handled as custom jobs/futures FDOs

HTTP/2 multiplexed streams (Belshe et al. 2015), Web Sockets (Rice et al. 2022), Linked Data Notifications (Capadisli and Guy 2017), AtomPub (Gregorio and de hOra 2007), SWORD (Jones and Jefferies 2021), Micropub (Parecki 2017), more typically ad-hoc jobs/futures REST resources

Operation interfaces: defining operations possible on an instance, interface of request/response messages

CRUD predefined in DOIP, custom operations through 0.DOIP/Op.ListOperations (can be FDOs of type 0.TYPE/DOIPOperation, more typically local identifiers like "getProvenance")


Stream interfaces: operations that can handle continuous information streams

Undefined in FDO. DOIP can support multiple byte stream elements (need custom FDO type to determine stream semantics)

HTTP 1.1 (Roy T. Fielding and Reschke 2014a) chunked transfer, HLS (RFC8216) (Pantos and May 2017), MPEG-DASH (ISO/IEC 23009-1 2022)

https (port 443) URLs to ensure following links stay secure.

12HATEOAS: Hypermedia as the Engine of Application State (Roy Thomas Fielding 2000), an important element of the REST architectural style.
Assessing FDO against FAIR

In addition to having “FAIR” in its name, the FAIR Digital Object guidelines (PR-RequirementSpec-3.0) also include G3: FDOs must offer compliance with the FAIR principles through measurable indicators of FAIRness.

Here we evaluate to what extent the FDO guidelines and its implementation with DOIP and Linked Data Platform (Bonino da Silva Santos et al. 2022) comply with the FAIR principles (Mark D. Wilkinson et al. 2016). Here we’ve used the RDA’s FAIR Data Maturity Model (FAIR Data Maturity Model Working Group 2020) as it has decomposed the FAIR principles to a structured list of FAIR indicators (Bahim et al. 2020), importantly considering Data and Metadata separately. In our interpretation for Table 5 on the following page we have for simplicity chosen to interpret “data” in FDOs as the associated bytestream of arbitrary formats, with remaining JSON or RDF structures always considered as metadata.

From this evaluation we observe:

- Linked Data in general is strong on metadata indicators, but LDP approach is weak as it has little concrete metadata guidance.
- FDO/DOIP are stronger on identifier indicators, while Linked Data approach for identifiers relies on best practices.
- Indicators on standard protocols (RDA-A1-04M, RDA-A1-04D, RDA-A1.1-01M, RDA-A1.1-01D) favour LDP’s mature standards (HTTP, URI) – the DOIPv2 specification (DOIPV2.0 2018) has currently only a couple of implementations and is expressed informally. The underlying Handle system for PIDs is arguably mature and commonly used by researchers (this article alone references about 80 DOIs), however DOIs are more commonly accessed as HTTP redirects through resolvers like https://doi.org/ and http://hdl.handle.net/ rather than the Handle protocol.
- RDA-A1-02M and RDA-A1-02D highlights access by manual intervention, which is common for http/https URIs, but also using above PID resolvers for DOIP implementation CORDRA (e.g. https://hdl.handle.net/21.14100/90ec1c7b-6f5e-4e12-9137-0cedd16d1bce), yet neither LDP, FDO nor DOIP specifications recommends human-readable representations to be provided.
- Neither DOIP nor LDP require license to be expressed (RDA-R1.1-01M, RDA-R1.1-02M, RDA-R1.1-03M), yet this is crucial for re-use and machine actionability of FAIR data and metadata to be legal.
- Machine-understandable types, provenance and data/metadata standards (RDA-R1.1-03M RDA-R1.3-02M, RDA-R1.3-02M, RDA-R1.3-02D) are important for machine actionability, but are currently unspecified for FDOs. (WD-ImplAttributesTypesProfiles) explores possible machine-readable FDO types, however the type systems themselves have not yet been formalised. Linked Data on the other side have too many semantic and syntactic type systems, making it difficult to write consistent clients.
- Indicators for FAIR data are weak for either approach, as too much reliance is put on metadata. For instance in Linked Data, given a URL of a CSV file, what is its persistant identifier or license information? FAIR Signposting (Van de Sompel, Klein, et al. 2022) can improve findability of metadata using HTTP Link relations, which enable an FDO-like overlay for any HTTP resource. In DOIP, responses for bytestreams can include the data identifier: if that is a PID (not enforced by DOIP), its metadata is accessible.
- Resolving FDOs via Handle PIDs to the corresponding DOIP server is currently undefined by FDO and DOIP specifications. 0.TYPE/D OPServletInfo lookup is only possible once DOIP server is known.
Table 5. Assessing RDA’s FAIR Data Maturity Model FAIR Data Maturity Model Working Group 2020; Bahim et al. 2020 (first 2 columns) against the FDO guidelines Bonino et al. 2019, FDO implemented with the protocol DOIPv2 DOIPv2.0 2018, Linked Data Platform (LDP) Bonino da Silva Santos et al. 2022 and examples from Linked Data practices in general. (— indicates Unspecified, may be possible with additional conventions)

<table>
<thead>
<tr>
<th>FAIR ID</th>
<th>Indicator</th>
<th>FDO guidelines</th>
<th>FDO/DOIP</th>
<th>FDO/LDP</th>
<th>Linked Data examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>RDA-F1-01M</td>
<td>Metadata is identified by a persistent identifier</td>
<td>FDOF4</td>
<td>Optional Metadata <em>FDO</em> w/separate PID</td>
<td>Content-negotiation to URL, not required to be PID</td>
<td>Metadata typically don’t have own PID</td>
</tr>
<tr>
<td>RDA-F1-01D</td>
<td>Data is identified by a persistent identifier</td>
<td>FDOF1</td>
<td>PIDs required (FDOF1). Handle, DOI.</td>
<td>FDOF-IR (Identifier Record). PID can be any URI</td>
<td>“Cool” URIs (Berners-Lee 1998), PURL services incl. purl.org, w3id.org</td>
</tr>
<tr>
<td>RDA-F1-02M</td>
<td>Metadata is identified by a globally unique identifier</td>
<td>FDOR4 FDOF8</td>
<td>Optional Metadata <em>FDO</em>, unspecified how to indicate</td>
<td>Content-negotiation to URL</td>
<td>Not required, content-negotiation can redirect to URL or Content-Location. FAIR Signposting.</td>
</tr>
<tr>
<td>RDA-F1-02D</td>
<td>Data is identified by a globally unique identifier</td>
<td>FDOF1</td>
<td>All FDOs have PIDs (FDOR1), DOIP uses Handle system</td>
<td>FDOF-IR (Identifier Record)</td>
<td>Always accessed by URL</td>
</tr>
<tr>
<td>RDA-F2-01M</td>
<td>Rich metadata is provided to allow discovery</td>
<td>FDOF2 FDOF4 FDOF8 FDOF9</td>
<td>FDO has key-value metadata. Unclear how to link to additional metadata.</td>
<td>FDOF-IR links to multiple metadata records</td>
<td>RDF-based metadata by content negotiation or FAIR Signposting. Embedded in landing page (RDFa).</td>
</tr>
<tr>
<td>RDA-F3-01M</td>
<td>Metadata includes the identifier for the data</td>
<td>—</td>
<td>id and type are required metadata elements PIDs, also implicit as requests must use PID</td>
<td>PID only required in FDOF-IR record.</td>
<td>PID inclusion typical, but often inconsistent (e.g. <a href="http://www.example.com">www.example.com</a> vs example.com) or missing (use of &lt;&gt; as this subject)</td>
</tr>
<tr>
<td>RDA-F4-01M</td>
<td>Metadata is offered in such a way that it can be harvested and indexed</td>
<td>FDOF10</td>
<td>No, registries not required (except Data Type Registries). Handle registry only searchable by PID.</td>
<td>—</td>
<td>Not specified, several registries/catalogues for vocabularies/types (e.g. (NCBO BioPortal 2022)). Indexing by search engines if exposing HTML w/schema.org.</td>
</tr>
<tr>
<td>FAIR ID</td>
<td>Indicator</td>
<td>FDO guidelines</td>
<td>FDO/DOIP</td>
<td>FDO/LDP</td>
<td>Linked Data examples</td>
</tr>
<tr>
<td>------------</td>
<td>----------------------------------------</td>
<td>----------------</td>
<td>---------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>RDA-A1-01M</td>
<td>Metadata contains information to enable the user to get access to the data</td>
<td>FDOF3 FDOF6</td>
<td>Directly by DOIP, but not included in FDO metadata, HTTP resolution may redirect to landing page</td>
<td>Any property can point to URIs, but unclear if it is data</td>
<td>Common with clickable “follow your nose” URLs</td>
</tr>
<tr>
<td>RDA-A1-02M</td>
<td>Metadata can be accessed manually (i.e. with human intervention)</td>
<td>—</td>
<td>(Cordra HTML landing page from handle.net URIs)</td>
<td>Optional content-negotiation, e.g. by Apache Marmotta, OpenLink Virtuoso</td>
<td>HTTP content-negotiation to HTML is common</td>
</tr>
<tr>
<td>RDA-A1-02D</td>
<td>Data can be accessed manually (i.e. with human intervention)</td>
<td>—</td>
<td>(Cordra HTML landing page from handle.net URIs)</td>
<td>Optional content-negotiation</td>
<td>Direct download, HTML landing pages common for DOIs</td>
</tr>
<tr>
<td>RDA-A1-03M</td>
<td>Metadata identifier resolves to a metadata record</td>
<td>FDOF8+FDOF2</td>
<td>—</td>
<td>—</td>
<td>Content-Location or HTTP redirection may indicate metadata URI</td>
</tr>
<tr>
<td>RDA-A1-03D</td>
<td>Data identifier resolves to a digital object</td>
<td>FDOF2</td>
<td>Required, but frequently not directly resolvable</td>
<td>Recommended, but any URI acceptable</td>
<td>Resolvable HTTP/HTTPS URIs are most common, now infrequent URNs are not directly resolvable</td>
</tr>
<tr>
<td>RDA-A1-04M</td>
<td>Metadata is accessible through standardised protocol</td>
<td>G9 FDOF3</td>
<td>Retrievable from PID (FDOF3), Informal DOIP standard maintained by DONA Foundation</td>
<td>LDP standard maintained by W3C, HTTP standards maintained by IETF, FDO components resolved by informal proposals (custom vocabulary, extra HTTP methods) or HTTP content negotiation</td>
<td>Formal HTTP standards maintained by IETF, HTTP content negotiation, informal FAIR Signposting</td>
</tr>
<tr>
<td>RDA-A1-04D</td>
<td>Data is accessible through standardised protocol</td>
<td>G9</td>
<td>(see above)</td>
<td>HTTP (Roy T. Fielding, Nottingham, et al. 2022)</td>
<td>HTTP/HTTPS, FTP (now less common), GridFTP (Allcock et al. 2005) (for large data), ARK (Kunze and Bermés 2022)</td>
</tr>
<tr>
<td>RDA-A1-05D</td>
<td>Data can be accessed automatically (i.e. by a computer program)</td>
<td>G4 FDOF3 FDOF6</td>
<td>Required, but few client libraries HTTP GET, content-negotiation for fdof/object</td>
<td>—</td>
<td>Ubiquitous, hundreds of HTTP libraries</td>
</tr>
<tr>
<td>FAIR ID</td>
<td>Indicator</td>
<td>FDO guidelines</td>
<td>FDO/DOIP</td>
<td>FDO/LDP</td>
<td>Linked Data examples</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------------------------------------------------</td>
<td>----------------</td>
<td>---------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------</td>
</tr>
<tr>
<td>RDA-A1.1-01M</td>
<td>Metadata is accessible through a free access protocol</td>
<td>G1 G8 G9</td>
<td>Partially realised: Handle system is open(^{13}) protocol (Sun, Reilly, et al. 2003). One server implementation (CNRI 2022), free(^{14}). One DOIPv2 implementation (Cordra): free under BSD-like license (not recognised as Open Source).</td>
<td>LDP is open W3C recommendation (Speicher et al. 2015)</td>
<td>DNS, HTTP, TLS, RDF standards are open, free and universal, large number of Open Source clients and servers.</td>
</tr>
<tr>
<td>RDA-A1.1-01D</td>
<td>Data is accessible through a free access protocol</td>
<td>(FDOR9)</td>
<td>TLS certificates, authentication field (details unspecified)</td>
<td>Implied</td>
<td>URI, DNS, HTTP, TLS</td>
</tr>
<tr>
<td>RDA-A2-01M</td>
<td>Metadata is guaranteed to remain available after data is no longer available</td>
<td>FDOF12</td>
<td>Required, but not currently defined</td>
<td>—</td>
<td>URI, DNS, HTTP, TLS. Non-free DRM may be used (e.g. subscription video streaming)</td>
</tr>
<tr>
<td>RDA-II-01D</td>
<td>Metadata uses knowledge representation expressed in standardised format</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>HTTP authentication, TLS certificates</td>
</tr>
<tr>
<td>RDA-II-02M</td>
<td>Metadata uses machine-understandable knowledge representation</td>
<td>FDOF8</td>
<td>Required</td>
<td>Optional RDF metadata with any vocabulary</td>
<td>—</td>
</tr>
</tbody>
</table>

\(^{13}\)The Handle.net system was previously covered by software patent US6135646A which expired in 2013.

\(^{14}\)The Handle.net public license is not OSI-approved (OSI 2022) as an open source license – it includes usage restrictions and requires Service Agreements. It is not a DOIP requirement to host a local Handle instance, e.g. EOSC provides the B2HANDLE service for acquiring Handle prefixes.
<table>
<thead>
<tr>
<th>FAIR ID</th>
<th>Indicator</th>
<th>FDO guidelines</th>
<th>FDO/DOIP</th>
<th>FDO/LDP</th>
<th>Linked Data examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>RDA-I1-02D</td>
<td>Data uses machine-understandable knowledge representation</td>
<td>G4 G7 FDOR2</td>
<td>No requirements on binary data formats</td>
<td>Only indirectly, LDP Basic Container reference only information resources</td>
<td>Common, specially for scientific data formats</td>
</tr>
<tr>
<td>RDA-I2-01M</td>
<td>Metadata uses FAIR-compliant vocabularies</td>
<td>G3 FDOF10</td>
<td>Informally required</td>
<td>Unspecified, implied by use of RDF?</td>
<td>FAIR practices for LD vocabularies increasingly common, sometimes inconsistent (e.g. PURFLs that don’t resolve) or incomplete (e.g. unknown license)</td>
</tr>
<tr>
<td>RDA-I2-01D</td>
<td>Data uses FAIR-compliant vocabularies</td>
<td></td>
<td></td>
<td></td>
<td>Uncommon, except for some XML and RDF-embedding formats, e.g. Extensible Metadata Platform (XMP) (ISO 16684-1 2019)</td>
</tr>
<tr>
<td>RDA-I3-01M</td>
<td>Metadata includes references to other metadata</td>
<td>FDOR8</td>
<td>Implied (attributes to PIDs), currently unspecified if given attribute is value or reference</td>
<td>—</td>
<td>By definition (Linked Data reference existing URIs (Linked Data 2015)), rdfs:seeAlso, FAIR signposting (Van de Sompel, Klein, et al. 2022) described by</td>
</tr>
<tr>
<td>RDA-I3-01D</td>
<td>Data includes references to other data</td>
<td>G6 FDOR3</td>
<td></td>
<td></td>
<td>URL hyperlinks common in several formats (HTML, PDF, JSON, XML).</td>
</tr>
<tr>
<td>RDA-I3-02M</td>
<td>Metadata includes references to other data</td>
<td>G6 FDOR3</td>
<td>Implied from custom FDO type’s attribute</td>
<td>LDP Direct Container members can be any resources</td>
<td>URI objects are frequently data references, may be indirect via PID</td>
</tr>
<tr>
<td>RDA-I3-02D</td>
<td>Data includes qualified references to other data</td>
<td>FDOR3</td>
<td>Only indirectly through FDO metadata</td>
<td>Indirectly through LDP membership</td>
<td>Uncommon: Link relations, FAIR Signposting</td>
</tr>
<tr>
<td>RDA-I3-03M</td>
<td>Metadata includes qualified references to other data (FDOR3)</td>
<td>FDOR8</td>
<td>Qualification by attribute keys defined per FDO Type</td>
<td>LDP Direct Container</td>
<td>Qualifications by property, PROV bundles (Lebo and Moreau 2013), schema.org/Role</td>
</tr>
<tr>
<td>FAIR ID</td>
<td>Indicator</td>
<td>FDO guidelines</td>
<td>FDO/DOIP</td>
<td>FDO/LDP</td>
<td>Linked Data examples</td>
</tr>
<tr>
<td>------------</td>
<td>---------------------------------------------------------------------------</td>
<td>----------------</td>
<td>-------------------------------</td>
<td>---------</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RDA-R1.01M</td>
<td>Plurality of accurate and relevant attributes are provided to allow reuse</td>
<td>FDOF4</td>
<td>Required. Kernel metadata attributes desired (PR-KernelAttributes-2.0) but not assigned PIDs yet.</td>
<td>Unspecified. Multiple metadata records can allow multiple semantic profiles.</td>
<td>Large number of general and domain-specific vocabularies can make it hard to find relevant attributes. Rough consensus on kernel metadata: schema.org (Schema.Org - Schema.Org 2022), Dublin Core Terms (DCMI Usage Board 2020), DCAT (Browning et al. 2020), FOAF (Brickley and L. Miller 2014)</td>
</tr>
<tr>
<td>RDA-R1.1-01M</td>
<td>Metadata includes information about the licence under which the data can be reused</td>
<td>—</td>
<td>licenseConditions URL/PID in kernel metadata (PR-KernelAttributes-2.0)</td>
<td>—</td>
<td>Dublin Core Terms dct:licence frequently recommended, frequently not required, e.g. by DCAT 2 (Browning et al. 2020)</td>
</tr>
<tr>
<td>RDA-R1.1-02M</td>
<td>Metadata refers to a standard reuse licence</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>SPDX and Creative Commons URIs common, identifiers often inconsistent</td>
</tr>
<tr>
<td>RDA-R1.1-03M</td>
<td>Metadata refers to a machine-understandable reuse licence</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>SPDX documents uncommon</td>
</tr>
<tr>
<td>RDA-R1.2-01M</td>
<td>Metadata includes provenance information according to community-specific standards</td>
<td>FDOR9 FDOR10</td>
<td>Unspecified (some Cordra types add getProvenance methods). PID Kernel attributes?</td>
<td>—</td>
<td>W3C PROV-O, PAV</td>
</tr>
<tr>
<td>FAIR ID</td>
<td>Indicator</td>
<td>FDO guidelines</td>
<td>FDO/DOIP</td>
<td>FDO/LDP</td>
<td>Linked Data examples</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------------------------------------------------</td>
<td>----------------</td>
<td>----------</td>
<td>---------</td>
<td>--------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RDA-R1.2-02M</td>
<td>Metadata includes provenance information according to a cross-community language</td>
<td>FDOR9 FDOR8</td>
<td>—</td>
<td>—</td>
<td>W3C PROV-O (Lebo, McGuinness, et al. 2013), PAV (Ciccarese et al. 2013), Dublin Core Terms (DCMI Usage Board 2020)</td>
</tr>
<tr>
<td>RDA-R1.3-01M</td>
<td>Metadata complies with a community standard</td>
<td>FDOR10 FROR8</td>
<td>(Emerging, e.g. DiSSCo Digital Specimen (Hardisty et al. 2022))</td>
<td>—</td>
<td>Common, e.g. DCAT 2 (Browning et al. 2020), BioSchemas (Gray et al. 2017)</td>
</tr>
<tr>
<td>RDA-R1.3-01D</td>
<td>Data complies with a community standard</td>
<td>(FDOR3)</td>
<td>—</td>
<td>—</td>
<td>Common, HTTP use registered IANA media types, additional scientific file formats frequently not standardised or identified</td>
</tr>
<tr>
<td>RDA-R1.3-02M</td>
<td>Metadata is expressed in compliance with a machine-understandable community standard</td>
<td>FDOF4 FDOF10</td>
<td>Recommended</td>
<td>—</td>
<td>Common practice for ontologies, especially in bioinformatics, e.g. BioPortal (NCBO BioPortal 2022), Darwin Core (Wieczorek et al. 2012)</td>
</tr>
<tr>
<td>RDA-R1.3-02D</td>
<td>Data is expressed in compliance with a machine-understandable community standard</td>
<td>(FDOR2)</td>
<td>No, FDO is typed but data can be any bytestream</td>
<td>—</td>
<td>Occassionally, (e.g. GFF3, FITS, ESRI)</td>
</tr>
</tbody>
</table>
EOSC INTEROPERABILITY FRAMEWORK

The European Open Science Cloud (EOSC) is a large EU initiative to promote Open Science by implementing a joint research infrastructure by federating existing and new services and focusing on interoperability, accessability, best practices as well as technical infrastructure (Ayris et al. 2016). The EOSC Interoperability Framework (Corcho, Eriksson, et al. 2021) details the principles for creating a common way to achieve interoperability between all digital aspects of research activities in EOSC, including data, protocols and software. The recommendations are realized through 4 layers, Technical (e.g. protocols), Semantic (e.g. metadata models), Organisational (e.g. recommendations) and Legal (e.g. agreements), with a particular aim to address the FAIR interoperability principles and building on the concept of FAIR Digital Objects.

As covered in our introduction on page 1, EOSC proposes FAIR Digital Objects as a way to improve interoperability, for instance invoked by scientific workflows, carried by metadata frameworks and semantic artefacts. Therefore we here find it important to summarize how FDO and Linked Data can help satisfy the EOSC requirements.

In Table 6 on page 27 we review the EOSC Interoperability Framework (EOSC IF) recommendations, and evaluate to what extent they are addressed by the principles of FDO and Linked Data or their common implementations.

Firstly, we observe that the EOSC IF recommendations are at a high level, mainly affecting governance and practices by communities. This Organizational level is also highlighted by the FDO recommendations, for instance the FDO Typing (PR-TypingFDOs-2.0) propose a governance structure to recognize community-endorsed services. While these community aspects are not mandated by Linked Data practices, best practices have become established for aspects like ontology development (Norris et al. 2021). EOSC IF’s Technical layer is likewise at a architecturally high level, such as service-level agreements, but also highlight PID policies which is strongly required by FDO, while Linked Data communities choose PID practices separately. The recommendations for the Semantic layer is largely already implemented by Linked Data practices, yet for FDO mostly consist of encouragements. For instance clear definitions of semantic concepts is required by FDO guidelines, but how to technically define them has not been formalised by FDO specifications.

The Legal layer of interoperability is perhaps the one most emphasised by EOSC, by enabling collaboration across organizational barriers to jointly build a research infrastructure, but this is an area that both FDO and Linked Data are relatively weak in directly supporting. The EOSC IF recommendations in this layer are largely related to governance practices and metadata, for instance licensing, privacy and usage policies; these are also essential for cross-institutional and cross-repository access of FAIR objects.

Likewise, search and indexing is important FAIR aspect for Findability, but is poorly supported globally by FDO and Linked Data. Efforts such as Open Research Knowledge Graph (ORKG) (Jaradeh et al. 2019), DataCite’s PID Graph (Fenner and Aryani 2019) and Google Knowledge Graph (Singhal 2012) have improved programmatic findability to some degree, however not significantly for domain-specific semantic artefacts, currently scattered across multiple semantic catalogues (Corcho, Ekaputra, et al. 2023). There is a strong role for organizations like EOSC to provide such broader registries, moving beyond scholarly output metadata federations. The EOSC Marketplace\textsuperscript{15} has for instance recently been expanded to include training material, software and data sources.

\textsuperscript{15}https://marketplace.eosc-portal.eu/
Table 6. Assessing EOSC Interoperability Framework (Corcho, Eriksson, et al. 2021, section 3.6) against the FDO guidelines (Bonino et al. 2019) and Linked Data practices.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Recommendation</th>
<th>FDO</th>
<th>Linked Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technical</td>
<td>Open Specification</td>
<td>FDO specifications are semi-open, process gradually more transparent</td>
<td>Open and transparent standard processes through W3C &amp; IETF</td>
</tr>
<tr>
<td>Technical</td>
<td>Common security &amp; privacy framework</td>
<td>Unspecified</td>
<td>TLS for encryption, multiple approaches for single-sign-on (e.g. ORCID, Life Science Login). Privacy largely unspecified.</td>
</tr>
<tr>
<td>Technical</td>
<td>Easy SLAs for service providers</td>
<td>Unspecified</td>
<td>None</td>
</tr>
<tr>
<td>Technical</td>
<td>Access data in different formats</td>
<td>None formalised, custom operations or relations</td>
<td>Content-negotiation, rel=alternate relations</td>
</tr>
<tr>
<td>Technical</td>
<td>Coarse-grained/fine-grained search tools</td>
<td>Freetext 0.DOIP/Op/Search on local DOIP, no federation</td>
<td>Coarse-grained e.g. Google Dataset Search, fine-grained (e.g. federated SPARQL) require detailed vocabulary/metadata insight</td>
</tr>
<tr>
<td>Technical</td>
<td>Clear PID policy</td>
<td>Strong FDO requirements, tends towards Handle system.</td>
<td>Not required, different communities set policies</td>
</tr>
<tr>
<td>Semantic</td>
<td>Clear definitions for concepts/metadata/schemas</td>
<td>Required by FDO requirements, but not yet formalised</td>
<td>Ontologies, SKOS, OWL</td>
</tr>
<tr>
<td>Semantic</td>
<td>Semantic artefacts w/ open licenses</td>
<td>All artefacts are PIDs, license not yet required by kernel metadata</td>
<td>Open License is best practice for ontology publishing</td>
</tr>
<tr>
<td>Semantic</td>
<td>Documentation for each semantic artefact</td>
<td>No direct rendering from FDO, no requirement for human-readable description</td>
<td>Ontology rendering, content-negotiation</td>
</tr>
<tr>
<td>Semantic</td>
<td>Repositories of artefacts</td>
<td>Required, but not formalised</td>
<td>Bioontologies, otherwise not usually federated</td>
</tr>
<tr>
<td>Semantic</td>
<td>Repositories w/ clear governance</td>
<td>Recommended</td>
<td>Largely self-governed repositories, if well-established may have clear governance.</td>
</tr>
<tr>
<td>Semantic</td>
<td>Minimal metadata model for federated discovery</td>
<td>Kernel metadata (PR-KernelAttributes-2.0) based on RDA recommendations (Weigel et al. 2018).</td>
<td>DCAT, schema.org, Dublin Core</td>
</tr>
<tr>
<td>Semantic</td>
<td>Crosswalks from minimal metadata model</td>
<td>FDO Typing recommends referencing existing type definitions, but not as separate crosswalks</td>
<td>Multiple crosswalks for common metadata models, but frequently not in semantic format</td>
</tr>
<tr>
<td>Semantic</td>
<td>Extensibility options for disciplinary metadata</td>
<td>Communities encouraged to establish own types</td>
<td>Extensible by design, domain-specific metadata may be at different granularity</td>
</tr>
<tr>
<td>Semantic</td>
<td>Clear protocols/building blocks for federation/harvesting of artefact catalogues</td>
<td>Collection types not yet defined</td>
<td>SWORD, OAI-PMH</td>
</tr>
</tbody>
</table>
### Table 6. Assessing EOSC Interoperability Framework (Corcho, Eriksson, et al. 2021, section 3.6) against the FDO guidelines (Bonino et al. 2019) and Linked Data practices.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Recommendation</th>
<th>FDO</th>
<th>Linked Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organisational</td>
<td>Interoperability-focused rules of participation recommendations</td>
<td>Recommended</td>
<td>Implied only by some communities, tendency to specialise</td>
</tr>
<tr>
<td>Organisational</td>
<td>Usage recommendations of standardised data formats</td>
<td>None</td>
<td>None – but common for metadata (e.g. JSON-LD)</td>
</tr>
<tr>
<td>Organisational</td>
<td>Usage recommendations of vocabularies</td>
<td>Recommended by community</td>
<td>Common (see RDMKit)</td>
</tr>
<tr>
<td>Organisational</td>
<td>Usage recommendations of metadata</td>
<td>Recommended by community</td>
<td>RO-Crate, Bioschemas</td>
</tr>
<tr>
<td>Organisational</td>
<td>Management of permanent organization names/Functions</td>
<td>Handle owner, but unclear contact. Contact info in DOIP service provider</td>
<td>ROR, DCAT contacts.</td>
</tr>
<tr>
<td>Legal</td>
<td>Standardised human and machine-readable licenses</td>
<td>None</td>
<td>SPDX, but not that frequently used</td>
</tr>
<tr>
<td>Legal</td>
<td>Permissive licenses for metadata (CC0, CC-BY-4.0)</td>
<td>Undefined</td>
<td>Both CC0, CC-BY-4.0 common, e.g. in DCAT</td>
</tr>
<tr>
<td>Legal</td>
<td>Different licenses for different parts</td>
<td>Each part as separate FDO can have separate license</td>
<td>DCAT, RO-Crate, Named graphs for splitting metadata</td>
</tr>
<tr>
<td>Legal</td>
<td>Mark expired/inexistent copyright</td>
<td>Undefined</td>
<td>Unclear, semantics assume copyright valid</td>
</tr>
<tr>
<td>Legal</td>
<td>Mark orphaned data</td>
<td>Tombstone for deleted data, but no owner of DOIP server means FDO disappears</td>
<td>Frequently data and endpoint has no known maintainer, archiving in common repositories becoming common</td>
</tr>
<tr>
<td>Legal</td>
<td>List recommended licenses</td>
<td>Undefined</td>
<td>Best practice recommendations</td>
</tr>
<tr>
<td>Legal</td>
<td>Track license evolution for dataset</td>
<td>Undefined</td>
<td>Versioning with PAV/PROV/DCAT</td>
</tr>
<tr>
<td>Legal</td>
<td>Policy/guidance for patent/trade secrets violation</td>
<td>Undefined</td>
<td>Undefined, legal owner may be specified. ODRL can express policies.</td>
</tr>
<tr>
<td>Legal</td>
<td>GDPR compliance for personal data</td>
<td>Undefined</td>
<td>Undefined</td>
</tr>
<tr>
<td>Legal</td>
<td>Restrict access/use if legally required</td>
<td>By transport protocol (undefined by FDO/DOIP)</td>
<td>Diverging approaches, typically landing pages w/ auth&amp;auth or click-thru</td>
</tr>
<tr>
<td>Legal</td>
<td>Harmonised terms-of-use</td>
<td>Undefined</td>
<td>Undefined</td>
</tr>
<tr>
<td>Legal</td>
<td>Alignment between EOSC and national legislation</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

**DISCUSSION**

We have evaluated the FAIR Digital Object concept using multiple frameworks, and contrasted FDO against existing experiences from Linked Data on the Web. In this section we discuss the implications of this evaluation, and propose how these two approaches can be better combined.

**Framework evaluation**

Having considered FDO and the Web architecture as interoperability frameworks (on page 7), we observe that neither are magic bullets, but each bring different aspects of interoperability. The Web comes...
with a large degree of flexibility and openness, however this means interoperability can suffer as services have different APIs and data models, although with common patterns. This is also true for Linked Data on the Web, with many overlapping ontologies and frequent inconsistencies in resolution mechanisms; although somewhat alleviated in recent years by schema.org becoming common metadata model for semantic markup inline in Web pages. The Web is based on a common HTTP protocol which has remained stable architecturally throughout its 32 years of largely backwards-compatible evolution. FDO on the other side sets down multiple rigid rules for identifiers, types, methods etc. that are advanterous for interoperability and predictability for FAIR consumption. Yet there is a large degree of freedom in how the FDO rules can be implemented by a given community, for instance there is no common metadata model or identifier resolution mechanism, and DOIP is just one possible transport method for FDOs, which itself does not enforce these rules.

When evaluating FDO implementations against the FDO guidelines (on page 10) we see that several technical pieces and community practices still need to be developed and further defined, for instance the FDO type system, how to declare FDO actions, how to resolve persistent identifiers, or how to know which pattern of FDO composition is used. Achieving fully interoperable FAIR Digital Objects would require further convergence on implementation practices, and it is not given that this needs to diverge from the established Web architecture. It is not clear from FDO guidelines if moving from HTTP/DNS to DOIP/Handle as a way to expose distributed digital objects will benefit FAIR practitioners, when both approaches require additional equally implementable restrictions and conventions, such as using persistent identifiers or pre-defining an object’s type.

Considering this, by comparing FDO and Web as middleware (on page 14) we saw that programmatic access to digital objects, a core promise of FDO, is not particularly improved by the use of the protocol DOIP as compared to HTTP, e.g. lack of concurrency and transparancy. Recent updates to HTTP have added many features needed for large-scale usage such as video streaming services (e.g. caching, multiplexing, cloud deployments), and having the option to transparently apply these also to FDOs seems like a strong incentive. Many programmatic features for distributed objects are however missing or needing custom extensions in both aspects, such as transactions, asynchronous operations and streaming.

By assessing FDO against the FAIR principles (on page 18) we found that both FDO implementations are underspecified in several aspects (licences, provenance, data references, data vocabularies, metadata persistence). While there are implementations of each of these in general Linked Data examples, there is no single set of implementation guides that fully realizes the FAIR principles. FAIRification efforts like the FAIR Cookbook (Rocca-Serra et al. 2023) and FAIR Implementation Profiles (Schultes, Magagna, et al. 2020) are bringing existing practices together, but there remains a potential role for FDO in giving a coherent set of implementation practices that can practically achieve FAIR. Significant effort, also within EOSC, is now moving towards FAIR metrics (Devaraaju et al. 2021), which in practice need to make additional assumptions on how FAIR principles are implemented, but these are not always formalized (Mark D Wilkinson et al. 2022) nor can they be taken to be universally correct (Verburg et al. 2023). Given that most of the existing FAIR guides and assessment tools are focused on Web and Linked Data, it would be reasonable for FDO to then provide a profile of such implementation choices that can achieve best of both worlds.

EOSC has been largely supportive of FDO, FAIR and related services. By contrasting the EOSC Interoperability Framework (on page 25) with FDO, we found that there are important dimensions that are not solved at a technical level, but through organization collaboration, legal requirements and building community practices. FDO recommendations highlight community aspects, but at the same time the largest FAIR communities in many science domains are already producing and consuming Linked Data. Just as the Linked Data community has a challenge in convincing more research fields to use Semantic Web technologies, FDO currently need to build many new communities in areas that have shown interest in that approach (e.g. material science). It may be advantageous for both these effort to be aligned and jointly promoted under the EOSC umbrella.

**What does FDO mean for Linked Data?**

The FAIR Digital Object approach raises many important points for Linked Data practitioners. At first glance, the explicit requirements of FDOs may seem to be easy to fulfill by different parts of the Semantic Web Cake (Berners-Lee 2000, slide 10), as has previously been proposed (Soiland-Reyes, Castro, et al.
2022). However, this deeper investigation, based on multiple frameworks, highlights that the openness and variability of how Linked Data is deployed can make it difficult to achieve the FDO goals without significant effort.

While RDF and Linked Data have been suggested as prime candidates for making FAIR data, we argue that when different developers have too many degrees of freedom (such as serialization formats, vocabularies, identifiers, navigation), interoperability is hampered – this makes it hard for machines to reliably consume multiple FAIR resources across repositories and data providers. Indeed, this may be one reason why the initial FDO effort steered away from Linked Data approaches, but now seems in a danger of opening the many same degrees of freedom within FDO.

We therefore identify the need for a new explicit FDO profile of Linked Data that sets pragmatic constraints and stronger recommendations for consistent and developer-friendly deployment of digital objects. Such a combination of efforts could utilise both the benefits of mature Semantic Web technologies (e.g. federated knowledge graph queries and rich validation) and data management practices that follow FDO guidance in order to grow an ecosystem of machine-actionable objects. It is beyond the scope of this work to detail such a profile, but we suggest the following potential key aspects:

- Use HTTP(S) as protocol
- Use URIs as identifiers, with persistent identifier promises
- Provide consistent identifier resolution that does not require heuristics
- Common core metadata model
- References are always URIs, and should be persistent identifiers
- Types, attributes and actions are self-defined by their identifier
- Use Web approaches directly where possible, rather than wrap in a new model

The FAIR and Linked Data communities likewise need to recognize the need for simpler, more pragmatic approaches that make it easier for FAIR practitioners to adapt the technologies with "just enough" semantics.

CONCLUSION

In this work, we have considered FAIR Digital Objects (FDO) as a potential distributed object system for FAIR data and compared it with established Web approaches focusing on Linked Data. We have described the background of the Semantic Web and FAIR Digital Objects, and evaluated both using multiple conceptual frameworks.

We find that both FDO and Linked Data approaches can significantly benefit from each-other and should be aligned further. Namely, Linked Data proponents need to make their technologies more approachable, agreeing on predictable and consistent implementations of FAIR principles.

The FDO recommendations show that FAIR thinking in this regard need to move beyond data publishing and into machine actionability across digital objects, and with broader community consensus. As flexibility for extensions is a necessary ingredient alongside rigidity for core concepts, the FDO community likewise need to settle on directly implementable specifications rather than just guidelines, and avoid making similar mistakes as learnt by early Semantic Web adopters.

By implementing the goals of FAIR Digital Objects with the mature technology stack developed for Linked Data, EOSC research infrastructures and researchers in general can create and use FAIR machine-actionable research outputs for decades to come.

ACKNOWLEDGMENTS

This work was funded by the European Union programmes Horizon 2020 under grant agreements H2020-INFRAEDI-02-2018 823830 (BioExcel-2), H2020-INFRAEOSC-2018-2 824087 (EOSC-Life) and Horizon Europe under grant agreements HORIZON-INFRA-2021-EMERGENCY-01 101046203 (BY-COVID), HORIZON-INFRA-2021-EOSC-01 101057388 (EuroScienceGateway), HORIZON-INFRA-2021-EOSC-01-05 101057344 (FAIR-IMPACT), HORIZON-INFRA-2021-TECH-01 101057437 (BioDT); HORIZON-CL4-2021-HUMAN-01-01 101070305 (ENEXA) and by UK Research and Innovation (UKRI) under the UK government’s Horizon Europe funding guarantee grants 10038963 (EuroScienceGateway), 10038992 (FAIR-IMPACT), 10038930 (BioDT).

We would like to acknowledge the FAIR Digital Object Forum (FDO 2022) community and working groups, where SSR and CG are members.
AUTHOR CONTRIBUTIONS
Contributions to this article according to the CASRAI CRediT taxonomy16:

Stian Soiland-Reyes Conceptualization, Formal Analysis, Funding acquisition, Investigation, Methodology, Writing – original draft, Writing – review & editing
Carole Goble Funding acquisition, Supervision, Writing – review & editing
Paul Groth Conceptualization, Methodology, Supervision, Writing – review & editing

REFERENCES


Berners-Lee, Tim (1998). Cool URIs don’t change. URL: https://www.w3.org/Provider/Style/URI (visited on 2023-02-02).


16https://credit.niso.org/


Browning, David, Peter Winstanley, Andrea Perego, Simon Cox, Riccardo Albertoni, and Alejandra Gonzalez Beltran (2020-02-04). *Data Catalog Vocabulary (DCAT) - Version 2*. W3C Recommendation. URL: https://www.w3.org/TR/2020/REC-vocab-dcat-2-20200204/.


Fenner, Martin and Amir Aryani (2019). “Introducing the PID Graph”. In: https://doi.org/10.5438/jwvf-8a66.


Kelly, Mike (2016-05-12). *JSON Hypertext Application Language*. Internet-Draft draft-kelly-json-hal-08.

Work in Progress. Internet Engineering Task Force. 11 pp. URL: https://datatracker.ietf.org/doc/draft-kelly-json-hal/08/.


Kontokostas, Dimitris and Holger Knublauch (2017-07). *Shapes Constraint Language (SHACL)*. W3C Recommendation. URL: https://www.w3.org/TR/2017/REC-shacl-20170720/.


Schreiber, Guus and Yves Raimond (2014-06). RDF 1.1 Primer. W3C Note. URL: http://www.w3.org/TR/2014/NOTE-rdf11-


Schultes, Erik and Peter Wittenburg (2019). “FAIR Principles and Digital Objects: Accelerating Convergence on a Data Infrastructure”. In: Data Analytics and Management in Data Intensive Domains: 20th International Conference, DAMDID/RCDL 2018, Moscow, Russia, October 9–12, 2018, Re-


Semmler, Tido et al. (2022). IPCC DDC: AWI AWICM1.1MR model output prepared for CMIP6 CMIP historical. World Data Center for Climate (WDCC) at DKRZ. HDL: 21.14100/2f6cf49d3-0608-3373-a47f-0e721b7eaa87 URL: https://www.wdc-climate.de/ui/entry?acronym=C6CMAWAWMhi


FDO Specifications


