

Evaluating FAIR Digital Object as a distributed object system

This manuscript ([permalink](#)) was automatically generated from [stain/2022-fdo-paper@7c2ead7](#) on August 23, 2022.

Authors


- **Stian Soiland-Reyes**

 [0000-0001-9842-9718](#) ·  [stain](#) ·  [soilandreyes](#)

Department of Computer Science, The University of Manchester, UK; Informatics Institute, Faculty of Science, University of Amsterdam, NL ·

Funded by [BioExcel-2](#) (European Commission [H2020-INFRAEDI-2018-1 823830](#)); [BY-COVID](#) (European Commission [HORIZON-INFRA-2021-EMERGENCY-01 101046203](#)); [FAIR-IMPACT](#) (European Commission (European Commission [HORIZON-INFRA-2021-EOSC-01 101057344](#)))

- **Carole Goble**

 [0000-0003-1219-2137](#) ·  [carolegoble](#) ·  [CaroleAnneGoble](#)

Department of Computer Science, The University of Manchester, UK · Funded by [BioExcel-2](#) (European Commission [H2020-INFRAEDI-2018-1](#)

[823830](#)); [EOSC-Life](#) (European Commission [H2020-INFRAEOSC-2018-2 824087](#)); [BY-COVID](#) (European Commission [HORIZON-INFRA-2021-](#)

[EMERGENCY-01 101046203](#)); [BioDT](#) (European Commission [HORIZON-INFRA-2021-TECH-01-01 101057437](#)); [FAIR-IMPACT](#) (European

Commission (European Commission [HORIZON-INFRA-2021-EOSC-01 101057344](#)))

- **Paul Groth**

 [0000-0003-0183-6910](#) ·  [pgroth](#) ·  [pgroth](#)

Informatics Institute, Faculty of Science, University of Amsterdam, NL

Abstract

TODO: Rewrite from notes below to actual abstract.

FAIR Digital Object is an emerging concept from EOSC. This is important. Worthwhile to understand how semantic technologies and semantic web vision relate to this emerging landscape. Here we do this systematically by comparing the technologies introduced under the banner of FAIR digital Object and Semantic Web.

What is the message of this paper?

1. These are the overlaps
2. This is what FDO is requiring but not commonly deployed
3. DOI indirection is emphasized. It is used in SW, but the idea for stability. Embrace it!
4. PROV had the idea of using indirection to help us represent provenance of objects - digital twin
5. Lessons for Semantic Web community. What is missing in FDO and in SW.
6. Contribution is also about thinking about Semantic Web as different architectural levels. Interoperability level, Middleware level. Governance level. Data level.
7. Go back to the old SW layer cake. Explains why we picked these frameworks.
8. Lessons for SW: Parts of the stack that is less. FDO contributions.

Semantic Web in a way already implements FDO, but other things that SW perhaps should drop in emphasis to better support FDO and FAIR vision. More about indirection, visibility.

Have all the ingredients, but not cooking properly.

Systematic through frameworks.

Emerging stack - how does it compare to what we've already done? What are the implications for our design and research? What new technology is needed?

Background

TODO Why we are examining FDO. Proposed by EOSC etc. Why we want to compare with Linked Data.

FAIR Digital Object

The concept of **FAIR Digital Object** [17] has been introduced as way to expose research data as active objects that conform to the FAIR principles [18]. This builds on the *Digital Object* (DO) concept [19], first introduced in 1995 [20] 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 X.1255 framework [21] which introduced an abstract *Digital Entity Interface Protocol* for managing such objects programmatically, first realized by the Digital Object Interface Protocol (DOIP) v1 [22].

Next steps for FDO

The FAIR Digital Object Forum [1] working groups are preparing more detailed [requirement documents](#) setting out the path for realizing FDOs, named *FDO Recommendations*. As of 2022-05-13, these documents are in draft stage, undergoing internal review, meanwhile the FDO Forum is formalizing the process for maturing and making these recommendations open for public review. As these drafts clarify the future aims and focus of FAIR Digital Objects, we provide their brief summaries below:

The **FDO Forum Document Standards** [2] 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, finalized as *FDO Forum Recommendation* (REC) following any revisions. In addition,

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 organize complex digital entities, associated with identifiers, metadata, and supporting automated processing [23].

Recently, FDOs have been recognized by the European Open Science Cloud (EOSC) as a suggested part of its Interoperability Framework [24], 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 [4]. 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 section [sec:next-step-fdo?]).

FDO approaches

FDO is an evolving concept. A set of FDO Demonstrators [13] highlight how current adapters are approaching implementations of FDO from different angles:

- Building on the Digital Object concept, using the simplified DOIP v2 specification [12], 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 realized using the reference implementation Cordra. FDO types are registered in the local Cordra instance, where they are specified using JSON Schema [26]) and PIDs are assigned using the Handle system. Several type registries have been established.
- Following the traditional Linked Data approach, but using the DOIP protocol, e.g. using JSON-LD and schema.org within DOIP (NIST for material science).
- Approaching the FDO principles from existing Linked Data practices on the Web (e.g. WorkflowHub use of RO-Crate and schema.org).

From this it becomes apparent that there is a potentially large overlap between the goals and approaches of FAIR

the forum may choose to *endorse* existing third-party notes and specifications.

The **FDO Requirement Specifications** [3] is an update of [4] as the foundational definition of FDO. This sets the criteria for classifying a digital entity as a FAIR Digital Object, allowing for multiple implementations. The requirements shown in table [tbl:fdo-checks] are largely equivalent, but here clarified with references to other FDO documents.

The **Machine actionability** [5] 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 artifacts, 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 [6] classifies different granularities for organizing 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 [7]

PID Profiles & Attributes [8] 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 [9]. 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*, compared with a richer set of more specific attributes as part of the *metadata* for an FDO, possibly represented as a separate FDO.

Granularity, Versioning, Mutability [10] 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 [11] is an endorsement of the DOIP v2.0 [12] specification as a potential FDO implementation, as it has been applied by several institutions [13]. 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 [14] 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 [15] is suggested as one type of service to list FDOs. This document highlights potential practices by repositories and their clients, but adds no particular requirements (e.g. how should failed upload checks be reported?).

Typing FAIR Digital Objects [16] 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 organizing and locating types. Operations applicable to an FDO is not predefined for a

Digital Objects and Linked Data, which we'll cover in the following section @.

From the Semantic Web to Linked Data

In order to describe *Linked Data* as it is used today, we'll start with an (opinionated) briefing 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 [27], at a time the Web had been widely established for information exchange, as a global set of hypermedia documents that 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 standardized 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 realized with the Resource Description Framework (RDF) [28] that used *triples* of subject-predicate-object statements, with its initial serialization format [29] 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 [30], 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, the meaning of a relation can be resolved by following its hyperlink to the definition within a *vocabulary*.

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 formalized 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) had a HTML *representation* for human readers, along with a RDF representation for 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 a large academic interest and growth of the Semantic Web, with the development of more formal representation system for ontologies, such as OWL, allowing complex class hierarchies and logic inference rules following *open world* paradigm (e.g. a *ex:Parent* is equivalent to a subclass of *foaf:Person* which must *ex:hasChild* at least one *foaf:Person*, then if we know *:Alice a ex:Parent* we can infer *:Alice ex:hasChild [a foaf:Person]* even if we don't know who that child is). More human-readable syntaxes of RDF such as Turtle (shown in this paragraph) evolved at this time, and conferences such as ISWC 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 [32] (pattern-based triple queries), [named graphs](#) (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, rdfliib.js), and numerous graph visualization (many of which struggle with usability for more than 20 nodes).

type, however operations naturally will require certain FDO types to work. How to define such FDO operations is not specified.

It is worth pointing out at that, except for the DOIP endorsement, all of these documents are abstract, in the sense that they permit any technical implementation of FDO, if used according to the recommendations.

The creation of RDF-based knowledge graphs grew particularly in fields like bioinformatics, e.g. for describing genomes and proteins. 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 which protein a DNA sequence would make, or 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 [33] (e.g. `dct:title skos:closeMatch schema:name` to indicate near equivalence of two properties). Automated ontology mappings have identified large potential overlaps (e.g. 372 definitions of `Person`) [34].

The move towards *open science* data sharing practices from the late 2000s encouraged knowledge providers to distribute collections of RDF descriptions as downloadable *datasets*⁴, 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 [35]), 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 adapters of the Semantic Web primarily utilized 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 [36]. Just like *link rot* affect regular Web pages and their citations from scholarly communication [37], for a majority of described RDF resources in the [Linked Open Data](#) (LOD) Cloud's gathering of more than thousand datasets, unfortunately they don't actually link to (still) downloadable (*dereferenceable*) Linked Data [38]. Another challenge facing potential adapters is the plethora of choices, not just to navigate, understand and select to reuse the many possible vocabularies and ontologies [39], but also technological choices on RDF serialization (at least [7 formats](#)), type system (RDFS [40], OWL [41], OBO [42], SKOS [33]), hash vs slash, HTTP status codes and PID redirection strategies [31].

Linked Data: Rebuilding the Web of Data

The **Linked Data** concept [43] was kickstarted as a counter-reaction to this development of the Semantic Web, as a set of best practices [44] to bring the Web aspect back into focus. Crucially to Linked Data is to *reuse existing URIs* where they exist, rather than always make new identifiers. This means a loosening of the semantic restrictions previously applied, and an emphasis on building navigatable 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* (Open Graph [45], RDFa [46], Microdata [47]) we find JSON-LD [48] as a Web-focused RDF serialization that aims for improved programmatic generation and consumption, including from Web applications. JSON-LD is as of 2022-05-13 used⁵ by 42.7% of the top 10 million websites [49].

Recently there has been a renewed emphasis to improve the *Developer Experience* [50] for consumption of Linked Data, for instance RDF Shapes (expressed in SHACL [51] or ShEx [52]) [53] can be used to validate RDF Data [54] before consuming it programmatically, 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 technologies, which hampers adaption by non-LD experts [55], 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 [56]. 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 [57], the use of Linked Data as part of REST Web APIs [58], and as shown by the big uptake by publishers to annotate the Web using schema.org [59], with vocabulary use patterns documented by copy-pastable JSON-LD examples, rather than by formalized ontologies or developer requirements to understand the full Semantic Web stack.

FAIR

Comparing FDO against existing frameworks

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

1. An Interoperability Framework and Distributed Platform for Fast Data Applications [60], which proposes quality measurements for comparing how frameworks support interoperability, particularly from a service architectural view
2. The FAIR Digital Object guidelines [4], validated against its implementations for completeness.
3. A Comparison Framework for Middleware Infrastructures [61], which suggest dimensions like openness, performance and transparency, mainly focused on remote computational methods
4. Cross-checks against RDA’s FAIR Data Maturity Model [62] to find how the FAIR principles are achieved in FDO, in particular considering access, sharing and openness
5. EOSC Interoperability Framework [24] which gives recommendations for technical, semantic, organizational and legal interoperability, particularly from a metadata perspective

The reason for using this wide selection of frameworks in our comparison is to exercise the different dimensions that together form FAIR Digital Objects: Data, Metadata, Service, Access, Operations, Computation. We have left out further comparisons on type systems, persistent identifiers and social aspects as principles and practices within these dimensions are still taking form within the FDO community (see section [sec:next-step-fdo?]).

Some of these frameworks invite a comparison on a conceptual level, while others relate better to implementations and current practices. For these we consider FAIR Digital Objects and the Web conceptually, and for implementations we contrast between the main FDO realization using the DOIPv2 protocol [12] against Linked Data in general.

Considering FDO/Web as interoperability framework for Fast Data

The Interoperability Framework for Fast Data Applications [60] categorizes 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 [63] better with regards to automated machine-to-machine interaction (and thus interoperability), which is a crucial aspect of the FAIR principles. In table [1] we use the interoperability framework to compare the current FAIR Digital Object approach against the Web and its Linked Data practices.

Table 1: Considering FDO and Web according to the quality levels of the Interoperability Framework for Fast Data [60].

Quality	FDO w/ DOIP	Web w/ Linked Data
---------	-------------	--------------------

Quality	FDO w/ DOIP	Web w/ Linked Data
Symbiotic: <i>to what extent multiple applications can agree to interact/align/collaborate/cooperate</i>	Purpose of FDO is to enable federated machine actionable digital objects for scholarly purposes, in practice this also requires agreement of or 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.	Web is loosely coupled and encourages collaboration and linking by URL. In practice, REST APIs [64] end up being mandated centrally by dominant (often commercial) providers [65], which clients are required to use as-is with special code per service. Use of Linked Data enables common tooling and semantic mapping across differences.
Pragmatic: <i>using interaction contracts so processes can be choreographed in workflows</i>	FDO types and operations enable detailed choreography (see CWFP). <code>⊖.TYPE/DOIPOperation</code> has lightweight definition of operation, <code>⊖.DOIP/Request</code> or <code>⊖.DOIP/Response</code> may give FDO Type or any other kind of “specifics” (incl. human readable docs). Semantics/purpose of operations not formalized (similar operations can be grouped with <code>⊖.DOIP/OperationReference</code>).	“Follow your nose” crawler navigation, which may lead to frequent dead ends. Operational composition, typically within a single API provider, documented by OpenAPI 3 [66], schema.org Actions [[67]], WSDL/SOAP [68], but frequently just as human-readable developer documentation/examples.
Semantic: <i>ensuring consistent understanding of messages, interoperability of rules, knowledge and ontologies</i>	FDO semantic enable navigation and typing. Every FDO have 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 & ontologies. JSON-LD used within some FDO objects (e.g. DISSCO Digital Specimen, NIST Material Science schema) [69]	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.
Syntactic: <i>serializing messages for digital exchange, structure representation</i>	DOIP serialize FDOs as JSON, metadata commonly use JSON, typed with JSON Schema. Multiple byte stream attachments of any media type.	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 previously main syntax used by APIs, JSON now dominant.
Connective: <i>transferring messages to another application, e.g. wrapping in other protocols</i>	DOIP [12] is transport-independent, commonly TLS TCP/IP port 9000), DOIP over HTTP	HTTP/1.1 (TCP/IP port 80), HTTP/1.1+TLS (TCP/IP 443), HTTP/2 (as HTTP/1* but binary), HTTP/3 (like HTTP/2+TLS but UDP)
Environmental: <i>how applications are deployed and affected by its environment, portability</i>	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).	HTTP services widely deployed in a myriad of ways, ranging from single instance servers, horizontally & vertically scaled application servers, to (for static content) multi-cloud Content-Delivery Networks (CDN). Current scalable cloud technologies for Web hosting may not support HTTP features previously seen as important for Semantic Web, e.g. content negotiation and semantic HTTP status codes.

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

Web have already showed us we can compose workflows of heterogeneous Web Services [70]. However, this is mostly done via developer or human interaction [71]. Similarly, FDO does not enable automatic composition because operation semantics are not well defined. There is a question as to whether the plethora of documentation and broad developer usage that is available for Web APIs can be developed for FDO.

A difference between Web 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 realized with DOIP requires JSON. On the semantic front, FDO requires 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 standards bodies (e.g. IETF and Web Consortium) are mainly open and have a diverse representation [1]. In contrast, FDO has a small number of implementations and corresponding protocols. This is not to say that they cannot be developed in the future, but we note that the functionality provided by FDO implementations can be easily implemented using Web technologies. It's also a question as to whether a highly constrained protocol revolving around persistent identifiers is in fact necessary. For example, DOIs are already implemented on the web [2].

Mapping of Metamodel concepts

The Interoperability Framework for Fast Data also provide 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 [72].

Table 2: Mapping the Metamodel concepts from the Interoperability Framework for Fast Data [60] to equivalent concepts for FDO and Web.

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

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 in section [sec:middleware?]), as well that HTTP has poor direct support for processes, as the Web is primarily stateless by design.

Assessing FDO implementations

The FAIR Digital Object guidelines [4] sets out recommendations for FDO implementations. In Table 3 we evaluate the two current implementations, using DOIPv2 [12] and using Linked Data Platform [73], as proposed by [74].

Table 3: Checking FDO guidelines [4] against its current implementations as DOIP [12] and Linked Data Platform (LDP) [74], with suggestions for required additions.

FDO Guideline	DOIP 2.0	FDO suggestions	Linked Data Platform	LDP suggestion
G1: <i>invest for many decades</i>	Handle system stable for 20 years, DOIP 2.0 since 2017.	Ensure FDO types will not be protocol-bound as DOIP might be updated/replaced	HTTP stable for 30 years, Semantic Web for 20 years. <code>http://</code> URIs replaced by <code>https://</code> .	Keep flexibility of RDF serialization formats which may change more frequently

FDO Guideline	DOIP 2.0	FDO suggestions	Linked Data Platform	LDP suggestion
<i>G2: trustworthiness</i>	DOI/Handle trusted by all major academic publishers and data repositories. DOIP relatively unknown, in effect only one implementation.	Further promote DOIP and justify its benefits. Build tutorials and OSI open source implementations. Standardize DOIP-over-HTTP alternative.	JSON-LD used by half of all websites [2], however previous bad experiences with Semantic Web may deter adapters	Ensure simplicity for end developers, rather than semantic overengineering. Example-driven documentation.
<i>G3: follows FAIR principles</i>	See table ??	Ensure all FAIR principles are covered, build complete examples.	Touched briefly, see table ??	Add explicit expression to show each FAIR principle fulfilled.
<i>G4: machine actionability</i>	CRUD and extension operations dynamically listed (see table @tbl:fdo-web-middleware)	Specify which operations should work for a given type, to reduce need for dynamic lookup. Specify input/output expectations formally (e.g. JSON Schema).	HTTP CRUD operations, Open API (see table @tbl:fdo-web-middleware)	Document operations so client can make subsequent HTTP calls.
<i>G5: abstraction principle</i>	Handle PIDs as abstraction base. DOIP operations can use any transport protocol.	Document transport protocols as FDOs, recommend which transport to use.	URI as abstraction base. Does not specify PID requirements.	Give stronger deployment recommendations.
<i>G6: stable binding between entities</i>	Machine-navigation through PIDs and operations specified per type. Unclear when metadata field is a PID or plain text.	Make datatype of fields explicit to support navigation.	Machine-navigation through URIs via properties and types. Unclear when URI should be followed or is just identifier, but always distinct from plain text.	
<i>G7: encapsulation</i>	Operations discovered at runtime (@.DOIP/Op.ListOperations).	Allow method discovery by type FDOs in advance (see PR- TypingFDOs-2.0-20220608).	HTTP methods discovered at runtime (OPTIONS), idempotent methods attempted directly. Unsupported methods reported using LDP constraints to human-readable text.	Declare supported methods in advance, e.g. OpenAPI [66]

FDO Guideline	DOIP 2.0	FDO suggestions	Linked Data Platform	LDP suggestion
G8: <i>technology independence</i>	In theory independent, in reality depends on single implementations of Handle system and DOIP	Encourage open source DOIP testbeds and lighter reference implementations	Multiple HTTP implementations, multiple LDP implementations. No FDOF implementations.	Develop demonstrator of FDOF usage based on existing LDP server.
G9: <i>standard compliance</i>	Handle [75], DOIP [12]. FDO requirements not standardized yet.	Formalize standard process of FDO requirements [WD-DOC?]	HTTP, LDP. FDOF not yet standardized	Formalize FDOF from FDOF-SEM working group
FDOF1: <i>PID as basis</i>	Extensive use of Handle system.	Clarify how local testing handles can be used during development, how “temporary” FDOs should evolve [PID? policy]. Register <code>0.DOIP/*</code> and <code>0.FDO/*</code> as PIDs.		
FDOF2: <i>PID record w/ type</i>	Unspecified how to resolve from Handle to DOIP Service (!), in practice <code>10320/loc</code> , <code>0.TYPE/DOIPService</code> , <code>URL</code> , <code>URL_REPLICA</code>	Document requirements for PID Record ()		
FDOF3: <i>PID resolvable to bytestream & metadata</i>	Byte stream resolvable (<code>0.DOIP/Retrieve</code>). No method/attribute defined for separate metadata, only directly in PID Record. Unclear meaning of multiple items and bytestream chunks.	Clarify expectations for multiple items. Recommend chunks to not be used.	URIs resolvable by default. Multiple ways to resolve metadata, unclear preference.	Add FAIR Signposting and preference order.
FDOF4: <i>Additional attributes</i>	Freetext attribute keys. Attributes should be defined for FDO type (?).	Require that attribute keys should be PIDs (or have predefined mapping to PIDs). Explicitly allow attributes not already defined in type.	All attributes individually identified. Any Linked Data attributes can be used by URI or with mapped prefix.	Clarify type expectations of required/recommended/optional attributes.

FDO Guideline	DOIP 2.0	FDO suggestions	Linked Data Platform	LDP suggestion
FDOF5: <i>Interface: operation by PID</i>	Extended operations use PID, but “pid-like” DOIP operations/types are not registered as handles.	Register <code>0.DOIP/*</code> and <code>0.FDO/*</code> as PIDs. Clarify that operations can be mapped to protocol directly.	CRUD operations used directly in HTTP (e.g. PUT). Unclear how to provide PID for additional operations.	Specify how additional operations should be called over HTTP.
FDOF6: <i>CRUD operations + extensions</i>	<code>0.DOIP/Op.Create</code> etc.		PUT, GET, POST, DELETE, PATCH, extensions uncommon.	
FDOF7: <i>FDOF Types related to operations</i>	Not yet formalized, only discoverable on a given FDO of type. WD-PID leaves this open.	Add explicit relation between type and operations		
FDOF8: <i>Metadata as FDO, semantic assertions</i>				
FDOF9: <i>Different metadata levels</i>				
FDOF10: <i>Metadata schemas by community</i>				
FDOF11: <i>FDO collections w/ semantic relations</i>				
FDOF12: <i>Deleted FDO preserve PID w/ tombstone</i>				

Note that the draft update to FDO specification [3] (see box [sec:next-step-fdo?]) clarifies these definitions with equivalent identifiers [6](#) and relates them to further FDO requirements such as FDO Data Type Registries.

Comparing FDO and Web as middleware infrastructures

Table 4: Comparing FAIR Digital Object (with the DOIP 2.0 protocol [12]) and Web technologies (using Linked Data) as middleware infrastructures [61]

Quality	FDO w/ DOIP	Web w/ Linked Data
Openness: <i>framework enable extension of applications</i>	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 .	The Web is inheritedly 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.
Scalability: <i>application should be effective at many different scales</i>	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)	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 manage by backend.
Performance: <i>efficient and predictable execution</i>	DOIP has been shown moderately scalable to 100 millions of objects, create operation at 900 requests/second [76]. DOIP protocol is reusable for many operations, multiple requests may be answered out of order (by <code>requestId</code>). Multiple connections possible. Setup is typically through TCP and TLS which adds latency.	HTTP traffic is about 10% of global Internet traffic, excluding video and social networks [77]. HTTP 1 connections are serial and reusable, and concurrent connections is common. HTTP/2 adds asynchronous responses and multiplexed streams [78] but still has TCP+TLS startup costs. For reduced latency [79], HTTP/3 [80] use QUIC [81] rather than TCP, already adapted heavily (30% of EMEA traffic) of which Instagram & Facebook video is the majority of traffic.

Quality	FDO w/ DOIP	Web w/ Linked Data
Distribution transparency: <i>application perceived as a consistent whole rather than independent elements.</i>	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.	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 [82], content-negotiation [83] or FAIR Signposting [7].
Access transparency: <i>local/remote elements accessed similarly</i>	FDOs always accessed through PID indirection, but this means difficult to make private test setup.	Global HTTP protocol frequently used locally and behind firewalls, but at risk of non-global URIs (e.g. <code>http://localhost/object/1</code>) and SSL issues (e.g. self-signed certificates, local CAs)
Location transparency: <i>elements accessed without knowledge of physical location</i>	FDOs always accessed through PIDs. Multiple locations possible in Handle system, can expose geo-info.	PIDs and URL redirects. DNS aliases and IP routing can hide location. Geo-localized servers common for large cloud deployments.
Concurrency transparency: <i>concurrent processing without interference</i>	No explicit concurrency measures. FDO kernel metadata can include checksum and date.	HTTP operations are classified as being stateless/idempotent or not (e.g. <code>PUT</code> changes state, but can be repeated on failure), although these constraints are occasionally violated by Web applications. Cache control, <code>ETag</code> (~ checksum) and modification date in HTTP headers allows detection of concurrent changes on a single resource.
Failure transparency: <i>service provisioning resilient to failures</i>	DOIP status codes, e.g. <code>0.DOIP/Status.104</code> , additional codes can be added as custom attributes	HTTP status codes e.g. <code>404 Not Found</code> , structured error documents in Open API (??)
Migration transparency: <i>allow relocating elements without interfering application</i>	Update of PID record URLs, indirection through <code>0.TYPE/DOIPServiceInfo</code> (not always used consistently). No redirection from DOIP service.	HTTP <code>30x</code> status codes provide temporary or permanent redirections, commonly used for PURLs but also by endpoints.
Persistence transparency: <i>conceal deactivation/reactivation of elements from their users</i>	FDO requires use of PIDs for object persistence, including a thumbstone 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).	URLs are not required to persist, although encouraged [84]. Persistence requires convention to use PIDs/PURLs and HTTP <code>410 Gone</code> . An URL may change its content, change in type may sometimes force new URLs if exposing extensions like <code>.json</code> . Memento [85] expose versioned snapshots. WebDAV <code>VERSION-CONTROL</code> method [86] (used by SVN).
Transaction transparency: <i>coordinate execution of atomic/isolated transactions</i>	No transaction capabilities declared by FDO or DOIP. Internal synchronization possible in backend for Extended operations.	Limited transaction capabilities (e.g. <code>If-Unmodified-Since</code>) on same resource. WebDAV locking mechanisms [87] with <code>LOCK</code> and <code>UNLOCK</code> methods.
Modularity: <i>application as collection of connected/distributed elements</i>	FDOs are inheritedly 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.	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.

Quality	FDO w/ DOIP	Web w/ Linked Data
Encapsulation: <i>separate interface from implementation. Specify interface as contract, multiple implementations possible</i>	Indirection by PID gives separation. FDO principles are protocol independent, although it may be unclear which protocol to use for which FDO (although <code>0.DOIP/Transport</code> can be specified after already contacting DOIP). Cordra supports native DOIP , DOIP over HTTP and Cordra REST API	HTTP/1.1 semantics can seamlessly upgrade to HTTP/2 and HTTP/3. <code>http</code> vs <code>https</code> URIs exposes encryption detail 7 . Implementation details may leak into URIs (e.g. <code>search.aspx</code>), countered by deliberate design of URI patterns [89] and PIDs via Persistent URLs (PURL).
Inheritance: <i>Deriving specialized interface from another type</i>	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 <code>\$ref</code> use of JSON Schema references [26])	Syntactically Media Type with multiple suffixes [90] (mainly used with <code>+json</code>), declaration of subtypes as profiles (RFC6906) [91] . In metadata, semantic type systems (RDFS [40] , OWL2 [41] , SKOS [33]). OpenAPI 3 [66] inheritance and Polymorphism . XML <code>xsd:schemaLocation</code> & <code>xsd:type</code> [92] , JSON <code>\$schema</code> [26] , JSON-LD <code>@context</code> [93] . Large number of domain-specific and general ontologies define semantic types, but finding and selecting remains a challenge.
Signal interfaces: <i>asynchronous handling of messages</i>	DOIP 2.0 is synchronous, in FDO async operations undefined. Could be handled as custom jobs/futures FDOs	HTTP/2 multiplexed streams [78] , Web Sockets [94] , Linked Data Notifications [95] , AtomPub [96] , SWORD [97] , Micropub, more typically ad-hoc jobs/futures REST resources
Operation interfaces: <i>defining operations possible on an instance, interface of request/response messages</i>	CRUD predefined in DOIP, custom operations through <code>0.DOIP/Op.ListOperations</code> (can be FDOs of type <code>0.TYPE/DOIPOperation</code> , more typically local identifiers like "getProvenance")	CRUD predefined in HTTP methods [98] , (extended by registration), URI Templates [82] , OpenAPI operations [66] , HATEOAS ⁸ incl. Hydra [99] , schema.org Actions [67] , JSON HAL [100] & Link headers (RFC8288) [101]
Stream interfaces: <i>operations that can handle continuous information streams</i>	Undefined in FDO. DOIP can support multiple byte stream elements (need custom FDO type to determine stream semantics)	HTTP 1.1 [102] chunked transfer , HLS (RFC8216) [103] , MPEG-DASH (ISO/IEC 23009-1:2019) [104]

...

As for the aspect of *Performance*, it is interesting to note that while the first version of DOIP [\[22\]](#) supported multiplexed channels similar to HTTP/2, allowing concurrent transfer of several digital objects. However multiplexing was removed for the much simplified DOIP 2.0 [\[12\]](#), which do support multiple asynchronous requests, but unlike DOIP 1.0 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 ⁹.

Assessing FDO against FAIR

Table 5: Assessing RDA's FAIR Data Maturity Model [\[62,105\]](#) (first 2 columns) against the FDO guidelines [\[4\]](#), FDO implemented as DOIP [\[12\]](#), Linked Data Platform [\[74\]](#) and Linked Data practices. –: Unspecified.

FAIR ID	Indicator	FDO guidelines	FDO/DOIP	FDO/LDP	Linked Data examples
---------	-----------	----------------	----------	---------	----------------------

FAIR ID	Indicator	FDO guidelines	FDO/DOIP	FDO/LDP	Linked Data examples
RDA-F1-01M	Metadata is identified by a persistent identifier	FDOF4	Optional <i>Metadata FDO</i> w/separate PID	Content-negotiation to URL, not required to be PID	Metadata typically don't have own PID
RDA-F1-01D	Data is identified by a persistent identifier	FDOF1	PIDs required (FDOF1). Handle, DOI.	FDOF-IR (Identifier Record). PID can be any URI?	“Cool” URIs [84] , PURL services incl. <code>purl.org</code> , <code>w3id.org</code>
RDA-F1-02M	Metadata is identified by a globally unique identifier	FDOR4 FDOF8	Optional <i>Metadata FDO</i>	Content-negotiation to URL	Not required, content-negotiation can redirect to URL or <code>Content-Location</code> . FAIR Signposting.
RDA-F1-02D	Data is identified by a globally unique identifier	FDOF1	All FDOs have PIDs (FDOR1), DOIP uses Handle system	FDOF-IR (Identifier Record)	Always accessed by URL
RDA-F2-01M	Rich metadata is provided to allow discovery	FDOF2 FDOF4 FDOF8 FDOF9	FDO has key-value metadata. Unclear how to link to additional metadata.	FDOF-IR links to multiple metadata records	RDF-based metadata by content negotiation or FAIR Signposting. Embedded in landing page (RDFa).
RDA-F3-01M	Metadata includes the identifier for the data	–	<code>id</code> and <code>type</code> are required metadata elements PIDs, also implicit as requests must use PID	PID only required in FDOF-IR record.	PID inclusion typical, but often inconsistent (e.g. <code>www.example.com</code> vs <code>example.com</code>) or missing (use of <code><></code> as <i>this</i> subject)
RDA-F4-01M	Metadata is offered in such a way that it can be harvested and indexed	FDOF10	No, registries not required (except Data Type Registries). Handle registry only searchable by PID.	Not specified	Not specified, several registries/catalogues for vocabularies/types (e.g. [106]). Indexing by search engines if exposing HTML w/schema.org.
RDA-A1-01M	Metadata contains information to enable the user to get access to the data	FDOF3 FDOF6	?doip	?ldp	?ld
RDA-A1-02M	Metadata can be accessed manually (i.e. with human intervention)	–	?doip	?ldp	?ld
RDA-A1-02D	Data can be accessed manually (i.e. with human intervention)	–	?doip	?ldp	?ld
RDA-A1-03M	Metadata identifier resolves to a metadata record	FDOF8+F DOF2	?doip	?ldp	?ld
RDA-A1-03D	Data identifier resolves to a digital object	FDOF2	?doip	?ldp	?ld

FAIR ID	Indicator	FDO guidelines	FDO/DOIP	FDO/LDP	Linked Data examples
RDA-A1-04M	Metadata is accessed through standardised protocol	G9 FDOF3	Retrievable from PID (FDOF3). Informal DOIP standard maintained by DONA Foundation	HTTP standards maintained by IETF, FDO components resolved by informal proposals (custom vocabulary, extra HTTP methods) or HTTP content negotiation)	Formal HTTP standards maintained by IETF, HTTP content negotiation, FAIR Signposting
RDA-A1-04D	Data is accessible through standardised protocol	G9	?doip	?o	?ld
RDA-A1-05D	Data can be accessed automatically (i.e. by a computer program)	G4 FDOF3 FDOF6	?doip	?ldp	?ld
RDA-A1.1-01M	Metadata is accessible through a free access protocol	G1 G8 G9	Partially realized: Handle system is open protocol [[107]]¹⁰ . One server implementation [108] , free but license not OSI-approved [109] . One DOIPv2 implementation (CORDRA): free under BSD-like license (not recognized as Open Source).	LDP is open W3C recommendation. Multiple LDP implementations .	DNS, HTTP, TLS, RDF standards are open, free and universal, large number of Open Source clients and servers .
RDA-A1.1-01D	Data is accessible through a free access protocol	G9	?doip	?ldp	?ld
RDA-A1.2-01D	Data is accessible through an access protocol that supports authentication and authorisation	(FDOR9)	TLS certificates, authentication field (details unspecified)	Unspecified	HTTP authentication, TLS certificates
RDA-A2-01M	Metadata is guaranteed to remain available after data is no longer available	FDOF12	FDO thumbstone required	Unspecified, however FDOF-IR links to separate metadata records	Unspecified
RDA-I1-01M	Metadata uses knowledge representation expressed in standardised format	FDOF8	Required, but not currently defined	Unspecified	Always implied by use of RDF syntaxes.
RDA-I1-01D	Data uses knowledge representation expressed in standardised format	?fdo	?doip	?ldp	?ld
RDA-I1-02M	Metadata uses machine-understandable knowledge representation	FDOF8	Required	Unspecified	Always implied by use of RDF syntaxes.

FAIR ID	Indicator	FDO guidelines	FDO/DOIP	FDO/LDP	Linked Data examples
RDA-I1-02D	Data uses machine-understandable knowledge representation	G4 G7 FDOR2	No requirements on binary data formats	LDP Basic Container reference only information resources	?ld
RDA-I2-01M	Metadata uses FAIR-compliant vocabularies	G3 FDOF10	Informally required	Unspecified, implied by use of RDF?	FAIR practices for LD vocabularies increasingly common, sometimes inconsistent (e.g. PURLs that don't resolve) or incomplete (e.g. unknown license)
RDA-I2-01D	Data uses FAIR-compliant vocabularies	–	?doip	?ldp	?ld
RDA-I3-01M	Metadata includes references to other metadata	FDOR8	Implied (attributes to PIDs), currently unspecified if given attribute is value or reference	Unspecified	By definition (Linked Data reference existing URIs [110]), <code>rdfs:seeAlso</code> , FAIR signposting [7] described by
RDA-I3-01D	Data includes references to other data	G6 FDOR3 FDOR11	?doip	?ldp	?ld
RDA-I3-02M	Metadata includes references to other data	G6 FDOR3 FDOR8	?doip	?ldp	?ld
RDA-I3-02D	Data includes qualified references to other data	FDOR3 FDOR11	?doip	?ldp	?ld
RDA-I3-03M	Metadata includes qualified references to other metadata	(FDOR3)	Qualification by attribute key	LDP Direct Container	Qualifications by property, PROV bundles [111] ,
RDA-I3-04M	Metadata include qualified references to other data	(FDOR3)	Qualification by attribute key	LDP Indirect Container	Qualifications by property, n-ary indirection (schema.org Role [112] , <code>prov:specializationOf</code> [113] , OAI-ORE Proxy [114])
RDA-R1-01M	Plurality of accurate and relevant attributes are provided to allow reuse	FDOF4	Required. Kernel metadata attributes desired, not yet decided.	Unspecified. Multiple metadata records can allow multiple semantic profiles.	Large number of general and domain-specific vocabularies can make it hard to find relevant attributes. Rough consensus on kernel metadata: schema.org [115] , Dublin Core Terms [116] , DCAT [117] , FOAF [118]
RDA-R1.1-01M	Metadata includes information about the licence under which the data can be reused	?	Unspecified (but will be in PID Kernel metadata?)	Unspecified	Dublin Core Terms <code>dct:license</code> frequently recommended, but not required, e.g. by DCAT 2 [117]
RDA-R1.1-02M	Metadata refers to a standard reuse licence	–	?doip	?ldp	?ld

FAIR ID	Indicator	FDO guidelines	FDO/DOIP	FDO/LDP	Linked Data examples
RDA-R1.1-03M	Metadata refers to a machine-understandable reuse licence	–	?doip	?ldp	SPDX
RDA-R1.2-01M	Metadata includes provenance information according to community-specific standards	FDOR9 FDOR10	Unspecified (some CORDRA types add getProvenance methods). PID Kernel attributes? Unspecified W3C PROV-O, PAV		
RDA-R1.2-02M	Metadata includes provenance information according to a cross-community language	FDOR9 FDOR8	?doip	?ldp	?ld
RDA-R1.3-01M	Metadata complies with a community standard	FDOR10 FROR8	?doip	?ldp	?ld
RDA-R1.3-01D	Data complies with a community standard	(FDOR3)	?doip	?ldp	?ld
RDA-R1.3-02M	Metadata is expressed in compliance with a machine-understandable community standard	FDOF4 FDOF10	Recommended	Unspecified	Common practice, specially in bioinformatics, e.g. BioSchemas [119], BioPortal [106]
RDA-R1.3-02D	Data is expressed in compliance with a machine-understandable community standard	(FDOR2)	No, FDO is typed but data can be any bytestream	Unspecified	Common, but not required (e.g. GFF3 , FITS , ESRI)

EOSC Interoperability Framework

TODO: Introduce EOSC IF

The EOSC Interoperability Framework [[24](#)] (...), and in section 3.6 recommends:

TODO: Draft table

Layer	Recommendation	FDO	Linked Data
Technical	Open Specification		
Technical	Common security&privacy framework		
Technical	Easy SLAs for service providers	None	None

Layer	Recommendation	FDO	Linked Data
Technical	Access data in different formats	None formalized, custom operations or relations	Content-negotiation, rel=alternate relations
Technical	Coarse-grained/fine-grained search tools	None	?
Technical	Clear PID policy	FDO requirements [], tends towards Handle system.	Not required, different communities set policies
Semantic	Clear definitions for concepts/metadata/schemas	FDO requirements[], not yet formalized	Ontologies, SKOS, ...
Semantic	Semantic artefacts w/ open licenses	All artefacts are PIDs w/ license required by kernel metadata?	Open License is best practice for ontology publishing
Semantic	Documentation for each semantic artefact	No direct rendering from FDO, no requirement for human-readable description	Ontology rendering, content-negotiation
Semantic	Repositories of artefacts	Required, but not formalized	Bioontologies, etc
Semantic	Repositories w/ clear governance	Recommended	Largely self-governed repositories, if well-established may have clear governance.
Semantic	Minimal metadata model for federated discovery	Kernel metadata	DCAT, ++
Semantic	Crosswalks from minimal metadata model	Can be part of definition?	Several crosswalks
Semantic	Extensibility options for disciplinary metadata	Communities encouraged to establish own types	Extensible by design, domain-specific metadata may be at different granularity
Semantic	Clear protocols/building blocks for federation/harvesting of artefact catalogues	Collection types not yet defined	SWORD, OAI-PMH
Organisational	Interoperability-focused rules of participation recommendations		
Organisational	Usage recommendations of standardised data formats	None	
Organisational	Usage recommendations of vocabularies	None	RDMKit
Organisational	Usage recommendations of metadata.	None	RO-Crate, Bioschemas
Organisational	Management of permanent organization names/functions	Handle owner, but unclear contact. Contact info in DOIP service provider	ROR. DCAT contacts.
Legal	Standardised human and machine-readable licenses	None	SPDX
Legal	Permissive licenses for metadata (CC0, CC-BY-4.0)	Undefined	Both CC0, CC-BY-4.0 common, e.g. in DCAT.
Legal	Different licenses for different parts	Each part as separate FDO can have separate license.	DCAT, RO-Crate
Legal	Mark expired/inexistent copyright	Undefined	Unclear, semantics assume copyright valid
Legal	Mark orphaned data	Thumbstone for deleted data, but no owner means FDO disappears	

Layer	Recommendation	FDO	Linked Data
Legal	List recommended licenses	Undefined	Best practice recommendations
Legal	Track license evolution for dataset	Undefined	Versioning with PAV/PROV/DCAT
Legal	Policy/guidance for patent/trade secrets violation	Undefined	Undefined, legal owner may be specified
Legal	GDPR compliance for personal data		
Legal	Restrict access/use if legally required	By transport protocol (undefined by FDO/DOIP)	Diverging approaches, typically landing pages w/ auth&auth or click-thru
Legal	Harmonized terms-of-use	Undefined	Undefined
Legal	Alignment between EOSC and national legislation	Not applicable	Not applicable

Discussion

TODO

(What does it 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 [120]. However, our deeper investigation, based on multiple frameworks, highlights that the openness and variability of how Linked Data is deployed makes it difficult to achieve the FDO goals without significant effort.

While RDF and Linked Data has 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.

We therefore identify the need for an 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 will utilize 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 a rigid (yet flexible) ecosystem of machine-actionable scholarly objects.

Simplifying the Semantic Web Architecture

- [121] updating Semantic Web Architecture
- [122] ?

Kingsley Uyi Idehen points out that the traditional Semantic Web Cake has a serious flaw in implying that XML is a requirement and points out that the Semantic Web is *not* distinct from the World Wide Web [123] .

Below we reconsider the Semantic Web Architecture as updated by Kingsley, and simplify it with the most obvious technical choices as of 2022 to align it with the main FDO requirements:

- Semantic Web of Linked Data: Use HTTP 1.1 or newer. HTTP resolution of PID must redirect to JSON-LD, optionally via a HTML landing page with FAIR Signposting link headers.

- Document Type: Use JSON-LD 1.1 or newer. `@context` must resolve by versioned PID¹¹. Use flattened, compacted profile of JSON-LD for consistent JSON.
- Sentence Part identifiers: Use PIDs and `#` -fragments of PIDs, allowing other URLs for non-FDO references. Use DOIs or Handle
- Abstract language: Use a single graph. Explicitly declare `@type` and `s:name` of all subjects.
- Query: Use SPARQL 1.1 or newer.
- Dictionary: Use `https://schema.org/` as base vocabulary, extend using PID to vocabularies in SKOS.
- Rules: Use SHACL Core in JSON-LD
- Transmission: Always use TLS and `https://` URIs (including in new PIDs¹²). Use services like Let's Encrypt for ad-hoc certificates.

As technological improvements to Semantic Web continue, and FDO requirements mature, this FDO-LD profile will need to be regularly revised, for instance every 3 years.

Random Notes

(Likely to be deleted from paper)

- <https://www.nist.gov/programs-projects/facilitating-adoption-fair-digital-object-framework-material-science>
 - <https://github.com/usnistgov/mgi-json-schema/>
 - <https://pages.nist.gov/material-schema/>
- <https://doi.org/20.5000.1025/ZZX7-CEFZ>
 - <https://sandbox.dissco.tech/#objects/test/448aa5396edcee5940e4>

References

1. **FAIR Digital Objects Forum** | <https://fairdo.org/>
2. **FDO Forum Document Standards**
C Weiland, U Schwardmann, P Wittenburg, C Kirkpatrick, R Hanisch, Z Trautt
FDO Forum (2022-01-29)
3. **FDO Requirement Specifications**
FDO-TSIG working group
FDO Forum (2022-03-17)
4. **FAIR digital object framework**
Luiz Bonino, Oeter Wittenburg, Bonnie Carroll, Alex Hardisty, Mark Leggott, Carlo Zwölf
FDOF technical implementation guideline (2019-11-22) {<https://github.com/GEDE-RDA-Europe/GEDE/blob/master/FAIR%20Digital%20Objects/FDOF/FAIR%20Digital%20Object%20Framework-v1-02.docx>}
5. **FDO Machine Actionability**
Claus Weiland, Sharif Islam, Daan Broder, Ivonne Anders, PEter Wittenburg
FDO Forum (2022-02-25)
6. **FDO Configuration Types**
Larry Lannom, Karsten Peters-von Gehlen, Ivonne Anders, Andreas Pfeil, Alexander Schlemmer, Zach Trautt, Peter Wittenburg
FDO Forum (2022-03-17)
7. **FAIR Signposting Profile - Signposting the Scholarly Web** <https://signposting.org/FAIR/>
8. **FDO PID Profiles & Attributes**
Ivonne Anders, Maggie Hellström, Sharif Islam, Thomas Jeikal, Larry Lannom, Ulrich Schwardmann, Peter Wittenburg
FDO Forum (2022-03-17)
9. **RDA Recommendation on PID Kernel Information**
Tobias Weigel, Beth Plale, Mark Parsons, Gabriel Zhou, Yu Luo, Ulrich Schwardmann, Robert Quick, Margareta Hellström, Kei Kurakawa
Research Data Alliance (2018) <https://doi.org/gp5fpd>
DOI: [10.15497/rda00031](https://doi.org/10.15497/rda00031)
10. **FDO – Granularity, Versioning, Mutability**
FDO-TSIG Working Group
FDO Forum (2022-03-17)
11. **DOIP endorsement request**
FDO-TSIG Working Group
FDO Forum (2022-03-26)
12. **Digital object interface protocol specification, version 2.0**
DONA Foundation
DONA foundation (2018-11-12) <https://hdl.handle.net/0.DOIP/DOIPV2.0>
13. **FAIR Digital Object Demonstrators 2021**
Peter Wittenburg, Ivonne Anders, Christophe Blanchi, Merret Buurman, Carole Goble, Jonas Grieb, Alex Hardisty, Sharif Islam, Thomas Jeikal, Tibor Kálmán, ... Philipp Wieder

Zenodo (2022-01-18) <https://doi.org/gp4wm4>

DOI: [10.5281/zenodo.5872645](https://doi.org/10.5281/zenodo.5872645)

14. **Upload of FDO**

Christophe Blanchi, Daan Broeder, Thomas Jejkal, Islam Sharif, Alexander Schlemmer, Dieter van Uytvanck, Peter Wittenburg

FDO Forum (2022-03-20)

15. **ResourceSync Framework Specification - Table of Contents** <http://www.openarchives.org/rs/toc>

16. **Typing FAIR Digital Objects**

Larry Lannom, U Schwardmann, C Blanchi, P Wittenburg

FDO Forum (2022-03-10)

17. **FAIR principles and digital objects: Accelerating convergence on a data infrastructure**

Erik Schultes, Peter Wittenburg

Data analytics and management in data intensive domains: 20th international conference, DAMDID/RCDL 2018, moscow, russia, october 9–12, 2018, revised selected papers (2019)

DOI: [10.1007/978-3-030-23584-0_1](https://doi.org/10.1007/978-3-030-23584-0_1) · ISBN: 978-3-030-23583-3

18. **The FAIR Guiding Principles for scientific data management and stewardship**

Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, ... Barend Mons

Scientific Data (2016-03-15) <https://doi.org/bdd4>

DOI: [10.1038/sdata.2016.18](https://doi.org/10.1038/sdata.2016.18) · PMID: [26978244](https://pubmed.ncbi.nlm.nih.gov/26978244/) · PMCID: [PMC4792175](https://pubmed.ncbi.nlm.nih.gov/PMC4792175/)

19. **A framework for distributed digital object services**

Robert Kahn, Robert Wilensky

International Journal on Digital Libraries (2006-04)

https://www.doi.org/topics/2006_05_02_Kahn_Framework.pdf

DOI: [10.1007/s00799-005-0128-x](https://doi.org/10.1007/s00799-005-0128-x)

20. **A framework for distributed digital object services**

Robert Kahn, Robert Wilensky

CNRI (1995-05-13) <http://www.cnri.reston.va.us/k-w.html>

21. **X.1255 : Framework for discovery of identity management information** <https://www.itu.int/rec/T-REC-X.1255-201309-I>

22. **Digital Object Interface Protocol Version 1.0 | DONA Foundation** <https://www.dona.net/doipv1doc>

23. **Digital objects as drivers towards convergence in data infrastructures**

Peter Wittenburg, George Strawn, Barend Mons, Luiz Bonino, Erik Schultes

<https://b2share.eudat.eu> (2019-01-06)

DOI: [10.23728/b2share.b605d85809ca45679b110719b6c6cb11](https://doi.org/10.23728/b2share.b605d85809ca45679b110719b6c6cb11)

24. **EOSC interoperability framework**

Oscar Corcho, Magnus Eriksson, Krzysztof Kurowski, Milan Ojsteršek, Christine Choirat, Mark van de Sanden, Frederik Coppens

Publications Office of the EU (2021-02-05) <https://doi.org/10.2777/620649>

DOI: [10.2777/620649](https://doi.org/10.2777/620649)

25. **DOIP and Examples — Cordra documentation** <https://www.cordra.org/documentation/api/doip.html>

26. **draft-bhutton-json-schema-00** <https://datatracker.ietf.org/doc/html/draft-bhutton-json-schema-00>

27. **Weaving the Web: the original design and ultimate destiny of the World Wide Web by its inventor**
Tim Berners-Lee, Mark Fischetti
HarperSanFrancisco (1999)
ISBN: 9780062515865
28. **RDF 1.1 Primer** <http://www.w3.org/TR/rdf11-primer/>
29. **Resource Description Framework (RDF) Model and Syntax Specification** <https://www.w3.org/TR/1999/REC-rdf-syntax-19990222/>
30. **Process modelling for information system description**
Stefan K Stanczyk
The Open University (1987) <https://doi.org/gp6znp>
DOI: [10.21954/ou.ro.0000f821](https://doi.org/10.21954/ou.ro.0000f821)
31. **Cool URIs for the semantic web**
Leo Sauermann, Richard Cyganiak, Max Völkel
Universität des Saarlandes (2011-07-13) <https://doi.org/gp6znq>
DOI: [10.22028/d291-25086](https://doi.org/10.22028/d291-25086)
32. **SPARQL 1.1 Overview** <http://www.w3.org/TR/sparql11-overview/>
33. **SKOS Simple Knowledge Organization System Primer** <http://www.w3.org/TR/skos-primer/>
34. **How Matchable Are Four Thousand Ontologies on the Semantic Web**
Wei Hu, Jianfeng Chen, Hang Zhang, Yuzhong Qu
Lecture Notes in Computer Science (2011) <https://doi.org/bwdp52>
DOI: [10.1007/978-3-642-21034-1_20](https://doi.org/10.1007/978-3-642-21034-1_20)
35. **API-centric Linked Data integration: The Open PHACTS Discovery Platform case study**
Paul Groth, Antonis Loizou, Alasdair JG Gray, Carole Goble, Lee Harland, Steve Pettifer
Journal of Web Semantics (2014-12) <https://doi.org/f6wxhf>
DOI: [10.1016/j.websem.2014.03.003](https://doi.org/10.1016/j.websem.2014.03.003)
36. **A systematic analysis of term reuse and term overlap across biomedical ontologies**
Maulik R Kamdar, Tania Tudorache, Mark A Musen
Semantic Web (2017-08-07) <https://doi.org/gbskfd>
DOI: [10.3233/sw-160238](https://doi.org/10.3233/sw-160238) · PMID: [28819351](https://pubmed.ncbi.nlm.nih.gov/28819351/) · PMCID: [PMC5555235](https://pubmed.ncbi.nlm.nih.gov/PMC5555235/)
37. **Scholarly Context Not Found: One in Five Articles Suffers from Reference Rot**
Martin Klein, Herbert Van de Sompel, Robert Sanderson, Harihar Shankar, Lyudmila Balakireva, Ke Zhou, Richard Tobin
PLoS ONE (2014-12-26) <https://doi.org/brcc>
DOI: [10.1371/journal.pone.0115253](https://doi.org/10.1371/journal.pone.0115253) · PMID: [25541969](https://pubmed.ncbi.nlm.nih.gov/25541969/) · PMCID: [PMC4277367](https://pubmed.ncbi.nlm.nih.gov/PMC4277367/)
38. **A more decentralized vision for Linked Data**
Axel Polleres, Maulik Rajendra Kamdar, Javier David Fernández, Tania Tudorache, Mark Alan Musen
Semantic Web (2020-01-31) <https://doi.org/gkdcmh>
DOI: [10.3233/sw-190380](https://doi.org/10.3233/sw-190380)
39. **The Landscape of Ontology Reuse Approaches**
Valentina Anita Carriero, Marilena Daquino, Aldo Gangemi, Andrea Giovanni Nuzzolese, Silvio Peroni, Valentina Presutti, Francesca Tomasi
Applications and Practices in Ontology Design, Extraction, and Reasoning (2020-11-12) <https://doi.org/gntv2w>
DOI: [10.3233/ssw200033](https://doi.org/10.3233/ssw200033)

40. **RDF Schema 1.1** <http://www.w3.org/TR/rdf-schema/>
41. **OWL 2 Web Ontology Language Document Overview (Second Edition)** <http://www.w3.org/TR/owl2-overview/>
42. **Mapping between the OBO and OWL ontology languages**
Syed Tirmizi, Stuart Aitken, Dilvan A Moreira, Chris Mungall, Juan Sequeda, Nigam H Shah, Daniel P Miranker
Journal of Biomedical Semantics (2011) <https://doi.org/bn3fsc>
DOI: [10.1186/2041-1480-2-s1-s3](https://doi.org/10.1186/2041-1480-2-s1-s3) · PMID: [21388572](https://pubmed.ncbi.nlm.nih.gov/21388572/) · PMCID: [PMC3105495](https://pubmed.ncbi.nlm.nih.gov/PMC3105495/)
43. **Linked Data - The Story So Far**
Christian Bizer, Tom Heath, Tim Berners-Lee
International Journal on Semantic Web and Information Systems (2009-07-01) <https://doi.org/fc8zjt>
DOI: [10.4018/jswis.2009081901](https://doi.org/10.4018/jswis.2009081901)
44. **Linked Data - Design Issues** <https://www.w3.org/DesignIssues/LinkedData.html>
45. **The Open Graph protocol** <https://ogp.me/>
46. **RDFa 1.1 Primer - Third Edition** <http://www.w3.org/TR/rdfa-primer/>
47. **HTML Standard** <https://html.spec.whatwg.org/multipage/microdata.html>
48. **JSON-LD 1.1** <https://www.w3.org/TR/json-ld/>
49. **Usage Statistics of JSON-LD for Websites, August 2022** <https://w3techs.com/technologies/details/da-jsonld>
50. **Designing a Linked Data developer experience** (2018-12-28)
<https://ruben.verborgh.org/blog/2018/12/28/designing-a-linked-data-developer-experience/>
51. **Shapes Constraint Language (SHACL)** <https://www.w3.org/TR/shacl/>
52. **Shape Expressions (ShEx) 2.1 Primer** <http://shex.io/shex-primer/>
53. **Using Shape Expressions (ShEx) to Share RDF Data Models and to Guide Curation with Rigorous Validation**
Katherine Thornton, Harold Solbrig, Gregory S Stupp, Jose Emilio Labra Gayo, Daniel Mietchen, Eric Prud'hommeaux, Andra Waagmeester
The Semantic Web (2019) <https://doi.org/gnd3bq>
DOI: [10.1007/978-3-030-21348-0_39](https://doi.org/10.1007/978-3-030-21348-0_39)
54. **Validating RDF Data**
Jose Emilio Labra Gayo, Eric Prud'hommeaux, Iovka Boneva, Dimitris Kontokostas
Springer International Publishing (2018) <https://doi.org/ghks5j>
DOI: [10.2200/s00786ed1v01y201707wbe016](https://doi.org/10.2200/s00786ed1v01y201707wbe016)
55. **Survey of tools for Linked Data consumption**
Jakub Klímek, Petr Škoda, Martin Nečaský
Semantic Web (2019-05-23) <https://doi.org/gp6znr>
DOI: [10.3233/sw-180316](https://doi.org/10.3233/sw-180316)
56. **The Semantic Web identity crisis: In search of the trivialities that never were**
Ruben Verborgh, Miel Vander Sande
Semantic Web Journal (2020-01) <https://ruben.verborgh.org/articles/the-semantic-web-identity-crisis/>
DOI: [10.3233/sw-190372](https://doi.org/10.3233/sw-190372)
57. **RightField: embedding ontology annotation in spreadsheets**
K Wolstencroft, S Owen, M Horridge, O Krebs, W Mueller, JL Snoep, F du Preez, C Goble
Bioinformatics (2011-05-26) <https://doi.org/b4xvb2>

DOI: [10.1093/bioinformatics/btr312](https://doi.org/10.1093/bioinformatics/btr312) · PMID: [21622664](https://pubmed.ncbi.nlm.nih.gov/21622664/)

58. **REST and Linked Data**
Kevin R Page, David C De Roure, Kirk Martinez
Proceedings of the Second International Workshop on RESTful Design - WS-REST '11 (2011) <https://doi.org/bv3fzq>
DOI: [10.1145/1967428.1967435](https://doi.org/10.1145/1967428.1967435)
59. **On Schema.org and Why It Matters for the Web**
Peter Mika
IEEE Internet Computing (2015-07) <https://doi.org/gp5dvm>
DOI: [10.1109/mic.2015.81](https://doi.org/10.1109/mic.2015.81)
60. **An Interoperability Framework and Distributed Platform for Fast Data Applications**
José Carlos Martins Delgado
Data Science and Big Data Computing (2016) <https://doi.org/gp3rds>
DOI: [10.1007/978-3-319-31861-5_1](https://doi.org/10.1007/978-3-319-31861-5_1)
61. **A Comparison Framework for Middleware Infrastructures.**
Apostolos Zarras
The Journal of Object Technology (2004) <https://doi.org/cj5q8r>
DOI: [10.5381/jot.2004.3.5.a2](https://doi.org/10.5381/jot.2004.3.5.a2)
62. **The FAIR data maturity model: An approach to harmonise FAIR assessments**
Christophe Bahim, Carlos Casorrán-Amilburu, Makx Dekkers, Edit Herczog, Nicolas Loozen, Konstantinos Repanas, Keith Russell, Shelley Stall
Data Science Journal (2020-10-27)
DOI: [10.5334/dsj-2020-041](https://doi.org/10.5334/dsj-2020-041)
63. **Handbook of computer-communications standards: The open systems (OSI) model and OSI-related standards**
William Stallings
Sams (1990)
ISBN: 9780672226977
64. **Architectural styles and the design of network-based software architectures**
Roy Thomas Fielding
University of California, Irvine (2000) <https://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>
65. **Reflections on the REST architectural style and "principled design of the modern web architecture" (impact paper award)**
Roy T Fielding, Richard N Taylor, Justin R Erenkrantz, Michael M Gorlick, Jim Whitehead, Rohit Khare, Peyman Oreizy
Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering (2017-08-21)
<https://doi.org/gfk22x>
DOI: [10.1145/3106237.3121282](https://doi.org/10.1145/3106237.3121282)
66. **OpenAPI Specification v3.1.0 | Introduction, Definitions, & More** <https://spec.openapis.org/oas/v3.1.0.html>
67. **Schema.org Actions - schema.org** <https://schema.org/docs/actions.html>
68. **Web Services Description Language (WSDL) Version 2.0 Part 0: Primer** <http://www.w3.org/TR/wsdl20-primer/>
69. **FAIR digital object demonstrators 2021**
Peter Wittenburg, Ivonne Anders, Christophe Blanchi, Merret Buurman, Carole Goble, Jonas Grieb, Alex Hardisty, Sharif Islam, Thomas Jejkal, Tibor Kálmán, ... Philipp Wieder
Zenodo (2022) <https://zenodo.org/record/5872645>
DOI: [10.5281/zenodo.5872645](https://doi.org/10.5281/zenodo.5872645)

70. **The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud**
Katherine Wolstencroft, Robert Haines, Donal Fellows, Alan Williams, David Withers, Stuart Owen, Stian Soiland-Reyes, Ian Dunlop, Aleksandra Nenadic, Paul Fisher, ... Carole Goble
Nucleic Acids Research (2013-05-02) <https://doi.org/ggbw4>
DOI: [10.1093/nar/gkt328](https://doi.org/10.1093/nar/gkt328) · PMID: [23640334](https://pubmed.ncbi.nlm.nih.gov/23640334/) · PMCID: [PMC3692062](https://pubmed.ncbi.nlm.nih.gov/PMC3692062/)
71. **Perspectives on automated composition of workflows in the life sciences**
Anna-Lena Lamprecht, Magnus Palmblad, Jon Ison, Veit Schwämmle, Mohammad Sadnan Al Manir, Ilkay Altintas, Christopher JO Baker, Ammar Ben Hadj Amor, Salvador Capella-Gutierrez, Paulos Charonyktakis, ... Katherine Wolstencroft
F1000Research (2021-09-07) <https://doi.org/gqprp7>
DOI: [10.12688/f1000research.54159.1](https://doi.org/10.12688/f1000research.54159.1) · PMID: [34804501](https://pubmed.ncbi.nlm.nih.gov/34804501/) · PMCID: [PMC8573700](https://pubmed.ncbi.nlm.nih.gov/PMC8573700/)
72. **HTTP Semantics**
R Fielding, M Nottingham, J Reschke (editors)
RFC Editor (2022-06) <https://doi.org/gqprqb>
DOI: [10.17487/rfc9110](https://doi.org/10.17487/rfc9110)
73. **Linked data platform 1.0**
Linked Data Platform Working Group
W3C (2015-02-26) <https://www.w3.org/TR/2015/REC-ldp-20150226/>
74. **FAIR Digital Object Framework Documentation** <https://fairdigitalobjectframework.org/>
75. **Handle System Overview**
S Sun, L Lannom, B Boesch
RFC Editor (2003-11) <https://doi.org/ggn83z>
DOI: [10.17487/rfc3650](https://doi.org/10.17487/rfc3650)
76. <https://www.rd-alliance.org/sites/default/files/Cordra.2022.pdf>
77. **Global Internet Phenomena Report 2022**
Sandvine
<https://www.sandvine.com/global-internet-phenomena-report-2022>
78. **Hypertext Transfer Protocol Version 2 (HTTP/2)**
M Belshe, R Peon
RFC Editor (2015-05) <https://doi.org/gp32q9>
DOI: [10.17487/rfc7540](https://doi.org/10.17487/rfc7540)
79. <https://blog.cloudflare.com/http-3-vs-http-2/>
80. **draft-ietf-quic-http-34** <https://datatracker.ietf.org/doc/html/draft-ietf-quic-http-34>
81. **QUIC: A UDP-Based Multiplexed and Secure Transport**
J Iyengar, M Thomson (editors)
RFC Editor (2021-05) <https://doi.org/gkctr>
DOI: [10.17487/rfc9000](https://doi.org/10.17487/rfc9000)
82. **URI Template**
J Gregorio, R Fielding, M Hadley, M Nottingham, D Orchard
RFC Editor (2012-03) <https://doi.org/gp33dw>
DOI: [10.17487/rfc6570](https://doi.org/10.17487/rfc6570)
83. **Content negotiation - HTTP | MDN** https://developer.mozilla.org/en-US/docs/Web/HTTP/Content_negotiation

84. **Hypertext Style: Cool URIs don't change.** <https://www.w3.org/Provider/Style/URI>
85. **HTTP Framework for Time-Based Access to Resource States -- Memento**
H Van de Sompel, M Nelson, R Sanderson
RFC Editor (2013-12) <https://doi.org/ggqvps>
DOI: [10.17487/rfc7089](https://doi.org/10.17487/rfc7089)
86. **Versioning Extensions to WebDAV (Web Distributed Authoring and Versioning)**
G Clemm, J Amsden, T Ellison, C Kaler, J Whitehead
RFC Editor (2002-03) <https://doi.org/gp37bd>
DOI: [10.17487/rfc3253](https://doi.org/10.17487/rfc3253)
87. **HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV)**
L Dusseault (editor)
RFC Editor (2007-06) <https://doi.org/gp37bf>
DOI: [10.17487/rfc4918](https://doi.org/10.17487/rfc4918)
88. **Upgrading to TLS Within HTTP/1.1**
R Khare, S Lawrence
RFC Editor (2000-05) <https://doi.org/gp33dv>
DOI: [10.17487/rfc2817](https://doi.org/10.17487/rfc2817)
89. **Hypertext Style: Cool URIs don't change.** <https://www.w3.org/Provider/Style/URI.html>
90. **Media Types with Multiple Suffixes**
Manu Sporny, Amy Guy
IETF Datatracker <https://datatracker.ietf.org/doc/draft-ietf-mediaman-suffixes/00/>
91. **The 'profile' Link Relation Type**
E Wilde
RFC Editor (2013-03) <https://doi.org/gp32q7>
DOI: [10.17487/rfc6906](https://doi.org/10.17487/rfc6906)
92. **W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures** <http://www.w3.org/TR/xmlschema11-1/>
93. **JSON-LD 1.1** <http://www.w3.org/TR/json-ld/>
94. **WebSockets Standard** <https://websockets.spec.whatwg.org/>
95. **Linked Data Notifications** <https://www.w3.org/TR/ldn/>
96. **The Atom Publishing Protocol**
J Gregorio, B de hOra (editors)
RFC Editor (2007-10) <https://doi.org/gp4p2c>
DOI: [10.17487/rfc5023](https://doi.org/10.17487/rfc5023)
97. **SWORD 3.0 Specification** <https://swordapp.github.io/swordv3/swordv3.html>
98. **Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content**
R Fielding, J Reschke (editors)
RFC Editor (2014-06) <https://doi.org/gh4jxc>
DOI: [10.17487/rfc7231](https://doi.org/10.17487/rfc7231)
99. **Hydra W3C Community Group** <https://www.hydra-cg.com/>
100. **draft-kelly-json-hal-08** <https://datatracker.ietf.org/doc/html/draft-kelly-json-hal-08>

101. **Web Linking**
M Nottingham
RFC Editor (2017-10) <https://doi.org/gf8jcd>
DOI: [10.17487/rfc8288](https://doi.org/10.17487/rfc8288)
102. **Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing**
R Fielding, J Reschke (editors)
RFC Editor (2014-06) <https://doi.org/gp32q8>
DOI: [10.17487/rfc7230](https://doi.org/10.17487/rfc7230)
103. **HTTP Live Streaming**
W May
RFC Editor (2017-08) <https://doi.org/gp32rc>
DOI: [10.17487/rfc8216](https://doi.org/10.17487/rfc8216)
104. **ISO/IEC 23009-1:2019**
14:00-17:00
ISO <https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/07/93/79329.html>
105. **FAIR data maturity model: Specification and guidelines**
Research Data Alliance FAIR Data Maturity Model Working Group
Research Data Alliance (2020) <https://zenodo.org/record/3909563#.YGRNnq8za70>
DOI: [10.15497/rda00050](https://doi.org/10.15497/rda00050)
106. **NCBO BioPortal** <https://bioportal.bioontology.org/ontologies>
107. **Handle System Protocol (ver 2.1) Specification**
S Sun, S Reilly, L Lannom, J Petrone
RFC Editor (2003-11) <https://doi.org/ggn83x>
DOI: [10.17487/rfc3652](https://doi.org/10.17487/rfc3652)
108. **Handle.Net Registry** https://www.handle.net/download_hnr.html
109. **Licenses & Standards | Open Source Initiative** <https://opensource.org/licenses>
110. **Data - W3C** <https://www.w3.org/standards/semanticweb/data>
111. **Linking Across Provenance Bundles** <https://www.w3.org/TR/2013/NOTE-prov-links-20130430/>
112. **Introducing 'Role'**
Unknown
<http://blog.schema.org/2014/06/introducing-role.html>
113. **PROV-O: The PROV Ontology** <https://www.w3.org/TR/prov-o/#specializationOf>
114. **ORE Specification - Abstract Data Model** <http://www.openarchives.org/ore/1.0/datamodel#Proxies>
115. **Schema.org - Schema.org** <https://schema.org/>
116. **DCMI Metadata Terms** <https://www.dublincore.org/specifications/dublin-core/dcmi-terms/>
117. **Data Catalog Vocabulary (DCAT) - Version 2** <https://www.w3.org/TR/vocab-dcat-2/>
118. **FOAF spec** <http://xmlns.com/foaf/spec/>
119. **Bioschemas - Bioschemas** <http://bioschemas.org/>
120. **Semantic Web - XML2000 - slide "Architecture"** <https://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html>

121. **A Functional Semantic Web Architecture**
Aurona Gerber, Alta van der Merwe, Andries Barnard
Lecture Notes in Computer Science (2008) <https://doi.org/cn3mbp>
DOI: [10.1007/978-3-540-68234-9_22](https://doi.org/10.1007/978-3-540-68234-9_22)
122. **Semantic Web Architecture and its Impact on E-learning Systems Development**
Samir Abou El-Seoud, Hosam El-Sofany, Omar Karam
International Journal of Emerging Technologies in Learning (ijET) (2015-11-13) <https://doi.org/gqdxh>
DOI: [10.3991/ijet.v10i5.4754](https://doi.org/10.3991/ijet.v10i5.4754)
123. **Semantic Web Layer Cake Tweak, Explained**
Kingsley Uyi Idehen
OpenLink Software Blog (2017-07-24) <https://medium.com/openlink-software-blog/semantic-web-layer-cake-tweak-explained-6ba5c6ac3fab>

1. For a brief introduction to DOIP 2.0 [12], see [25].↵
2. 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.↵
3. URIs 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* [31].↵
4. *Datasets* that distribute RDF graphs should not be confused with [RDF Datasets](#) used for partitioning *named graphs*.↵
5. Presumably this large uptake of JSON-LD is mainly for the purpose of Search Engine Optimization (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.↵
6. [3] renames `_FDOF*` to `_FDOR*`, `FDOF3/FDOF4` are swapped to `FDOR4/FDOR3`. ↵
7. The `http` protocol (port 80) can in theory also upgrade [88] to TLS encryption, as commonly used by [Internet Printing Protocol](#) for `ipp` URIs, but on the Web, best practice is explicit `https` (port 443) URLs to ensure following links stay secure.↵
8. HATEOAS: Hypermedia as the Engine of Application State [64], an important element of the REST architectural style.↵
9. 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.↵
10. The `Handle.net` system was previously covered by software patent [US6135646A](#) which [expired](#) in 2013.↵
11. Note that while schema.org's JSON-LD examples frequently use `"@context": "https://schema.org"` this is not a versioned PID (not even a complete URI). While releases of this particular vocabulary strive to not break backward compatibility, we suggest using for instance <https://schema.org/version/14.0/schemaorg-current-https.jsonld> or an indirect `w3id`↵
12. The schema.org JSON-LD context used `http` PIDs since 2011, with a gradual transition to `https`, see [schemaorg/schemaorg#2853](https://schema.org/schemaorg#2853) and [schemaorg/schemaorg#2886](https://schema.org/schemaorg#2886)↵

