Evaluating FAIR Digital Object as a distributed object system

This manuscript (permalink) was automatically generated from stain/2022-fdo-paper@f3e33a7 on November 30, 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. Worthwile 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?

1 Introduction

The FAIR principles [1] 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 widely recognised by the research community and policy makers as a goal to strive for. In particular, the European Open Science Cloud (EOSC) has promoted FAIR data. The EOSC Interoperability Framework [2] 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 [3] (FDO) concept 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 [4] has been seen as an established set of principles based on Semantic Web technologies that can achieve the vision of the FAIR principles [5] [6]. Yet regular researchers and developers of emerging platforms for computation and data management are reluctant to adapt such a FAIR Linked Data approach fully, opting instead for custom in-house models and JSON-derived formats from RESTful Web services [7] [8]. While such focus on simplicity gives rapid development and highly specialised services, it raises wider concerns on interoperability [?].

One challenge that may, perhaps counter-intuitively, steer developers towards a not-invented-here mentality [9] [10] when exposing their data on the Web is the heterogeneity and apparent complexity of Semantic Web approaches themselves [11].

These approaches, 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 examine the relationships between FAIR and FAIR Digital Objects, contrasted with Linked Data and the Web in general.

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.

1.1 FAIR Digital Object

The concept of **FAIR Digital Objects** [3] has been introduced as way to expose research data as active objects that conform to the FAIR principles [1]. This builds on the *Digital Object* (DO) concept [30], first introduced in 1995 [31] 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 [32] which introduced an abstract *Digital Entity Interface Protocol* for managing such objects programmatically, first realised by the Digital Object Interface Protocol (DOIP) v1 [33].

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 [34].

Recently, FDOs have been recognized by the European Open Science Cloud (EOSC) as a suggested part of its Interoperability Framework [2], 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 [15]. 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 1.1.1).

1.1.1 Next steps for FDO

The FAIR Digital Object Forum [12] working groups are preparing more detailed requirement documents setting out the path for realising FDOs, named *FDO Recommendations*. As of 2022-11-30, these documents are in draft stages in Google Docs for internal review and not yet formally been made public. As these drafts clarify the future aims and focus of FAIR Digital Objects, we provide their brief summaries below:

The **FDO Forum Document Standards** [13] 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** [14] is an update of [15] 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 [tbl:fdo-checks] are largely equivalent, but here clarified with references to other FDO documents.

The **Machine actionability** [16] 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 [17] 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 [18]

PID Profiles & Attributes [19] 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*

1.1.2 FDO approaches

FDO is an evolving concept. A set of FDO Demonstrators [24] highlight how current adapters are approaching implementations of FDO from different angles:

- Building on the Digital Object concept, using the simplified DOIP v2 specification [23], 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. FDO types are registered in the local Cordra instance, where they are specified using JSON Schema [36]) 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 Digital Objects and Linked Data, which we'll cover in the section 1.2.

Types working group [20]. 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 [21] 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 [22] is an endorsement of the DOIP v2.0 [23] specification as a potential FDO implementation, as it has been applied by several institutions [24]. 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 [25] 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 [26] 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 [27] 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 [28] 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 [29].

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.

1.2 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*.

1.2.1 A brief history of the Semantic Web

The **Semantic Web** was developed as a vision by Tim Berners-Lee [37], at a time the Web had been widely established for information exchange, as a global set of hypermedia documents that eare 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) [38] that used *triples* of subject-predicate-object statements, with its initial serialisation format [39] 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 [40], the main innovation of RDF was the use of global identifiers in the form of URIs[2] 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 documents[3], 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 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) 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 [46] 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 [47] (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, rdflib.js), and numerous graph visualisation (many of which struggle with usability for more than 20 nodes).

The creation of RDF-based knowledge graphs grew particularly in fields like bioinformatics, e.g. for describing genomes and proteins [48,49]. 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 [50] (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`) [51] .

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 [52]), 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 utillised 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 [53]. Just like *link rot* affect regular Web pages

and their citations from scholarly communication [54], for a majority of described RDF resources in the Linked Open Data (LOD) Cloud's gathering of more than thousand datasets, unfortunately they do not actually link to (still) downloadable (*dereferenceable*) Linked Data [55]. 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 [56] , but also technological choices on RDF serialisation (at least 7 formats), type system (RDFS [57], OWL [58], OBO [59], SKOS [50]), hash vs slash, HTTP status codes and PID redirection strategies [45].

1.2.2 Linked Data: Rebuilding the Web of Data

The **Linked Data** concept [4] was kickstarted as a set of best practices [60] to bring the Web aspect back into focus. Crucially 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* (Open Graph [61], RDFa [62], Microdata [63]) we find JSON-LD [64] as a Web-focused RDF serialisation that aims for improved programmatic generation and consumption, including from Web applications. JSON-LD is as of 2022-05-13 used[5] by 42.7% of the top 10 million websites [65].

Recently there has been a renewed emphasis to improve the *Developer Experience* [66] for consumption of Linked Data, for instance RDF Shapes (expressed in SHACL [67] or ShEx [68]) [69] can be used to validate RDF Data [70] 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 [71], 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 [72]. 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 [73], the use of Linked Data as part of REST Web APIs [74] , and as shown by the big uptake by publishers to annotate the Web using schema.org [75], 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.

2 Comparing FDO and existing approaches

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

1. An Interoperability Framework and Distributed Platform for Fast Data Applications [76], which proposes quality measurements for comparing how frameworks support interoperability, particularly from a service architectural view
2. The FAIR Digital Object guidelines [15], validated against its implementations for completeness.
3. A Comparison Framework for Middleware Infrastructures [77], which suggest dimensions like openness, performance and transparency, mainly focused on remote computational methods
4. Cross-checks against RDA's FAIR Data Maturity Model [78] to find how the FAIR principles are achieved in FDO, in particular considering access, sharing and openness
5. EOSC Interoperability Framework [2] which gives recommendations for technical, semantic, organisational and legal interoperability, particularly from a metadata perspective

The reason for this wide-ranged 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 1.1.1).

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 realisation using the DOIPv2 protocol [23] against Linked Data in general.

2.1 Considering FDO/Web as interoperability framework for Fast Data

The Interoperability Framework for Fast Data Applications [76] 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 [79] 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 [76].

Quality	**FDO w/ DOIP**	**Web w/ Linked Data**
Symbiotic: *to what extent multiple applications can agree to interact/align/collaborate/cooperate*	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 [80] end up being mandated centrally by dominant (often commercial) providers [81], 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: *using interaction contracts so processes can be choreographed in workflows*	FDO types and operations enable detailed choreography (see CWFP). `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 formalized (similar operations can be grouped with `0.DOIP/OperationReference`).	"Follow your nose" crawler navigation, which may lead to frequent dead ends. Operational composition, typically within a single API provider, documented by OpenAPI 3 [82], schema.org Actions [[83]), WSDL/SOAP [84], but frequently just as human-readable developer documentation/examples.
Semantic: *ensuring consistent understanding of messages, interoperability of rules, knowledge and ontologies*	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) [24]	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: *serializing messages for digital exchange, structure representation*	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.

Quality	FDO w/ DOIP	Web w/ Linked Data
Connective: *transferring messages to another application, e.g. wrapping in other protocols*	DOIP [23] 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: *how applications are deployed and affected by its environment, portability*	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 hetereogeneous Web Services [85]. However, this is mostly done via developer or human interaction [86]. Similiarly, 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 Web standards bodies (e.g. IETF and W3C) follow the OpenStand principles [87] 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 FDO conference [88]. 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 [89].

It's 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 [90] using HTTP redirects with the common `https://doi.org/` prefix, hiding their Handle nature as an implementation detail [91].

2.1.1 Mapping of Metamodel concepts

The Interoperability Framework for Fast Data also provide a brief *metamodel* which we use in table [2] to map and examplify corresponding concepts in FDO's DOIP realization and the Web using HTTP semantics [92].

Table 2: Mapping the Metamodel concepts from the Interoperability Framework for Fast Data [76] 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`

Metamodel concept	FDO/DOIP concept	Web/HTTP concept
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 2.3), as well that HTTP has poor direct support for processes, as the Web is primarily stateless by design.

2.2 Assessing FDO implementations

The FAIR Digital Object guidelines [15] sets out recommendations for FDO implementations. In Table 3 we evaluate the two current implementations, using DOIPv2 [23] and using Linked Data Platform [93], as proposed by [94].

Table 3: Checking FDO guidelines [WD-RequirementSpec-1.0-20220317?] against its current implementations as DOIP [23] and Linked Data Platform (LDP) [94], with suggestions for required additions.

FDO Guideline	DOIP 2.0	FDO suggestions	Linked Data Platform	LDP suggestion
G1: *invest for many decades*	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. `http://` URIs replaced by `https://`.	Keep flexibility of RDF serialization formats which may change more frequently
G2: *trustworthiness*	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 [?], however previous bad experiences with Semantic Web may deter adapters	Ensure simplicity for end developers, rather than semantic overengineering. Example-driven documentation.
G3: *follows FAIR principles*	See table ??	Ensure all FAIR principles are covered, build complete examples.	Touched briefly, see table ??	Add explicit expression to show each FAIR pcinciple fulfilled.

FDO Guideline	DOIP 2.0	FDO suggestions	Linked Data Platform	LDP suggestion
G4: *machine actionability*	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.
G5: *abstraction principle*	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.
G6: *stable binding between entities*	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.	
G7: *encapsulation*	Operations discovered at runtime (`0.DOIP/Op.ListOperations`).	Allow method discovery by type FDOs in advance (see PR-TypingFDOs-2.0-20220608).	HTTP methods discovered at runtime (`OPTIONS`), indempotent methods attempted directly. Unsupported methods reported using LDP constraints to human-readable text.	Declare supported methods in advance, e.g. OpenAPI [82]
G8: *technology independence*	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: *standard compliance*	Handle [95], DOIP [23]. 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

FDO Guideline	DOIP 2.0	FDO suggestions	Linked Data Platform	LDP suggestion
FDOF1: *PID as basis*	Extensive use of Handle system.	Clarify how local testing handles can be used during development, how “temporary” FDOs should evolve [**PID?** policy]. Register `0.DOIP/*` and `0.FDO/*` as PIDs.		
FDOF2: *PID record w/ type*	Unspecified how to resolve from Handle to DOIP Service (!), in practice `10320/loc`, `0.TYPE/DOIPService`, `URL`, `URL_REPLICA`	Document requirements for PID Record ()		
FDOF3: *PID resolvable to bytestream & metadata*	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.	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: *Additional attributes*	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.
FDOF5: *Interface: operation by PID*	Extended operations use PID, but “pid-like” DOIP operations/types are not registered as handles.	Register `0.DOIP/*` and `0.FDO/*` 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.

FDO Guideline	DOIP 2.0	FDO suggestions	Linked Data Platform	LDP suggestion
FDOF6: *CRUD operations + extensions*	`0.DOIP/Op.Create`, `Op.Retrieve`, `Op.Update`, `Op.Delete` but also `0.DOIP/Op.Search`.	Document	`PUT`, `GET`, `POST`, `DELETE`, `PATCH`, `HEAD` – extension operations (e.g. WebDAV `COPY`) not used, resource patterns [96] are used instead.	Document how operation resources can be discovered from an LPD container. Document search API.
FDOF7: *FDOF Types related to operations*	Not yet formalized, by DOIP discoverable on a given FDO rather than type. PR-TypingFDOs leaves this open.	Add explicit relation between type and operations	`OPTIONS` per LDP Resource, but not by type. Common types (`ldp:Resource`, `ldp:Container`) indicate LDP support, but are not requried.	Always make LDP types explicit in FDO profile.
FDOF8: *Metadata as FDO, semantic assertions*	DOIP includes all metadata in PID Record. Separate Metadata FDO need custom property.	Specify a `0.FDO/metadata` or similar to point to Metadata FDOs.	Assertions are always with semantics, using RDF vocabularies. Unspecified how to find additional metadata resources, `rdfs:seeAlso` is common.	Use FAIR Signposting `describedby` link relation to additional metadata PIDs
FDOF9: *Different metadata levels*	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/attributes. Unclear how levels are separated within single FDO representation (need FDOF8?).	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.	Undefined how to handle multiple metadata granularities or domains, alternative LDP containers can present different views on same stored objects.	Define how to navigate to alternate views and their semantic implications, e.g. `owl:sameAs`
FDOF10: *Metadata schemas by community*	Metadata schemas are in practice managed on single CORDA server as local types, using JSON Schema.	Require types to be FDOs with registered PIDs, implement shared types.	Plethora of existing RDF vocabularies/ontologies managed by larger communities, e.g. OBO Foundry [doi:https://doi.org/10.1038/nbt1346]	Rather document better how individual ad-hoc schemas can be started for prototypes.
FDOF11: *FDO collections w/ semantic relations*	Collection type undefined by DOIP. Informal use of `HAS_PARTS` Handle attribute (e.g. [97]).		LDP Containers required by specification, also user-created (eg. `BasicContainer`).	Clarify relation to other collections like DCAT 3 [98], Schema.org Dataset, OAI-ORE [99]

FDO Guideline	DOIP 2.0	FDO suggestions	Linked Data Platform	LDP suggestion
FDOF12: *Deleted FDO preserve PID w/ tombstone*	Tombstone for deleted resource undefined by DOIP. `0.DOIP/Status.104` status code does not distinguish “Not Found” or “Gone”	Formalize tombstone requirements with new FDO type	`410 Gone` recommended, but `404 Not Found` common. No requirement for tombstone serialization	Formalize tombstone requirements and serialization

Note that the draft update to FDO specification [WD-RequirementSpec-1.0-20220317?] (see box 1.1.1) clarifies these definitions with equivalent identifiers [6] and relates them to further FDO requirements such as FDO Data Type Registries.

A key observation from this is that simply using DOIP does not achieve many of the FDO guidelines. Rather the guidelines set out how a protocol like DOIPs should be used to achieve FAIR Digital Object goals. The DOIP Endorsement [22] sets out that to comply, DOIP must be used according to the set of FDO requirement documents (see box 1.1.1), and notes *Achieving FDO compliance requires more than DOIP and full compliance is thus left to system designers*. Likewise, a Linked Data approach will need to follow the same requirements to comply as an FDO implementation.

From our evaluation we can observe: * G1 and G2 call for stability and trustworthiness. While the foundations of both DOIP 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 which way types and operations are discovered, with FDO seemingly more rigorous. In practice, however, we see that DOIP 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 affects the architectural view – FDOs are more of an . * FDO puts strong emphasis on the use of PIDs (FDOF1, FDOF2, FDOF3, FDOF5), but in current practice DOIP 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, perhaps `prov:specializationOf` [100] * FDO collections are not yet defined for DOIP, 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 2.4). * DOIP 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.

2.3 Comparing FDO and Web as middleware infrastructures

TODO: Introduce middleware infrastructures [77].

In this section we take into account 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 then argue that the Web, with its now ubiquitous use of REST API [80], can be compared as a similar global middleware. Note that while early moves for developing Semantic Web Services [101] 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 ★★★ *Linked Data* [102].

For this purpose, we here utillize the Comparison Framework for Middleware Infrastructures [77] that formalize multiple dimensions of openness, scalability, transparency, as well as characteristics known from Object-oriented programming such as modularity, encapsulation and inheritance.

Table 4: Comparing FAIR Digital Object (with the DOIP 2.0 protocol [23]) and Web technologies (using Linked Data) as middleware infrastructures [77]

Quality	**FDO w/ DOIP**	**Web w/ Linked Data**
Openness: *framework enable extension of applications*	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: *application should be effective at many different scales*	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: *efficient and predictable execution*	DOIP has been shown moderately scalable to 100 millions of objects, create operation at 900 requests/second [103]. 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.	HTTP traffic is about 10% of global Internet traffic, excluding video and social networks [104]. HTTP 1 connections are serial and reusable, and concurrent connections is common. HTTP/2 adds asynchronous responses and multiplexed streams [105] but still has TCP+TLS startup costs. For reduced latency [106], HTTP/3 [107] use QUIC [108]) rather than TCP, already adapted heavily (30% of EMEA traffic) of which Instagram & Facebook video is the majority of traffic.
Distribution transparency: *application perceived as a consistent whole rather than independent elements.*	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 [109], content-negotiation [110] or FAIR Signposting [18].
Access transparency: *local/remote elements accessed similarly*	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.	Global HTTP protocol frequently used locally and behind firewalls, but at risk of non-global URIs (e.g. `http://localhost/object/1`) and SSL issues (e.g. self-signed certificates, local CAs)
Location transparency: *elements accessed without knowledge of physical location*	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: *concurrent processing without interference*	No explicit concurrency measures. FDO kernel metadata can include checksum and date.	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 occassionally violated by Web applications. Cache control, `ETag` (~ checksum) and modification date in HTTP headers allows detection of concurrent changes on a single resource.
Failure transparency: *service provisioning resiliant to failures*	DOIP status codes, e.g. `0.DOIP/Status.104`, additional codes can be added as custom attributes	HTTP status codes e.g. `404 Not Found`, specific meaning of standard codes can be documented in Open API. Custom codes uncommon.

Quality	FDO w/ DOIP	Web w/ Linked Data
Migration transparency: *allow relocating elements without interferring application*	Update of PID record URLs, indirection through `0.TYPE/DOIPServiceInfo` (not always used consistently). No redirection from DOIP service.	HTTP `30x` status codes provide temporary or permanent redirections, commonly used for PURLs but also by endpoints.
Persistence transparency: *conceal deactivation/reactivation of elements from their users*	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 [111]. 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 [112] expose versioned snapshots. WebDAV `VERSION-CONTROL` method [113] (used by SVN).
Transaction transparency: *coordinate execution of atomic/isolated transactions*	No transaction capabilities declared by FDO or DOIP. Internal synchronization possible in backend for Extended operations.	Limited transaction capabilities (e.g. `If-Unmodified-Since`) on same resource. WebDAV locking mechanisms [114] with `LOCK` and `UNLOCK` methods.
Modularity: *application as collection of connected/distributed elements*	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 inheritently 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.
Encapsulation: *separate interface from implementation. Specify interface as contract, multiple implementations possible*	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)	HTTP/1.1 semantics can seemlessly upgrade to HTTP/2 and HTTP/3. `http` vs `https` URIs exposes encryption detail [7]. Implementation details may leak into URIs (e.g. `search.aspx`), countered by deliberate design of URI patterns [116] and PIDs via Persistent URLs (PURL).
Inheritance: *Deriving specialized 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 `$ref` use of JSON Schema references [36])	Syntactically Media Type with multiple suffixes [117] (mainly used with `+json`), declaration of subtypes as profiles (RFC6906) [118]. In metadata, semantic type systems (RDFS [57], OWL2 [58], SKOS [50]). OpenAPI 3 [82] inheritance and Polymorphism. XML `xsd:schemaLocation` or `xsd:type` [119], JSON `$schema` [36]), JSON-LD `@context` [120]. 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 [105], Web Sockets [121], Linked Data Notifications [122], AtomPub [123], SWORD [124], Micropub, 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"`)	CRUD predefined in HTTP methods [125], (extended by registration), URI Templates [109], OpenAPI operations [82], HATEOAS[8] incl. Hydra [126], schema.org Actions [[83]), JSON HAL [127] & Link headers (RFC8288) [128]
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 [129] chunked transfer, HLS (RFC8216) [130], MPEG-DASH (ISO/IEC 23009-1:2019) [131]

Observations:

- As for the aspect of *Performance*, it is interesting to note that while the first version of DOIP [33] 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 [23], 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].
- HTTP have strong support for scalability and caching, but this mostly assumes read-operations from static resources. FDO have no view on immutability or validity of retrieved objects, but this should be taken into consideration to support large-scale usage.
- HTTP optimizations 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 protocol
- Programmatically the Web is rather unstructured as middleware, as there are many implementation choices. Usually it is semantically undeclared what to expect for a given URI/service, and programmers follow documented examples for a particular service rather than automated programmatic exploration across providers. This mean we can rather 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.

2.4 Assessing FDO against FAIR

In addition to having "FAIR" in its name, the FAIR Digital Object guidelines [15] also include *G3: FDOs must offer compliance with the FAIR principles through measurable indicators of FAIRness.* [**PR-RequirementSpec-2.0?**]. Here we evaluate to which extent the FDO guidelines and its implementation with DOIP and Linked Data Platform [94] comply with the FAIR principles [1]. Here we've used the RDA's FAIR Data Maturity Model [132] as it has decomposed the FAIR principles to a structured list of FAIR indicators [78], importantly considering *Data* and *Metadata* separately. In our interpretation for Table 5 we have for simplicity chosen to interpret "data" in FDOs as the associated bytestream of arbitrary formats, with remainining JSON/RDF structures always considered as metadata.

Table 5: Assessing RDA's FAIR Data Maturity Model [78,132] (first 2 columns) against the FDO guidelines [15], FDO implemented with the protocol DOIPv2 [23], Linked Data Platform (LDP) [94] and examples from Linked Data practices in general. (— indicates *Unspecified*, may be possible with additional conventions)

FAIR ID	Indicator	FDO guidelines	FDO/DOIP	FDO/LDP	Linked Data examples
RDA-F1-01M	Metadata is identified by a persistent identifier	FDOF4	Optional *Metadata FDO* 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 [111], PURL services incl. `purl.org`, `w3id.org`
RDA-F1-02M	Metadata is identified by a globally unique identifier	FDOR4 FDOF8	Optional *Metadata FDO*, unspecified how to indicate	Content-negotiation to URL	Not required, content-negotiation can redirect to URL or `Content-Location`. FAIR Signposting.

FAIR ID	Indicator	FDO guidelines	FDO/DOIP	FDO/LDP	Linked Data examples
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	—	`id` and `type` 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. `www.example.com` vs `example.com`) or missing (use of `<>` as *this* 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. [133]). 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	Directly by DOIP, but not included in FDO metadata. `handle.net` HTTP resolution may redirect to landing page	Any property can point to URIs, but unclear if it is data	Common with clickable "follow your nose" URLs
RDA-A1-02M	Metadata can be accessed manually (i.e. with human intervention)	—	(Cordra HTML landing page from `handle.net` URIs)	Optional content-negotiation, e.g. by Apache Marmotta, OpenLink Virtuoso	HTTP content-negotiation to HTML is common
RDA-A1-02D	Data can be accessed manually (i.e. with human intervention)	—	(Cordra HTML landing page from `handle.net` URIs)	Optional content-negotiation	Direct download, HTML landing pages common for DOIs
RDA-A1-03M	Metadata identifier resolves to a metadata record	FDOF8+FDOF2	—	—	`Content-Location` or HTTP redirection may indicate metadata URI
RDA-A1-03D	Data identifier resolves to a digital object	FDOF2	Required, but frequently not directly resolvable	Recommended, but any URI acceptable	Resolvable HTTP/HTTPS URIs are most common, now infrequent URNs are not directly resolvable
RDA-A1-04M	Metadata is accessed through standardised protocol	G9 FDOF3	Retrievable from PID (FDOF3). Informal DOIP standard maintained by DONA Foundation	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)	Formal HTTP standards maintained by IETF, HTTP content negotiation, informal FAIR Signposting
RDA-A1-04D	Data is accessible through standardised protocol	G9	(see above)	HTTP [92]	HTTP/HTTPS, FTP (now less common), GridFTP [134] (for large data), ARK [135]
RDA-A1-05D	Data can be accessed automatically (i.e. by a computer program)	G4 FDOF3 FDOF6	Required, but few client libraries		Ubiquitous, hundreds of HTTP libraries

FAIR ID	Indicator	FDO guidelines	FDO/DOIP	FDO/LDP	Linked Data examples
RDA-A1.1-01M	Metadata is accessible through a free access protocol	G1 G8 G9	Partially realized: Handle system is open protocol [[136]][10]. One server implementation [137], free[^license]. 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	(see above)	URI, DNS, HTTP, TLS	URI, DNS, HTTP, TLS. Non-free DRM may be used (e.g. subscription video streaming)
RDA-A1.2-01D	Data is accessible through an access protocol that supports authentication and authorisation	(FDOR9)	TLS certificates, `authentication` field (details unspecified)	Implied	HTTP authentication, TLS certificates
RDA-A2-01M	Metadata is guaranteed to remain available after data is no longer available	FDOF12	—	Unspecified, however FDOF-IR links to separate metadata records	—
RDA-I1-01M	Metadata uses knowledge representation expressed in standardised format	FDOF8	Required, but not currently defined	—	Always implied by use of RDF syntaxes.
RDA-I1-01D	Data uses knowledge representation expressed in standardised format	—	—	—	Common (e.g. HDF5, JSON, XML), yet common scientific data formats frequently not standardized
RDA-I1-02M	Metadata uses machine-understandable knowledge representation	FDOF8	Required	Optional RDF metadata with any vocabulary	Always implied by use of RDF syntaxes.
RDA-I1-02D	Data uses machine-understandable knowledge representation	G4 G7 FDOR2	No requirements on binary data formats	Only indirectly, LDP Basic Container reference only information resources	Common, specially for scientific data formats
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	—	—	—	Uncommon, except for some XML and RDF-embedding formats, e.g. Extensible Metadata Platform (XMP) [138]

FAIR ID	Indicator	FDO guidelines	FDO/DOIP	FDO/LDP	Linked Data examples
RDA-I3-01M	Metadata includes references to other metadata	FDOR8	Implied (attributes to PIDs), currently unspecified if given attribute is value or reference	—	By definition (Linked Data reference existing URIs [139]), `rdfs:seeAlso`, FAIR signposting [18] `describedby`
RDA-I3-01D	Data includes references to other data	G6 FDOR3 FDOR11	—	—	URL hyperlinks common in several formats (HTML, PDF, JSON, XML).
RDA-I3-02M	Metadata includes references to other data	G6 FDOR3 FDOR8	Implied from custom FDO type's attribute	LDP Direct Container members can be any resources	URI objects are frequently data references, may be indirect via PID
RDA-I3-02D	Data includes qualified references to other data	FDOR3 FDOR11	Only indirectly through FDO metadata	Indirectly through LDP membership	Uncommon: Link relations, FAIR Signposting
RDA-I3-03M	Metadata includes qualified references to other metadata	(FDOR3)	Qualification by attribute keys defined per FDO Type	LDP Direct Container	Qualifications by property, PROV bundles [140], schema.org/Role
RDA-I3-04M	Metadata include qualified references to other data	(FDOR3)	Qualification by attribute keys defined per FDO type	LDP Indirect Container	Qualifications by property, n-ary indirection (schema.org Role [141], `prov:specializationOf` [142], OAI-ORE Proxy [143])
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 [144], Dublin Core Terms [145], DCAT [146], FOAF [147]
RDA-R1.1-01M	Metadata includes information about the licence under which the data can be reused	—	Unspecified (should be in PID Kernel metadata?)	—	Dublin Core Terms `dct:license` frequently recommended, frequently not required, e.g. by DCAT 2 [146]
RDA-R1.1-02M	Metadata refers to a standard reuse licence	—	—	—	SPDX and Creative Commons URIs common, identifiers often inconsistent
RDA-R1.1-03M	Metadata refers to a machine-understandable reuse licence	—	—	—	SPDX documents uncommon
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		

FAIR ID	Indicator	FDO guidelines	FDO/DOIP	FDO/LDP	Linked Data examples
RDA-R1.2-02M	Metadata includes provenance information according to a cross-community language	FDOR9 FDOR8	—	—	W3C PROV-O [100], PAV [148], Dublin Core Terms [149]
RDA-R1.3-01M	Metadata complies with a community standard	FDOR10 FRORR8	(Emerging, e.g. DiSSCo Digital Specimen [@{doi:10.1162/dint_a_00134		
}])	—	Common, e.g. DCAT 2 [150], BioSchemas [151]			
RDA-R1.3-01D	Data complies with a community standard	(FDOR3)	—	—	Common, HTTP use registered IANA media types, additional scientific file formats frequently not standardized or identified
RDA-R1.3-02M	Metadata is expressed in compliance with a machine-understandable community standard	FDOF4 FDOF10	Recommended	—	Common practice for ontologies, specially in bioinformatics, e.g. BioPortal [133], Darwin Core [152]
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	—	Occassionally, (e.g. GFF3, FITS, ESRI)

From this evaluation we observe:

- Linked Data in general is strong on metadata indicators, but LDP approach is weak as it has no metadata guidance
- FDO/DOIP are stronger on identifier indicators
- 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 [23] 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 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.
- 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 [18] can improve this 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 resolution of Handle PIDs to the corresponding DOIP server is currently undefined by FDO and DOIP specifications. `0.TYPE/DOIPServiceInfo` lookup is only possible once DOIP server is known.

3 EOSC Interoperability Framework

TODO: Introduce EOSC IF

The EOSC Interoperability Framework [2] in section 3.6 recommends:

Layer	Recommendation	FDO	Linked Data
Technical	Open Specification	FDO specifications are semi-open, process gradually more transparent	Open and transparent standard processes through W3C & IETF
Technical	Common security & privacy framework	Unspecified	TLS for encryption, multiple approaches for single-sign-on (e.g. ORCID, Life Science Login). Privacy largely unspecified.
Technical	Easy SLAs for service providers	Unspecified	None
Technical	Access data in different formats	None formalized, custom operations or relations	Content-negotiation, `rel=alternate` relations
Technical	Coarse-grained/fine-grained search tools	Freetext `0.DOIP/Op.Search` on local DOIP, no federation	Coarse-grained e.g. Google Dataset Search, fine-grained (e.g. federated SPARQL) require detailed vocabulary/metadata insight
Technical	Clear PID policy	Strong FDO requirements, tends towards Handle system.	Not required, different communities set policies
Semantic	Clear definitions for concepts/metadata/schemas	Required by FDO requirements, but not yet formalized	Ontologies, SKOS, OWL
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 (currently unspecified)	DCAT, ++
Semantic	Crosswalks from minimal metadata model	FDO Typing recommends referencing existing type definitions, but not as separate crosswalks	Multiple crosswalks for common metadata models, but frequently not in semantic format
Semantic	Extensibility options for diciplinary 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

Layer	Recommendation	FDO	Linked Data
Organisational	Interoperability-focused rules of participation recommendations	Recommended	Implied only by some communities, tendency to specialize
Organisational	Usage recommendations of standardised data formats	None	None – but common for metadata (e.g. JSON-LD)
Organisational	Usage recommendations of vocabularies	Recommended by community	Common (see RDMKit)
Organisational	Usage recommendations of metadata	Recommended by community	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, Named graphs for splitting metadata
Legal	Mark expired/inexistent copyright	Undefined	Unclear, semantics assume copyright valid
Legal	Mark orphaned data	Tombstone for deleted data, but no owner of DOIP server means FDO disappears	Frequently data and endpoint has no known maintainer, archiving in common repositories becoming common
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	Undefined	Undefined
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

Observations: * The recommendations from EOSC IF are at a higher level that mainly affect governance and practices by communities * Technical aspects highlighted by EOSC IF * Search/indexing is important FAIR aspect for Findability, but is poorly supported by current FDO and Linked Data. There is a strong role for organizations like EOSC to provide broader registries than more specialized metadata federations like OpenAIRE. * FDO principles have strong recommendations for community development of organizational aspects. * Both FDO and LD are weak on legal aspects like licensing, privacy and usage policies – these are essential for cross-institutional and cross-repository access of FAIR objects

4 Discussion

TODO

4.1 (What does it mean for Linked Data?)

The FAIR Digital Object approach raises many important points for Linked Data practictioners. At first glance, the explicit requirements of FDOs may seem to be easy to furfill by different parts of the Semantic Web Cake [153]. 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 utillize 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.

4.2 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

5 References

1. **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 · PMID: 26978244 · PMCID: PMC4792175

2. **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

3. **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} · ISBN: 978-3-030-23583-3

4. **Linked data - the story so far**
Christian Bizer, Tom Heath, Tim Berners-Lee
International journal on Semantic Web and information systems (2009-07) http://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/jswis.2009081901
DOI: 10.4018/jswis.2009081901

5. **Assessing FAIR data principles against the 5-star open data principles**
Ali Hasnain, Dietrich Rebholz-Schuhmann
The semantic web: ESWC 2018 satellite events: ESWC 2018 satellite events, heraklion, crete, greece, june 3-7, 2018, revised selected papers (2018) http://link.springer.com/10.1007/978-3-319-98192-5_60
DOI: 10.1007/978-3-319-98192-5_60 · ISBN: 978-3-319-98191-8

6. **FAIR data points supporting big data interoperability**
Luiz Olavo Bonino Da Silva Santos, Mark D Wilkinson, Arnold Kuzniar, Rajaram Kaliyaperumal, Mark Thompson, Michel Dumontier, Kees Burger
Enterprise interoperability in the digitized and networked factory of the future (2016)
https://www.researchgate.net/publication/{309468587_FAIR_Data_Points_Supporting_Big_Data_Interoperability}
ISBN: 9781847040442

7. **Conclusion and future challenges**
Albert Meroño-Peñuela, Pasquale Lisena, Carlos Martínez-Ortiz
Web data apis for knowledge graphs: Easing access to semantic data for application developers (2021)
DOI: 10.1007/978-3-031-01917-3_7 · ISBN: 978-3-031-00789-7

8. **An analysis of public REST web service apis**
Andy Neumann, Nuno Laranjeiro, Jorge Bernardino
IEEE Transactions on Services Computing (2021-07-01)
DOI: 10.1109/tsc.2018.2847344

9. **Do Developers Make Unbiased Decisions? - The Effect of Mindfulness and Not-Invented-Here Bias on the Adoption of Software Components**
Anisa Stefi
University of Münster, Münster, Germany (2015) https://doi.org/grcpwg

DOI: 10.18151/7217489

10. **To develop or to reuse? Two perspectives on external reuse in software projects**
Anisa Stefi, Thomas Hess
Software business (2015)
DOI: 10.1007/978-3-319-19593-3_18 · ISBN: 978-3-319-19592-6

11. **Web data apis over SPARQL**
Albert Meroño-Peñuela, Pasquale Lisena, Carlos Martínez-Ortiz
Web data apis for knowledge graphs: Easing access to semantic data for application developers (2021)
DOI: 10.1007/978-3-031-01917-3_3 · ISBN: 978-3-031-00789-7

12. **FAIR Digital Objects Forum |** https://fairdo.org/

13. **FDO forum document standards**
C Weiland, U Schwardmann, P Wittenburg, C Kirkpatrick, R Hanisch, Z Trautt
FDO Forum (2022-01-29) https://docs.google.com/document/d/{1lPNBBROjEoZ6fTfrtdqcMa3Q2G27PoC_}/edit

14. **FDO forum FDO requirement specifications**
G Strawn, P Wittenburg, St Soiland-Reyes, K Peters, L Lannom, U Schwardmann, Ch Blanch
FDO Forum (2022-08-22) https://docs.google.com/document/d/{1aGA}-{TBr4XpORhMPtnf_}--{Nb4FYJccgeSvGmGh68jNws}/edit

15. **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}

16. **FDO machine actionability**
Claus Weiland, Sharif Islam, Daan Broder, Ivonne Anders, Peter Wittenburg
FDO Forum (2022-06-11) https://docs.google.com/document/d/{1GHFPAUGpNvYaxctkx}-{CpvY1vKf_aGZpSlWGOWvyXSiQ}/edit

17. **FDO configuration types**
Larry Lannom, Karsten Peters-von Gehlen, Ivonne Anders, Andreas Pfeil, Alexander Schlemmer, Zach Trautt, Peter Wittenburg
FDO Forum (2022-06-08)
https://docs.google.com/document/d/{1ivvg3C_QWSO9PIQwkKT89xG4fBhNAs7_6b0Dz11EwDg}/edit

18. **FAIR Signposting Profile - Signposting the Scholarly Web** https://signposting.org/FAIR/

19. **FDO PID profiles & attributes**
Ivonne Anders, Maggie Hellström, Sharif Islam, Thomas Jejkal, Larry Lannom, Ulrich Schwardmann, Peter Wittenburg
FDO Forum (2022-06-07)
https://docs.google.com/document/d/{1c2mZziq5pIPmLxMHLcYqlWrjYsc2ezGMXvp0E46iljo}/edit

20. **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

21. **FDO – granularity, versioning, mutability**
FDO-TSIG Working Group

FDO Forum (2022-08-26)
https://docs.google.com/document/d/{1VJTYHxLXIbjHSoFnMEkKaTbpTz9CagoIJyoSafkVh__I}/edit

22. **DOIP endorsement request**
FDO-TSIG Working Group
FDO Forum (2022-06-08) https://docs.google.com/document/d/{1me0L8C5yDe39cYP1Sxud4Y10hxhphOimLB}-K-{KgHQUk}/edit

23. **Digital object interface protocol specification, version 2.0**
DONA Foundation
DONA foundation (2018-11-12) https://hdl.handle.net/0.DOIP/DOIPV2.0

24. **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)
DOI: 10.5281/zenodo.5872645

25. **FDO – upload of FDO**
Christophe Blanchi, Daan Broeder, Thomas Jejkal, Islam Sharif, Alexander Schlemmer, Dieter van Uytvanck, Peter Wittenburg
FDO Forum (2022-06-08) https://docs.google.com/document/d/{1fDR5VHbVIa2AbLsBR58idrfn__Ib3x6Fk}-{__LJ4c__Ftt4}/edit

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

27. **Typing FAIR digital objects**
Larry Lannom, U Schwardmann, C Blanchi, P Wittenburg
FDO Forum (2022-06-08) https://docs.google.com/document/d/{1X0hcOVIqP7iYIJf9u}-{7x3RwcXK8ecsauy0FZg__6}-Bg0/edit

28. **Implementation of attributes, types, profiles and registries**
C Blanchi, M Hellström, L Lannom, U Schwardmann, P Wittenburg
FDO Forum (2022-11-27) https://docs.google.com/document/d/{1RrOiwMhkl}-{hRzWmlluA2iXCzHK}-{bj7__80LlMXgWx4w4}/edit\#

29. **Two Examples on How FDO Types can Support Machine and Human Readability**
Ulrich Schwardmann, Tibor Kálmán
Research Ideas and Outcomes (2022-10-12) https://doi.org/grcq23
DOI: 10.3897/rio.8.e96014

30. **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

31. **A framework for distributed digital object services**
Robert Kahn, Robert Wilensky
CNRI (1995-05-13) http://www.cnri.reston.va.us/k-w.html

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

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

34. **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

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

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

37. **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

38. **RDF 1.1 Primer** http://www.w3.org/TR/rdf11-primer/

39. **Resource Description Framework (RDF) Model and Syntax Specification** https://www.w3.org/TR/1999/REC-rdf-syntax-19990222/

40. **Process modelling for information system description**
Stefan K Stanczyk
The Open University (1987) https://doi.org/gp6znp
DOI: 10.21954/ou.ro.0000f821

41. **Uniform Resource Identifier (URI): Generic Syntax**
T Berners-Lee, R Fielding, L Masinter
RFC Editor (2005-01) https://doi.org/ggqvpr
DOI: 10.17487/rfc3986

42. **"info" URI Registry (Frozen)** https://oclc-research.github.io/infoURI-Frozen/

43. **Identifiers.org and MIRIAM Registry: community resources to provide persistent identification**
N Juty, N Le Novere, C Laibe
Nucleic Acids Research (2011-12-02) https://doi.org/cx2776
DOI: 10.1093/nar/gkr1097 · PMID: 22140103 · PMCID: PMC3245029

44. **Internationalized Resource Identifiers (IRIs)**
M Duerst, M Suignard
RFC Editor (2005-01) https://doi.org/gjvnbg
DOI: 10.17487/rfc3987

45. **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

46. **The Semantic Web — ISWC 2002**
Ian Horrocks, James Hendler (editors)
Lecture Notes in Computer Science (2002) https://doi.org/fwjmhp
DOI: 10.1007/3-540-48005-6

47. **SPARQL 1.1 Overview** http://www.w3.org/TR/sparql11-overview/

48. **State of the nation in data integration for bioinformatics.**
Carole Goble, Robert Stevens
Journal of Biomedical Informatics (2008-10-01)

DOI: 10.1016/j.jbi.2008.01.008 · PMID: 18358788

49. **Open PHACTS: semantic interoperability for drug discovery**
Antony J Williams, Lee Harland, Paul Groth, Stephen Pettifer, Christine Chichester, Egon L Willighagen, Chris T Evelo, Niklas Blomberg, Gerhard Ecker, Carole Goble, Barend Mons
Drug Discovery Today (2012-11) https://doi.org/f4fdkd
DOI: 10.1016/j.drudis.2012.05.016 · PMID: 22683805

50. **SKOS Simple Knowledge Organization System Primer** http://www.w3.org/TR/skos-primer/

51. **How matchable are four thousand ontologies on the semantic web**
Wei Hu, Jianfeng Chen, Hang Zhang, Yuzhong Qu
The semantic web: Research and applications (2011)
DOI: 10.1007/978-3-642-21034-1_20 · ISBN: 978-3-642-21033-4

52. **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

53. **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 · PMID: 28819351 · PMCID: PMC5555235

54. **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 · PMID: 25541969 · PMCID: PMC4277367

55. **A more decentralized vision for linked data**
Axel Polleres, Maulik Rajendra Kamdar, Javier David Fernández, Tania Tudorache, Mark Alan Musen
Satya Widya (2020-01-31)
DOI: 10.3233/sw-190380

56. **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)
DOI: 10.3233/ssw200033 · ISBN: 9781643681429

57. **RDF Schema 1.1** http://www.w3.org/TR/rdf-schema/

58. **OWL 2 Web Ontology Language Document Overview (Second Edition)** http://www.w3.org/TR/owl2-overview/

59. **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 · PMID: 21388572 · PMCID: PMC3105495

60. **Linked Data - Design Issues** https://www.w3.org/DesignIssues/LinkedData.html

61. **The Open Graph protocol** https://ogp.me/

62. **RDFa 1.1 Primer - Third Edition** http://www.w3.org/TR/rdfa-primer/

63. **HTML Standard** https://html.spec.whatwg.org/multipage/microdata.html

64. **JSON-LD 1.1** https://www.w3.org/TR/json-ld/

65. **Usage Statistics of JSON-LD for Websites, May 2022** https://w3techs.com/technologies/details/da-jsonld

66. **Designing a Linked Data developer experience** (2018-12-28)
https://ruben.verborgh.org/blog/2018/12/28/designing-a-linked-data-developer-experience/

67. **Shapes Constraint Language (SHACL)** https://www.w3.org/TR/shacl/

68. **Shape Expressions (ShEx) 2.1 Primer** http://shex.io/shex-primer/

69. **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, Andra Waagmeester
The semantic web: 16th international conference, ESWC 2019, portorož, slovenia, june 2–6, 2019, proceedings (2019)
DOI: 10.1007/978-3-030-21348-0_39 · ISBN: 978-3-030-21347-3

70. **Validating RDF Data**
Jose Emilio Labra Gayo, Eric Prud'hommeaux, Iovka Boneva, Dimitris Kontokostas
Synthesis Lectures on the Semantic Web: Theory and Technology (2017-09-28) https://doi.org/ghks5j
DOI: 10.2200/s00786ed1v01y201707wbe016

71. **Survey of tools for linked data consumption**
Jakub Klímek, Petr Škoda, Martin Nečaský
Satya Widya (2019-05-23)
DOI: 10.3233/sw-180316

72. **The semantic web identity crisis: In search of the trivialities that never were**
Ruben Verborgh, Miel Vander Sande
Satya Widya (2020-01-31)
DOI: 10.3233/sw-190372

73. **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 · PMID: 21622664

74. **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

75. **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

76. **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

77. **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

78. **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

79. **Handbook of computer-communications standards: The open systems (OSI) model and OSI-related standards**
William Stallings
Sams (1990)
ISBN: 9780672226977

80. **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

81. **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 - ESEC/FSE 2017 (2017-09-04)
http://dl.acm.org/citation.cfm?doid=3106237.3121282
DOI: 10.1145/3106237.3121282 · ISBN: 9781450351058

82. **OpenAPI Specification v3.1.0 | Introduction, Definitions, & More** https://spec.openapis.org/oas/v3.1.0.html

83. **Schema.org Actions - schema.org** https://schema.org/docs/actions.html

84. **Web Services Description Language (WSDL) Version 2.0 Part 0: Primer** http://www.w3.org/TR/wsdl20-primer/

85. **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/ggbwf4
DOI: 10.1093/nar/gkt328 · PMID: 23640334 · PMCID: PMC3692062

86. **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 · PMID: 34804501 · PMCID: PMC8573700

87. **The Modern Standards Paradigm - Five Key Principles**
OpenStand
https://open-stand.org/about-us/principles/

88. **First International Conference on FAIR Digital Objects**
Tina Loo (editor)
Pensoft Publishers (1970) https://doi.org/grcqxh
DOI: 10.3897/rio.coll.190

89. **DOIP API for HTTP Clients — Cordra documentation** https://www.cordra.org/documentation/api/doip-api-for-http-clients.html

90. **DOI Resolution Documentation** https://www.doi.org/factsheets/DOIProxy.html

91. **DOI Handbook - Resolution** https://www.doi.org/doi_handbook/3_Resolution.html

92. **HTTP Semantics**
R Fielding, M Nottingham, J Reschke (editors)
RFC Editor (2022-06) https://doi.org/gqprqb
DOI: 10.17487/rfc9110

93. **Linked data platform 1.0**
Linked Data Platform Working Group
W3C (2015-02-26) https://www.w3.org/TR/2015/REC-ldp-20150226/

94. **FAIR Digital Object Framework Documentation** https://fairdigitalobjectframework.org/

95. **Handle System Overview**
S Sun, L Lannom, B Boesch
RFC Editor (2003-11) https://doi.org/ggn83z
DOI: 10.17487/rfc3650

96. **Web API design best practices - Azure Architecture Center**
EdPrice-MSFT
https://learn.microsoft.com/en-us/azure/architecture/best-practices/api-design

97. **Data Information View** https://handle-esgf.dkrz.de/lp/21.14100/2fcf49d3-0608-3373-a47f-0e721b7eaa87

98. **Data Catalog Vocabulary (DCAT) - Version 3** https://www.w3.org/TR/2022/WD-vocab-dcat-3-20220510/

99. **ORE User Guide - Primer** http://www.openarchives.org/ore/1.0/primer

100. **PROV-O: The PROV Ontology** http://www.w3.org/TR/2013/REC-prov-o-20130430/

101. **Semantic Web Services**
Dieter Fensel, Federico Michele Facca, Elena Simperl, Ioan Toma
Springer Berlin Heidelberg (2011) https://doi.org/bv7nnc
DOI: 10.1007/978-3-642-19193-0

102. **5-star Open Data** http://5stardata.info/en/

103. https://www.rd-alliance.org/sites/default/files/Cordra.2022.pdf

104. **Global Internet Phenomena Report 2022**
Sandvine
https://www.sandvine.com/global-internet-phenomena-report-2022

105. **Hypertext Transfer Protocol Version 2 (HTTP/2)**
M Belshe, R Peon
RFC Editor (2015-05) https://doi.org/gp32q9
DOI: 10.17487/rfc7540

106. https://blog.cloudflare.com/http-3-vs-http-2/

107. **draft-ietf-quic-http-34** https://datatracker.ietf.org/doc/html/draft-ietf-quic-http-34

108. **QUIC: A UDP-Based Multiplexed and Secure Transport**

J Iyengar, M Thomson (editors)
RFC Editor (2021-05) https://doi.org/gkctrr
DOI: 10.17487/rfc9000

109. **URI Template**
J Gregorio, R Fielding, M Hadley, M Nottingham, D Orchard
RFC Editor (2012-03) https://doi.org/gp33dw
DOI: 10.17487/rfc6570

110. **Content negotiation - HTTP | MDN** https://developer.mozilla.org/en-US/docs/Web/HTTP/Content_negotiation

111. **Hypertext Style: Cool URIs don't change.** https://www.w3.org/Provider/Style/URI

112. **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

113. **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

114. **HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV)**
L Dusseault (editor)
RFC Editor (2007-06) https://doi.org/gp37bf
DOI: 10.17487/rfc4918

115. **Upgrading to TLS Within HTTP/1.1**
R Khare, S Lawrence
RFC Editor (2000-05) https://doi.org/gp33dv
DOI: 10.17487/rfc2817

116. **Hypertext Style: Cool URIs don't change.** https://www.w3.org/Provider/Style/URI.html

117. **draft-ietf-mediaman-suffixes-00 - Media Types with Multiple Suffixes** https://datatracker.ietf.org/doc/draft-ietf-mediaman-suffixes/00/

118. **The 'profile' Link Relation Type**
E Wilde
RFC Editor (2013-03) https://doi.org/gp32q7
DOI: 10.17487/rfc6906

119. **W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures** http://www.w3.org/TR/xmlschema11-1/

120. **JSON-LD 1.1** http://www.w3.org/TR/json-ld/

121. **WebSockets Standard** https://websockets.spec.whatwg.org/

122. **Linked Data Notifications** https://www.w3.org/TR/ldn/

123. **The Atom Publishing Protocol**
J Gregorio, B de hOra (editors)
RFC Editor (2007-10) https://doi.org/gp4p2c
DOI: 10.17487/rfc5023

124. **SWORD 3.0 Specification** https://swordapp.github.io/swordv3/swordv3.html

125. **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

126. **Hydra W3C Community Group** https://www.hydra-cg.com/

127. **draft-kelly-json-hal-08** https://datatracker.ietf.org/doc/html/draft-kelly-json-hal-08

128. **Web Linking**
M Nottingham
RFC Editor (2017-10) https://doi.org/gf8jcd
DOI: 10.17487/rfc8288

129. **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

130. **HTTP Live Streaming**
W May
RFC Editor (2017-08) https://doi.org/gp32rc
DOI: 10.17487/rfc8216

131. **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

132. **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

133. **NCBO BioPortal** https://bioportal.bioontology.org/ontologies

134. **The Globus Striped GridFTP Framework and Server**
W Allcock, J Bresnahan, R Kettimuthu, M Link
ACM/IEEE SC 2005 Conference (SC'05) https://doi.org/cgmc2b
DOI: 10.1109/sc.2005.72

135. **The ARK Identifier Scheme** https://datatracker.ietf.org/doc/id/draft-kunze-ark.html

136. **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

137. **Handle.Net Registry** https://www.handle.net/download_hnr.html

138. **ISO 16684-1:2019**
14:00-17:00
ISO https://www.iso.org/standard/75163.html

139. **Data - W3C** https://www.w3.org/standards/semanticweb/data

140. **Linking Across Provenance Bundles** https://www.w3.org/TR/2013/NOTE-prov-links-20130430/

141. **Introducing 'Role'**
Unknown
http://blog.schema.org/2014/06/introducing-role.html

142. **PROV-O: The PROV Ontology** https://www.w3.org/TR/prov-o/#specializationOf

143. **ORE Specification - Abstract Data Model** http://www.openarchives.org/ore/1.0/datamodel#Proxies

144. **Schema.org - Schema.org** https://schema.org/

145. **DCMI Metadata Terms** https://www.dublincore.org/specifications/dublin-core/dcmi-terms/

146. **Data Catalog Vocabulary (DCAT) - Version 2** https://www.w3.org/TR/vocab-dcat-2/

147. **FOAF Vocabulary Specification** http://xmlns.com/foaf/spec/

148. **PAV ontology: provenance, authoring and versioning**
Paolo Ciccarese, Stian Soiland-Reyes, Khalid Belhajjame, Alasdair JG Gray, Carole Goble, Tim Clark
Journal of Biomedical Semantics (2013) https://doi.org/gftcpx
DOI: 10.1186/2041-1480-4-37 · PMID: 24267948 · PMCID: PMC4177195

149. **DCMI Metadata Terms** (2020-01-20) https://www.dublincore.org/specifications/dublin-core/dcmi-terms/2020-01-20/

150. **Data Catalog Vocabulary (DCAT) - Version 2** https://www.w3.org/TR/2020/REC-vocab-dcat-2-20200204/

151. **Bioschemas - Bioschemas** http://bioschemas.org/

152. **Darwin Core: An Evolving Community-Developed Biodiversity Data Standard**
John Wieczorek, David Bloom, Robert Guralnick, Stan Blum, Markus Döring, Renato Giovanni, Tim Robertson, David Vieglais
PLoS ONE (2012-01-06) https://doi.org/fzrpwq
DOI: 10.1371/journal.pone.0029715 · PMID: 22238640 · PMCID: PMC3253084

153. **Semantic Web - XML2000 - slide "Architecture"** https://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html

---

1. For a brief introduction to DOIP 2.0 [23], see [35].↩

2. URIs [41] 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 [42], 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) [43]. RDF 1.1 formats use Unicode to support IRIs [44], which extends URIs to include international characters and domain names.↩

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* [45].↩

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 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.↩

6. [**WD-RequirementSpec-1.0-20220317?**] renames _FDOF*_ to _FDOR*_, FDOF3/FDOF4 are swapped to FDOR4/FDOR3. ↩

7. The `http` protocol (port 80) can in theory also upgrade [115] 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 [80], 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.↩