How FDO attributes can support machine- and human-readability? - a description along three examples

Ulrich Schwardmann[‡], Tibor Kálmán[‡]

‡ GWDG, Göttingen, Germany

Corresponding author: Ulrich Schwardmann (uschwar1@gwdg.de)

Academic editor: Francisco Andres Rivera Quiroz

Abstract

Based on the notion of a FAIR Digital Object (FDO) record, which consists of key-value pairs as attributes that are precisely defined in a Data Type Registry and selected in a profile, we show three examples of FDOs from different viewpoints how FDO records can be implemented as Handle PID records. As references to the attribute definitions, the keys determine the value space of the attribute. In the first two examples, the profiles enable human-readable keys and legacy digital objects to be integrated into FDO records. How legacy metadata from IANA media types that can be transformed into structured metadata of appropriate attribute definitions that then can be applied in profiles and FDO records, is described in the third example.

Keywords

FAIR Digital Object, FDO record, FDO profile, PID for Instruments, DARIAH repository, MIME type

Introduction

FAIR Digital Objects (FDOs) are typed by a well-described set of attributes, where attributes are key value pairs with a key that refers to a clear specification of the value. On one hand, such exact descriptions of the attribute keys allow machines to interpret values and, thus, they are crucial for machine actionability. On the other hand, a proper description of attributes can be a way to also enable human-readability of the used keys. Nevertheless, machine- and human-readability do not have to be mutually exclusive. Based on the outcome of the RDA working group on Data Type Registries (Lannom et al. 2015), one of the authors describes in detail how attribute keys can be defined

hierarchically in a machine- and human-readable way (Schwardmann 2016). The examples shown in the following describe how this works in practice.

This article is structured in the following way. First, we provide a brief overview of the FDO Record and FDO Profile utilised to structure attributes, as well as the generic concepts behind them. Following that, we shift our focus to three examples of FDO types with their attributes. Each of the examples highlight different viewpoints and approaches to build up FDOs.

Prerequisites

Here, we introduce the prerequisites that are needed across all sections. We describe the FDO Record and FDO Profile related to structure attributes. We also present some generic concepts behind them.

FDO Record: the set of attributes

The well-described set of attributes is integrated into an FDO Record, which is the core data abstraction that allows a machine to process any FDO in a deterministic manner. An FDO record is the structure that is returned when resolving a reference via a Persistent Identifier (PID).

The FDO Record consists of a set of attributes, each consisting of a key value pair (see Fig. 1). The key as string is always a reference to a definition of the value space (schema) which is machine readable. These definitions are themselves FDOs again. In such a way, the key determines the structure of the value and the values of the attributes always have to comply with the definition provided by the key. In general, such value spaces are determined by schematas and the schematas can be derived canonically from the attribute definitions. Such a structure could be serialised in many different manners.

FDO Profile: the expected attributes

The structure of the FDO record needs to be described in a way that machines know what they can expect in the record and how they should operate on it. This description is called an FDO profile. It contains a list of needed and allowed attribute keys for which the FDO record provides values.

For machine actions also, this FDO profiles have schemata, the so-called profile schemata, which can be found in a schema repository as part of the FDO framework. Like the attribute definitions, both the FDO profiles, as well as the schemata for them, are again FDOs.

The following attributes should be provided by an FDO record for machine actionability:

- A mandatory FDO profile attribute that describes the valid attributes in the FDO record.

- A Data attribute allowing the client to access the data associated with the FDO.
- An FDO Type attribute that enables the client to quickly determine the type of the data associated with the FDO.
- A metadata attribute that allows the client to obtain the metadata describing the FDO record and the data with which it is associated.
- Other optional community specific attributes: these are the FDO attributes that a client would look for with respect to the specifications given in the profile. It allows the client to obtain specific metadata that has an important role for community specific operations and selection processes.

Some of this community specific metadata information will be used for describing access control and licensing information. It is possible that multiple metadata attributes will be used to specify different aspects of the FDO record.

Even if not all of the requirements above are fulfilled by the examples below, we will show what has to done additionally to fulfil them.

Three Examples of FDO Types

In this article, we present three examples of FDO types with their attributes. Each of the examples highlight different viewpoints and approaches to build up FDOs.

The first use case (PIDs for instruments) and the second use case (DARIAH PID structure) are examples, where the values of all used attributes are provided inline in the FDO record itself. It is shown how one can integrate legacy attribute sets, providing inside repositories for the description of their digital objects and that are based on human-readable keys, into a machine readable setting.

The third use case (media types) is an example, where the values of the FDO record are provided as references to media type instances stored in a type registry. The differences of this approach and the automated ways of distinction between these two approaches are then discussed in the following. section.

We point out that, across the examples, there are some common characteristics selected with respect to the underlying technology:

- the Handle system is used for the persistent identifiers.
- the FDO record is provided by the Handle record of the PID.
- all the provided attributes can be found here as type-data pairs in the phrasing of the Handle system.

In the following, we introduce the three examples in details.

The PID for Instruments Example

The PID for instrument example goes back to the development of kernel metadata, which is seen as minimally required to reference and describe scientific instruments (Stocker et al. 2020). The value space for the attributes here contains often hierarchical objects and can also be lists of attributes.

An example of such an attribute definition is that of a single manufacturer of an instrument* [1] that occurs in a list of manufacturers.

Handle Record of a PID for Instruments

The example we provided uses references to the attribute definitions as keys for the values which are often lists or objects. The Handle Record of a PID for Instruments*[2] with the full list of attributes can be obtained from the Handle Proxy.

The structure of this Handle record complies to the description an FDO record in Fig. 1. Additionally, this structure is provided as an Information Profile*[3] at the ePIC Date Type Registry (Schwardmann 2021) and is used as profile for the Handle record*[4] of *hdl:21.T11998/0000-001A-3905-1*. In this Handle record, its profile is referenced with key *0.TYPE/0.PROFILE*.

However, since the relationship between the uniquely-defined attribute definitions and their semantical names is also provided by this profile, we can use these names in the Handle record as keys. In this way, an FDO record*[5] with the same values, but names from the profile as keys, obtains a human-readable form without losing any machine readability, because the names are resolved to unique attribute definitions by the referred profile.

In both cases, the full instrument descriptions are completely stored in the Handle database of the Handle PID service. The PID itself is a metadata object and can be seen as an FDO of its own.

This example fulfils almost completely the requirements for an FDO record. The profile almost completely describes the existing Handle record. The only attribute that is missing in the profile derived from the metadata for instruments in [3] is that with key *URL*. *URL* is a generic Handle attribute for resolution, here used for the reference to a landing page with additional metadata about the instrument. It points to an XML file, but this type notification is missing in the record as well. Without the URL attribute, the Handle does not redirect to a resource, but the requirements for an FDO record would be fulfilled.

PID4Inst in a Repository

Another option is to store the metadata objects of instrument descriptions in repositories. In this case, a schema is needed to describe the metadata elements that are needed for the description. The existing attribute definitions can be bundled into a single complex type

definition and, for this definition, the profile for instruments above can be used. Even if profiles are conceptually different from attribute definitions, they are syntactically similar and, therefore, exploitable by the same services.

From this complex profile - interpreted as a type definition - one can derive a schema for the repository entries. The result of such a schema derivation can then be fed into the ingest module of a repository, for example, as Properties4Instruments schema*[6] into the Cordra*[7] schema module for the definition of attribute types. In the derived schema, only some small changes were made, like for auto-generation of Handles and access dates.

As a next step, PID for instrument objects can be defined in the repository. (an example of such an object is here*[8]).

The DARIAH Example

This example evolved in the digital humanities in the context of the German DARIAH project (Kálmán et al. 2016, Kálmán et al. 2019), when the DARIAH repository*[9] was designed several years ago. The Handle record structure (at Fig. 2 and here is an example *[10]) was created long before FDO records had been discussed.

It uses key value pairs with human-readable keys as the type and provides relatively atomic values. The key here is a human-readable description for the value space that can be expected.

The use of human-readable keys, however, does not match the goal of machine readability of this desciption and additionally has the risk of uncertainty and ambiguity.

Attribute Definitions

In order to make these attributes machine readable, attribute definitions for the allowed value spaces have been stored in the ePIC data type registries. The basic information type for an email address*[11], for instance, can be used as the reference key for the value space given for the 'RESPONSIBLE' type above.

Attribute definitions for all the other attributes used in the DARIAH example are also provided by the ePIC data type registries.

An FDO Profile of Legacy Repository Records

In this way, one is able to define a profile for the legacy DARIAH Handle records*[12].

If this profile is the known profile of all objects in the DARIAH repository, the named references to the keys in the profile disambiguate the human-readable form of the Handle record.

Usually - and as we have seen in the previous PID4Inst example - the profile of the FDO would be another attribute of the FDO. This would require an adaption of the attributes of all digital objects of the DARIAH repository. Since all digital objects of the DARIAH repository follow the same profile and all its digital objects have the same PID prefix, it would be sufficient to implement this additional attribute at the prefix level. Together with a rule that attributes on a lower level dominate attributes on a higher level, this additional prefix attribute would make FDOs out of legacy digital objects that have been defined a long time ago.

Additionally, this example fulfils almost completely the requirements for an FDO record. Again, the URL attribute is not covered by the profile and the type of the referenced data, in this case, an HTML landing page, is not specified.

The Media Type Example

This example is based on the Internet Assigned Numbers Authority IANA media types*[13] (formerly known as MIME types).

RFCs and Templates

A IANA media type is given as a subtype of one of the content type directories application, audio, font, example, image, message, model, multipart, text or video and is described by a reference and a set of metadata attributes that are provided for each media type in a special file. Although this is an abuse of terminology, this file is called "template", because it is based on a template that describes a couple of key words that have to or should be filled in with unstructured text.

However, no template is provided for older formats like gif, jpeg, mpeg, plain or richtext. Here, the given references are pointing to RFC2045 and RFC2046 which are descriptions of Multipurpose Internet Mail Extensions (MIME). These media types are mentioned there, but they are neither standardised nor described.

The references for IANA media types point, in two thirds of the cases, to mail addresses of contact persons. Here the "template" is the only publicly available resource of information about the standard. Another less then a third of the cases refer to publicly available and human-readable RFCs. The small remainder refers to standards of other organisations without a directly referred resource of information about the standard.

A Type Definition for Media Types

Two options are available to provide attributes for IANA media types in FDO records. The first is a type definition as an enumeration of all IANA subtypes of the content types. An example for a controlled vocabulary*[14] of all IANA subtypes of the content type application can be found in the ePIC DTR.

Type definitions for the other content types can be described in a similar way. Here, a media type is given by an attribute, consisting of the reference to this type definition as key and a value out of the enumeration list of this type definition. This approach completely relies on the media type naming of IANA and does not provide information about any of the in-parts valuable additional metadata given by the IANA resources.

A second approach is a type definition for media types that covers the metadata as they are given by IANA, which is the information given by the "templates" and the IANA media type resource ([8]). In this case, a media type is an attribute consisting of the type definition for this metadata structure and a value that refers to a registered metadata record containing the information about a IANA media type. These references given as values are PIDs that represent the media type registered as FDO records in data type registries.

Obtian Structured Metadata Information from IANA Templates

The metadata elements in IANA "templates" are described in details at RFC 6838*[15] and are based on a couple of key words. Since the values in the "templates" are in most cases unstructured text, they are, in general, not machine readable.

However, if such a "template" is provided and it contained at least the keywords of the template, much of the information provided can be filtered out in an automated way (with some heuristics). This information can then be filled in a structured and machine-readable set of metadata elements for the media types that can be registered as an FDO representation of a IANA media type.

The structure for the registered metadata information is inspired by the keywords of the IANA template and some other information that can be extracted from the IANA media type presentation.

The content of this media-type metadada contains values for contact information, required or optional parameters, security, fragment identifier and interoperability considerations, applications, intended usage and restrictions on usage. There is additional information provided with deprecated alias names for this type, like magic number(s), file extension(s), Macintosh File Type Code(s) and Object Identifier(s) (OIDs). Additionally, in some cases, the encoding information is provided with four possible values: "7-bit", "8-bit", "binary" or "framed".

The structure is given by the type definition media-type-attributes*[16] for media type attributes and, from this, the type definition a schema*[17] for registration of new entries in the data type registry is derived.

All more than two thousand IANA media types are defined as types in the data type registry, based on this schema and can be viewed by a search on *type:"media-type-schema"**[18].

In this way, the second approach with a type definition, based on IANA type metadata, has the advantage that it provides a structured access to some useful additional information

beyond the media (sub-)type name. For instance, related or describing RFCs and other standards, responsible authorities, used file extensions and dates for registration, publishing and updates and other provenance information, as well as some rudimentary information about the encoding, can be found there.

These IANA media types in the data type registry now can be used as values in attributes of FDO records, represented by the PID of their definition. The key of such attributes would be the reference to the definition of their type media-type-attributes with PID 21.T11148/bc0e376cd6a01a2f8071. Therefore, here the question whether the Handle record fulfils the requirements for an FDO record is not relevant, because the example does not describe an FDO record itsself.

Operations on Data

This metadata information does, however, not describe the encoding in such a detail that it enables automated operations working on the bitstream structure of the typed data itself.

Automated processing of the data only works for specialised operations that are known to work on the basis of the media (sub-)type names or the PIDs referring to the registered metadata belonging to these media (sub-)type names. This is the case for both approaches of type definitions, either controlled vocabulary or registered metadata.

The question, which operations can and should be used for further processing, is answered in current user environments by special knowledge or user specific configuration of the software context. Browsers, for instance, often have specific configurable settings that determine the use of operations depending on file extensions.

A similar context-specific approach with preselected operations is possible for FDOs depending on media type attributes. The operations that work on data with a given media type attribute in the FDO record still need to be built by humans with the deeper understanding of the human-readable specification documents (RFCs) that describe the bitstream structure.

Ideally, it should also be possible to ask operation-repositories for suggestions of suitable operations depending on their media type attributes and the specifics of a certain software context. How such an operation infrastructure should be set up still has to be discussed and developed.

Media Types in FDO Records

As described above, the media type is given in an FDO record as an attribute with key. The key is either type definition for the controlled vocabulary of media (sub-)type names or it is the type definition for the media type metadata. As value, it is either an instance of a media (sub-)type name or a PID reference to an instance of media type metadata. Here, we suggest to use the latter option, because it can provide more information. If additional

information is needed on the reference, like the media (sub-)type name or used RFCs etc., then an additional attribute with appropriate inline values needs to be used.

In any case, the usage of either attribute needs to be announced in the profile for the FDO, such that an automated process knows in advance, what it can expect as information on the reference level and how it can achieve further information, if necessary.

Media-like Types

The approach of a type definition for the media-type metadata can also be applied to similar data-use cases that do not have a IANA media type standard for its data encoding (see Fig. 3). This might be of particular interest for community internal agreements about data encoding that are sufficiently precisely described and agreed inside the community, but where the effort to define a standard is not made.

In such cases, a couple of metadata keys should be defined as mandatory and especially a reference to the encoding description should be made.

In a further step into the direction of automation, the encoding description itself should be made explicitly machine readable, as it is already possible with a couple of such approaches for data serialisation systems, like FITS, ProtoBuf, Avro, Parquet, Arrow and HDF5.

Summary

We highlighted different approaches to build up FDOs with three examples. Each of the examples presented FDO types with their attributes. The “PIDs for instruments” use case and the “DARIAH PID structure” example provided the values of all used attributes inline in the FDO record itself. Furthermore, the “DARIAH” use-case also showed how FDOs can be made out of legacy digital objects, where legacy attribute sets were provided a long time ago. We also described, how machine- and human-readability can be supported by FDO types at the same time. Our third use-case, the “media types” example, described FDO types, where the values of the FDO record are provided as references to media type instances stored in a type registry.

Conflicts of interest

The authors have declared that no competing interests exist.

References

- Kálmán T, Kong X, Schwardmann U (2016) Die digitale Forschungsinfrastruktur DARIAH-DE: Angebotspalette für die Geistes- und Kulturwissenschaften. Bibliothek Forschung und Praxis https://doi.org/10.1515/bfp-2016-0041

- Kálmán T, Ďurčo M, Fischer F, Larrousse N, Leone C, Mörth K, Thiel C (2019) A landscape of data – working with digital resources within and beyond DARIAH. International Journal of Digital Humanities 1 (1): 113-131. https://doi.org/10.1007/s42803-019-00008-6
- Lannom L, Broeder D, Manepalli G (2015) Data Type Registries working group output. https://doi.org/10.15497/A5BCD108-ECC4-41BE-91A7-20112FF77458
- Schwardmann U (2016) Automated schema extraction for PID information types. 2016 IEEE International Conference on Big Data (Big Data) https://doi.org/10.1109/bigdata.2016.7840957
- Schwardmann U (2021) The ePIC PID Information Type Registry. GRO.data https://doi.org/10.25625/9DNRSJ/FO8H5Z
- Stocker M, Darroch L, Krahl R, Habermann T, Devaraju A, Schwardmann U, D'Onofrio C, Häggström I (2020) Persistent Identification of Instruments. Data Science Journal https://doi.org/10.5334/dsj-2020-018

Endnotes

*1 https://dtr-test.pidconsortium.eu/#objects/21.T11148/7adfcd13b3b01de0d875
*2 https://hdl.handle.net/21.T11998/0000-001A-3905-1?noredirect
*3 https://dtr-test.pidconsortium.eu/objects/21.T11148/17ce618137e697852ea6
*4 https://hdl.handle.net/21.T11998/0000-001A-3905-1?noredirect
*5 https://hdl.handle.net/21.T11998/0000-001A-3905-8?noredirect
*6 https://hdl.handle.net/21.T11148/c2c8c452912d57a44117
*7 https://www.cordra.org
*8 https://vm11.pid.gwdg.de:8445/objects/21.11145/8fefa88dea40956037ec
*9 https://de.dariah.eu/en/repository
*10 https://hdl.handle.net/21.11113/0000-000B-CA4C-D?noredirect
*11 https://dtr-test.pidconsortium.eu/#objects/21.T11148/e117a4a29bfd07438c1e
*12 https://dtr-test.pidconsortium.eu/#objects/21.T11148/f1eea855587d8b1f66da
*13 https://www.iana.org/assignments/media-types/media-types.xhtml
*14 https://dtr-test.pidconsortium.net/#objects/21.T11148/edff7f2829db22e260a3
*15 https://www.iana.org/go/rfc6838
*16 https://dtr-test.pidconsortium.eu/#objects/21.T11148/bc0e376cd6a01a2f8071
*17 https://dtr-test.pidconsortium.net/#objects/21.T11148/e9c41dba96b7ba84e058
*18 https://dtr-test.pidconsortium.net/#objects/?query=type:%22media-type-schema%22

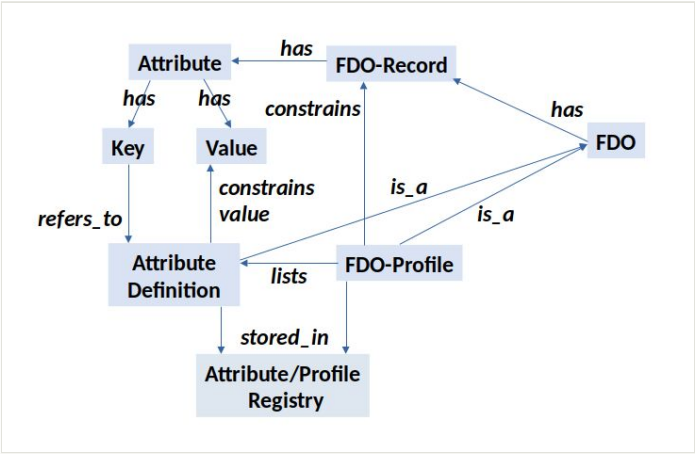
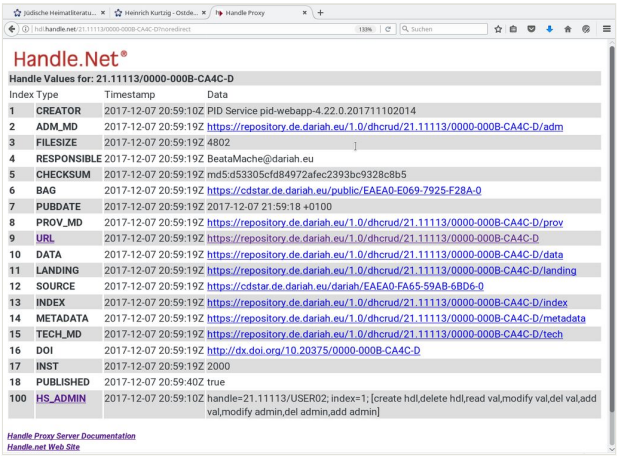


Figure 1.

The FDO record determined by attribute definitions and profiles.



Handle.Net®

Handle Values for: 21.11113/0000-000B-CA4C-D

Index	Type	Timestamp	Data
1	CREATOR	2017-12-07 20:59:10Z	PID Service pid-webapp-4.22.0.201711102014
2	ADM_MD	2017-12-07 20:59:19Z	https://repository.de.dariah.eu/1.0/dhcrud/21.11113/0000-000B-CA4C-D/adm
3	FILESIZE	2017-12-07 20:59:19Z	4802
4	RESPONSIBLE	2017-12-07 20:59:19Z	BeataMache@dariah.eu
5	CHECKSUM	2017-12-07 20:59:19Z	md5:d53305cfd84972afec2393bc9328c8b5
6	BAG	2017-12-07 20:59:19Z	https://cdstar.de.dariah.eu/public/EAEA0-E069-7925-F28A-0
7	PUBDATE	2017-12-07 20:59:19Z	2017-12-07 21:59:18 +0100
8	PROV_MD	2017-12-07 20:59:19Z	https://repository.de.dariah.eu/1.0/dhcrud/21.11113/0000-000B-CA4C-D/prov
9	URL	2017-12-07 20:59:19Z	https://repository.de.dariah.eu/1.0/dhcrud/21.11113/0000-000B-CA4C-D
10	DATA	2017-12-07 20:59:19Z	https://repository.de.dariah.eu/1.0/dhcrud/21.11113/0000-000B-CA4C-D/data
11	LANDING	2017-12-07 20:59:19Z	https://repository.de.dariah.eu/1.0/dhcrud/21.11113/0000-000B-CA4C-D/landing
12	SOURCE	2017-12-07 20:59:19Z	https://cdstar.de.dariah.eu/dariah/EAEA0-FA65-59AB-6BD6-0
13	INDEX	2017-12-07 20:59:19Z	https://repository.de.dariah.eu/1.0/dhcrud/21.11113/0000-000B-CA4C-D/index
14	METADATA	2017-12-07 20:59:19Z	https://repository.de.dariah.eu/1.0/dhcrud/21.11113/0000-000B-CA4C-D/metadata
15	TECH_MD	2017-12-07 20:59:19Z	https://repository.de.dariah.eu/1.0/dhcrud/21.11113/0000-000B-CA4C-D/tech
16	DOI	2017-12-07 20:59:19Z	http://dx.doi.org/10.20375/0000-000B-CA4C-D
17	INST	2017-12-07 20:59:19Z	2000
18	PUBLISHED	2017-12-07 20:59:40Z	true
100	HS_ADMIN	2017-12-07 20:59:10Z	handle=21.11113/USER02; index=1; [create hdl,delete hdl,read val,modify val,del val,add val,modify admin,del admin,add admin]

Handle Proxy Server Documentation
Handle.net Web Site

Figure 2.

The Handle record of a legacy digital object in the DARIAH repository.

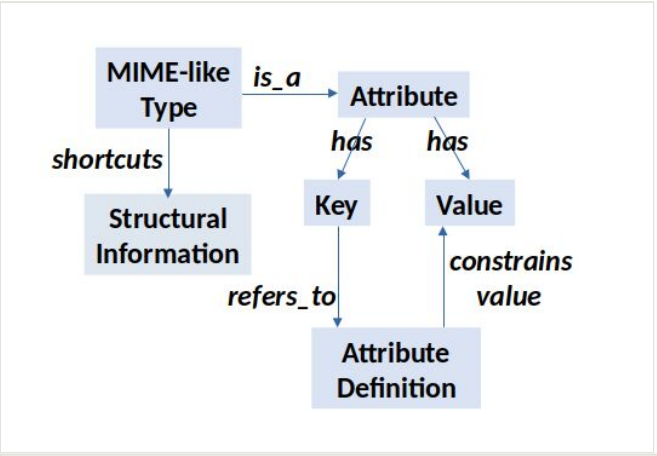


Figure 3.

Community defined types similar to MIME types could be a light-weight way to enable operations on FDOs.