Report of the RDA GEDE Web-Workshop on Adapting Repositories to DOIP from 22.5.2019 - Version 1

Zsuzsanna Szeredi, Peter Wittenburg, Giridhar Manepalli, Christophe Blanchi, Paul Trilsbeek, Robert Quick, Lu

This report will not cover the content of the presentations which can be seen on the web-site <https://github.com/GEDE-RDA-Europe/GEDE/tree/master/Digital-Objects/DO-Workshops/Web-Workshop-22-May-2019>. It will only report on discussion points raised during the meeting and contribute to this discussion. This version will be made available also as a Google Doc (??) allowing participants and others to comment. Based on this discussion a new version will be created.

# Forum Type

We tried to come to a more interactive interaction by discussing two concrete cases of adaptations and not only involve some experts in these discussions, but also the participants. A few observations can be made:

* In general we had few interventions from other than the presenters and the experts. The virtual medium does not easily support a more agile, fluent intervention style. Perhaps the participants liked the interactive style between the case presenters and the experts.
* The chat box was used quite actively by some which seems to be a more easy going way to make comments.
* However, these chats can also be seen as very much hampering discussions for the following reasons:
  + The active people can hardly focus on what is being said and in addition to follow the chats - some may be capable, others not.
  + Making a chat comment prevents an open discussion including the active people except we take measures.
  + Not all chat comments are meant as comments to the debate, some are just meant to give URLs etc. which is very helpful.
  + We seem to need an expert to follow the chats and to take measures when a "content comment" is being made, i.e., we need an additional person following the chats and raising them to the surface of the discussions when appropriate.
  + After this round it is obvious that one contribution and two case studies is too much for a 2 hour session. Already dealing with two case studies in two hours is a lot, since we first need to give the time to the participants what the case studies subject and problems/solutions are.

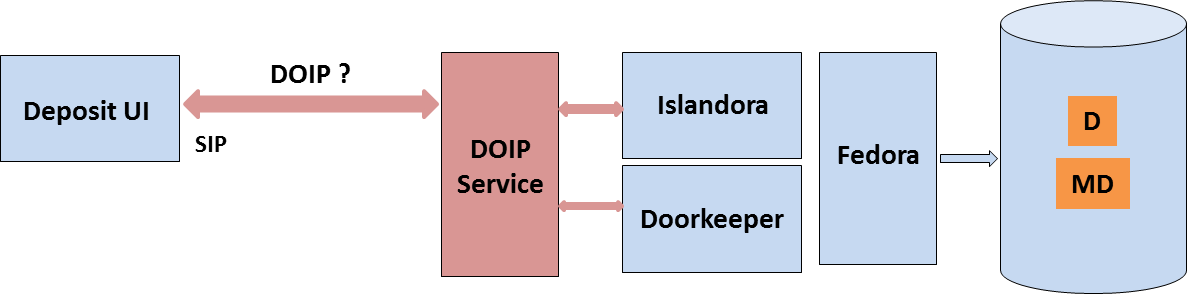
# DOBES Case

The main questions addressed were:

* Where to put some "logic" special to the approach chosen?
* Is the bundle concept compliant with the RDA collection proposal?
* How could this archive take profit from using DOIP?
* Should we think of a generic Fedora/D-Space solution and would it help in the case of DOBES?
* The Gatekeeper in DOBES executes a number of steps which will be repeated (partly) in other repositories? Can we think about a more generic workflow style?

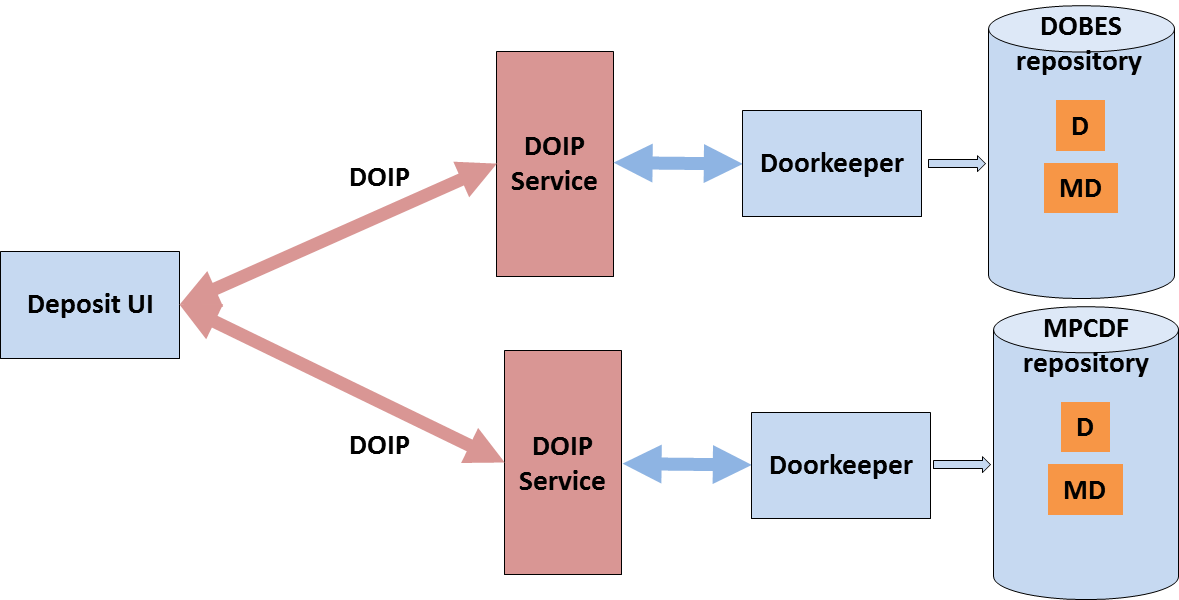
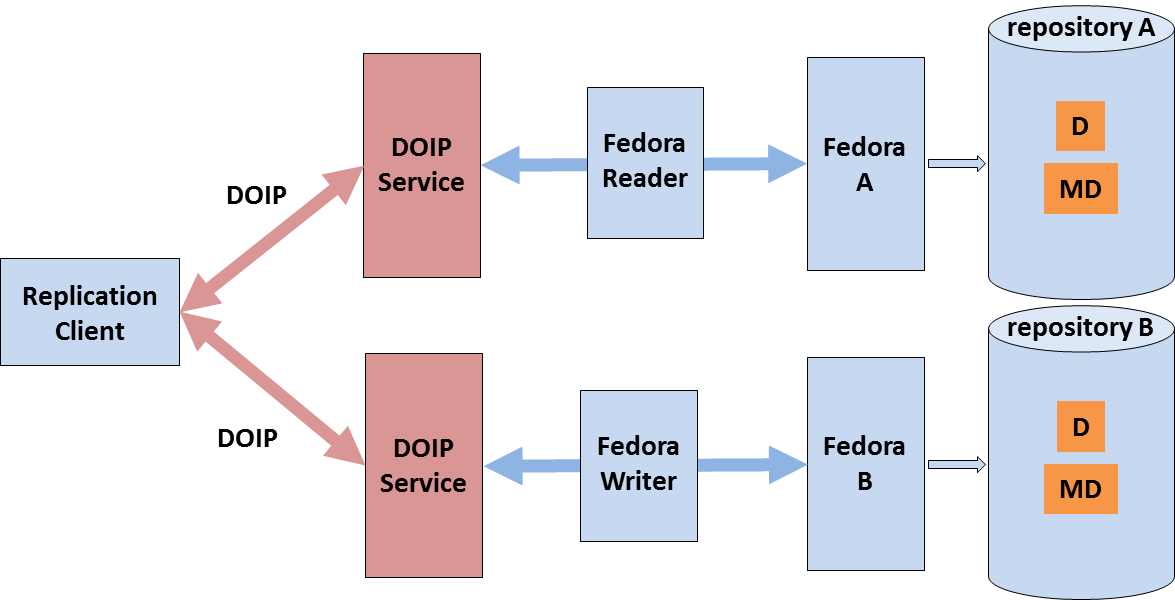
In the following we will address each question to continue the debates.

**Where to put some "logic" special to the approach chosen?**



The doorkeeper performs quite a number of important functions for each uploaded "information package" submitted by the Deposit Client which any user can use to upload sets of data and metadata. It seems to be a straightforward way to register the doorkeeper function as a DOIP operation with a Data Type Registry, define a specific type (SIP), give it an ID and then let the Deposit Client send a DOIP request including the Depositor ID, the Doorkeeper ID and the SIP as additional parameter. The DOIP service will interpret the DOIP request and start the Doorkeeper. Nothing needs to be changed except that the DTR entry needs to be specified, the DOIP service needs to be implemented and the Deposit Client needs to be adapted to generate the DOIP request. This all seems to be a doable task within a limited time including intensive testing.

The question is then whether the DOBES archive would gain anything by adapting?

* At first glimpse nothing will be gained.
* Assuming that the Doorkeeper function is available as a quickly downloadable image one could imagine that it is very easy to use the same setup in front of another archive instance. This is indicated in the diagram. Currently 4 dynamic copies of all uploaded data/metadata are being created. The inefficiency would be that all steps of checking etc. would have to be redone in the other archive instances. Since Fedora is a widely used repository system we assume here that the other repositories also support this package.
* A better option would be to do all checks at the "core" archive, since partly manual curation steps need to be taken, and to then do replication of all relevant data and metadata to create copies. Currently, the archive is using Rsync at the physical level, i.e., the copies do not have the full set of information and therefore can only be passive "backups". So the question is whether DOIP would help us in creating full copies as is indicated in the diagram. The replication client would have two similar functions: one to read all relevant information from the original Fedora instance and the other to write the same type of information to the copy instance. The DOIP is used to submit the requests and the Fedora Reader/Writer operations are straightforward to implement operations to support the replication. This solution could easily be ported to any other Fedora based repository and even more special reader/writer functions would allow us to support other repository software packages such as for example D-Space.
* The gain would be to have registered operations (reader, writer) which could be started on demand.

**Is the Bundle concept compliant with the RDA collection proposal?**

The bundle expresses specific relationships combined with an efficient solution for management and analytics. This special relationship could mean "sharing the same time axis" such as in parallel recordings of video, audio, gestures, eye movements, multidimensional annotations of time series, etc. which researcher want to visualize or analyse as a whole. Originally it was designed as a simple collection where every object (data and metadata) was represented the same way. In Fedora, which is used now, metadata has a special treatment.

The bundle has a metadata description encoded in XML and is expanded by special metadata describing the different time series added to the bundle. All entities have PIDs, i.e. the metadata description and the various streams are DOs. In Fedora the series metadata are also individual entities having PIDs.

**PID1**

**PID**

**bundle metadata**

**series1 data**

**series1 metadata**

**PID2**

**series2 data**

**series2 metadata**

**etc**

The bundle concept emerged in 2001/2, i.e., there was no RDA yet. The solution taken in the recent Fedora implementation comes close to what is required by the RDA DFT and the RDA Collection WG requirement. Every stream in the bundle is a DO and thus has separate PIDs and Metadata associated with it. The bundle is simply a collection with its own PID, metadata description and obviously pointers to the various series. To support operation with specific software "knowing about the bundle structure", this specific collection should have a specific type.

**Should we think of a generic Fedora/D-Space DOIP package and would it help in the case of DOBES?**

Most people use Fedora in the standard way, i.e., without a special relationship type as in the DOBES case. Every DO - be it atomic or a collection - could be organised the same way with explicit relationships documented in metadata. In this case it would make sense to use DOIP as a universal interface to Fedora, D-Space and many other repository packages. In each case one would have to translate the basic operations (CRUD) to meet the needs of the packages. Basis must be that each DO has a bit sequence, metadata descriptions associated with it and a PID.

In the case that only one of these packages is used there is no need to use DOIP of course, since each of these packages has upload services.

In the DOBES case a special Fedora package would not help so much.

**The Gatekeeper in DOBES executes a number of steps which will be repeated (partly) in other repositories? Can we think about a more generic workflow style?**

In fact, the Doorkeeper function executes a number of steps that will be used by many repositories in different style and sequence. It would be of interest to create a formal language to describe the actions in a declarative way and then have a generic workflow engine executing the commands. In the following table we list the operations carried out by the gatekeeper and specify some typical commands (here specified in an adhoc language which to make it working would have to be more complex).

|  |  |
| --- | --- |
| step in NL | semi-formal command for WF engine |
| upload each entity in the submission package dependent on package | for\_each entity (subm\_list) store bit sequence, return (path\_list) |
| validating CMDI metadata | for\_each md-entity (sub\_list) check CMDI, return error, take action according to options |
| validating data formats using FITS | for\_each non-md-entity (sub\_list) check FITS, return error, take action according to options |
| compute MD5 checksums for all | for\_each entity (sub\_list) create MD5, return (md5-list) |
| create the Handle Records | for\_each entity (sub\_list) create Handle\_Record (h\_record\_list) according to profile |
| minting Handles for all entities | for\_each entity (sub\_list) request a Handle providing (h\_record\_list), return (h\_list) |
| add Handle to the metadata | for\_each entity (sub\_list) update metadata with (h\_list) |
| generating default ACL access policies[[1]](#footnote-1) | for\_each entity (sub\_list) create an ACL using (default-acl-list) |
| create special metadata descriptions for all objects (in cas of Fedora for example FOXML) | for\_each entity (sub\_list) create an ACL using (default\_acl\_list) |
| trigger generating copies of all DOs | for\_each entity (sub\_list) get updated versions and create copiees using (destination\_list) |
| triggering the indexing for the internal search | for\_each md-entity (sub\_list) update the search\_engine\_index |
| generating DC | for\_each md-entity (sub\_list) create dublin-core metadata (dc\_list) |
| offer for the external search via OAI-PMH | for\_each md-entity (dc\_list) add it to the OAI-PMH portal |

Given such a declarative description of the actions to be taken, one could invoke the workflow engine by specifying a specific operation via DOIP.

**How could this archive take profit from using DOIP?**

An answer can best be discussed in the realm of the EUDAT discussion below. If one just looks at the DOBES repository as a single entity nothing needs to be changed.

# E-RPID Case (some comments from Robert and team)

The main questions addressed were:

* DOIP is not used to access the different repositories that store the data, but to build, manage etc. a metadata store that is being used as a mediator to access the data. Is that a typical case for DOIP?
* The CORDRA based metadata store associates PIDs with structural elements in the data streams stored in the repositories. This is a function often being used by efficient databases. Does this solution scale? Which additional feature does DOIP give?
* When would be a reusable package become available? Would it make sense to distribute a dockker image?

# Generic Issues

A number of generic comments about the DO concept and DOIP were made independent of the two cases presented. We start with those that are the most concrete.

**SWORD as an Alternative**

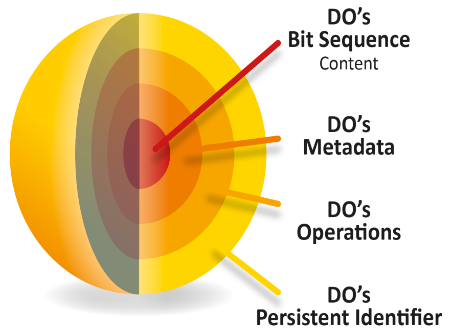
The SWORD (Simple Web service Offering Repository Deposit) protocol was raised as a possible alternative to DOIP. SWORD indeed is an interesting piece of work to create a generic "deposit API" to any repository system based on the ATOM publishing protocol. Its function was on purpose restricted to the deposit of (collections of) files although its first requirements were wider. Implementations for Fedora, D-Space and others were deployed. The D-Space implementation is using METS containers with a METS manifest containing some metadata. In the Fedora implementation METS containers are transformed into Fedora's object/stream architecture. SWORD is using the HTTP protocol for the interaction between SWORD compliant clients and repository ingest services. Version 2.0 of SWORD not only allowed ingests but the usual CRUD operations. It should be noted that Atom, the underlying protocol, is specified by RFCs ([RFC 4287](https://tools.ietf.org/html/rfc4287), [RFC 5023](https://tools.ietf.org/html/rfc5023)).

SWORD is thus not a generic protocol to interact with Digital Objects as DOIP is meant to be, it does not rely rigorously on the use of globally resolvable PIDs and it is not known despite its interesting features.

**EUDAT Case**

EUDAT is a distributed European Data Infrastructure project that incorporated 5 scientific domains at the beginning which was later extended stepwise and some large national data centres. One of the tasks was to create replicas of data hosted in various domain repositories on these large centres and to do this in a way which would currently be called FAIR-compliant, i.e., PIDs should be registered and metadata should be copied as well and further be linked with the corresponding data. This task demonstrated the principle problems we currently have with many data organisation implementations.

For each replication case a separate software stack had to be developed and would have to be maintained. We cannot report on all cases here but want to mention two typical cases with highly valuable data. Data of about 10 TB was stored partly in file systems and partly in a content management system. Metadata was included partly in the CMS, in embedded HTML documents and partly in scripts. In another case data from experiments and some metadata were embedded in a large database management system the content of which was changing.

This meant that with one exception it was not possible to issue simple requests such as “return the metadata for specific data” or “return the data or metadata that belong to a given PID”. EUDAT colleagues were requesting the use of a unified protocol such as REST over HTTP, but this did not solve the problems of course, since neither HTTP nor REST have knowledge about the relationships between entities such as expressed in the FAIR-DO model (see figure): The FAIR-DO model clearly specifies that given a PID one can easily access all relevant entities of a DO. Resolving the PID and then using DOIP does the job. In case of non-CRUD operations, the PID would also help launching suitable operations.

**Is DOIP meant as replacement for SOAP?**

As indicated above in the EUDAT example, operating with FAIR-compliant data means to have “knowledge” about the relationships between the informational entities of a FAIR Digital Object. SOAP may do its job to interface between web-services, but it does not have knowledge about DOs and the relationship between data, metadata of different types and PIDs. Therefore, DOIP goes far beyond SOAP and is meant to maintain these relationships.

On the other hand, one can argue that DOIP could be tunnelled via SOAP or any other protocol linking clients and services.

**One can manage repositories using HTTP and thus don’t need DOIP.**

Sure, one can use HTTP for managing repositories. But this argument ignores that one needs some logic on top of HTTP to maintain the relationships as they are requested by the FAIR principles and as they are implemented by FAIR-DOs. This argument states that the colleague did not yet understand that we need to change our data practices at the end to overcome the 80% waste of time due to data wrangling. We need to turn to protocols and methods that ensure the consistency of the emerging data space over many decades. An ad-hoc protocol that allows to do basic exchanges is not enough.

**Mapping or wrapping to other protocols?**

This is an IT and implementation question and does not touch the essentials of FAIR-DOs and related protocols such as DOIP. It needs to be checked whether SOAP, REST, TLS or even another evolving protocol is most efficient to tunnel DOIP.

**Can we combine this with Blockchains?**

Industry is highly interested to use blockchain technology to store transactions safely. Recent discussions an IoT-Meetings revealed that the combination of FAIR-DOs with globally resolvable PIDs on the one hand and blockchains on the other could indeed allow them to build trustworthy systems. Clearly identified DOs would be linked persistently with a blockchain, thus at all moments in time people could check the state of transactions for that object. It’s the systematic implementation of FAIR-DOs that makes the difference, since the PID record simply would need to be extended by another typed attribute containing the link to a blockchain, which is in IT terms just another type of metadata about the data.

**The type/operation combination is appealing, but it’s new and there are many open questions.**

**How many types do we expect – how can we reduce the number of types, how to prevent type proliferation?**

The first statement is in so far true that the application of the concept of Abstract Data Types to data, or more precise to Digital Objects, is new, raising many questions such as the one raised. It will certainly be a community challenge to prevent a proliferation of types. But this should not prevent us from using this possibility. The young history of software engineering as clearly shown how important encapsulation is to manage complexity. There is no doubt that the evolving data landscape will be enormously complex. Various actors will contribute to a Tower of Assertions and Derived Data on top of increasing amounts of raw data which together will form our digital memory. We need to admit that we are currently at the very beginning of this development where we are still allowed to play and test without risking the Tower of Babel situation. But we need to urgently change.

**How many APIs do centres need to support – is DOIP just another one?**

This question is understandable, but we cannot give an answer. Before TCP/IP was chosen as the common standard, the big computer companies came with their own network protocols. Currently; many different software stacks are being used to manage data and different kinds of metadata with the result that about 80% of our time is wasted. A change of data practices is required to reduce this substantially, therefore testing new protocols and ways is a necessity.

**Yet another API - when there are no success stories and no packages there will be no uptake?**

**Who will create the necessary adapters – the repositories can’t do it, since they lack money?**

It is indeed obvious that there need to come projects with building adapters as core. These should attempt to build a set of typical adaptation packages (Fedora, D-Space, a few others) which then could be taken as reference examples for building modified versions were necessary. Repositories indeed cannot take the effort. It is obvious that “running code” will be essential to make progress. Some projects including DOIP are being started, but more needs to be done.

**Do we get really an advantage from DOIP operations? Imagine 1000 service endpoints and then a change of service location, in case of DOIP one would have to update 1000 service endpoints.**

Each service is identified ideally with one PID and an entry in the DTR. In case that a service endpoint would change, it would require to change the DTR entry which is not a difficult task.

**We seem to re-invent the Semantic Web without Web or without Semantic?**

The semantic web is addressing different aspects. It was never meant to manage large data spaces. It was meant to deal with the semantic complexity. Therefore, the many excellent results of the semantic web will survive independent of how the data management will be solved.

# FAIR-DO Aspect

We need to understand that when talking about data management the FAIR principles will be foundational. They are now globally shared independent of the fact that a few dimensions such as data quality are not explicitly mentioned. The FAIR principles will open the way towards globally improving data practices and to overcome the huge inefficiencies. Therefore, it is important to argue from the point of FAIR-DOs as a way to implement them. There are many other ways as there were many ways to implement computer networking. The RDA DFT model summarised that model based on many case studies and the global resolution of PIDs, the standardisation of PID attributes and DOIP are core elements of the model.

If we treat DOIP just as another isolated protocol, we will miss the point of referring to the relationships between the various informational entities of a digital object that are crucial for all sorts of processing and for building a stable domain of data.

1. Here we assume that final access permissions are being generated by authorized persons via some user interface. [↑](#footnote-ref-1)