RDA Data Foundation and Terminology DFT 2: Analysis & Synthesis

Technical Editors: Peter Wittenburg, Gary Berg-Cross, Raphael Ritz
Copy Editor Karen Green (UNC)

July 2015

Version 1.5

When comparing all models listed in part 1 we see two major lines: a number of descriptions are about ***data organizations***[1] (describing a model) while others focus more on the processing of data according to certain ***workflows***. This two track approach is a standard way of modeling information systems. In the latter case data objects and their relations are described as elements in the data process continuum, while in the model descriptions the embedding of data objects in work and data flows is not in focus. In this way the two views are complementary and we assume that the workflow aspects will be taken up within the Data Fabric IG.

The general outline of documents from DFT WG is as follows:

- DFT 1: Overview
- **DFT 2: Analysis & Synthesis**
- DFT 3: Term Snapshot
- DFT 4: Use Cases
- DFT 5: Term Tool Description

In addition DFT offers a wiki tool for interacting views about terms.

In Chapter 1 we will focus on the organization of data with some attention on data identity as proposed for data objects. In 1.1 we will elaborate on details and in 1.2 we will summarize these details. Chapter 2 will present a synthesis. In Appendix A we list of all relevant terms that appear in this document.

1. Analysis of Data Models

1.1 Details

OAIS

The OAIS model (1.18[2]) is the blueprint for all data archives although it lacks some of the specificity which we might need for RDA. At least we can make use of part of its conceptualization of 2 types of packed data object - Submission Information Packages (SIP) and Archival Information Packages (AIP). Supporting the packages are associated terminology such as “ingest” for submission, “data management”, “archival storage” for AIPs, “access” and “preservation planning”. Most if not all

[1] We call “data organization” all aspects that have to do with how to define data objects, how to embed data objects in their context, how to register data objects, how to describe properties of data objects so that they can be checked on integrity, easily found, interpreted, re-used and preserved for a long time, how to manage relations between objects, how they are organized into collections, etc.

[2] Numbers such as 1.18 refer to the model section in the Model Overview document.

repository models described fit with the OAIS model. The OAIS model is about “archiving” explicitly, but some do not make a difference between a repository for archiving and processing purposes. However this paper does not address this aspect either.

Kahn Model and variants

The basic model which we can find in many descriptions is a modified or augmented version of Bob Kahn’s model described in the sketch 1.1.1.

1. A metadata[3] search will result in PIDs[4] to be found in the metadata description which will be offered to a resolution service.
2. The resolution service will respond with information extracted from the PID record[5] which we like to call "state information" as a current specific state of the referred object as documented by its metadata. This record may just be empty (no attributes mentioned in 1.10), or it may contain a number of attributes as indicated in 1.1.3 (Wittenburg), 1.2 (Kahn&Wilensky), 1.3 (DFT Concept), 1.5 (CLARIN), 1.6 (EPOS), 1.7 (ENES), 1.9 (EUDAT) and 1.17 (DataCite/EPIC). Model 1.1.2 (Chen) is compliant with this model since it speaks about the binding role of the PID record - it brings together a number of essential attributes and in this way might be considered an extension of the Kahn model. Also the other models mentioned which are compliant speak about a variety of attributes stored in the PID record. Since 1.10 (DataONE) does not store attributes with the PID they introduced “system metadata” which externally stores typical information others store in the PID record. System metadata can be distinguished from “scientific metadata” since the latter is provided by the depositor and thus should not be touched by repository people.
3. The PID record will be resolved amongst others to yield information about the locations of instances of the digital object’s bitstream in some repositories, to information allowing checking its integrity etc. and finally the selected repository which will deliver the bitstream encoding the content the consumer is looking for. The PID Information Type WG will elaborate on the information types involved in checking integrity etc.

As indicated by 1.1.2 (Chen), 1.1.3 (Wittenburg) and 1.7 (ENES) the PID record also can have information allowing location of the corresponding metadata description, a landing page, the original repository storing the DO, etc., thus the PID record has a kind of important binding role as Yin Chen expressed it.

Pfeiffenberger Model

Hans Pfeiffenberger addresses the problem that “One Work (a specific meaningful content)” (or what has been called a Research Object) may be stored in different repositories and in different digital representations (bitstreams[6]). An example might be 2 representations of an image, one in JPEG and another in TIFF. All these versions may have different PIDs or share some despite that the identities are different (different bitstreams). This does not deviate from Kahn’s model, however, it

[3] In the DFT WG we will not discuss definitive details of metadata since this is the task of MD IG, but some preliminary discussion is appropriate and it is agreed that metadata attributes include typing the digital object it is associated with so that this information becomes available via search.

[4] PIDs could be found also in other environments such as in caches, as references in ePublications, etc.

[5] It is agreed that PIDs are associated with attributes that contain state information (of the DO such as fingerprint and the instances of its bitstream such as locations) of the digital object referred to. The term "record" in this document does not imply a specific type of implementation but some logical record that holds the combined information.

[6] The term "bitstream" is used here as synonym for "bit sequence" which in some cases sounds more appropriate.

raises the issue of the choices needed to clearly document relationships by the project using repositories or any other means of preserving data with relations:

- Obviously people can assign two different PIDs pointing to different instances at different repositories (independent whether this is the same bitstream or not). Using the same PID for two different bitstreams does not seem to be optimal, but in many cases it will be reality since it will be quite some effort to maintain the relations between different versions across different repositories. The ways "works" may end up in different repositories cannot be controlled but it may be guided by best practice suggestions.
- If a PID record is used to include information allowing users to check integrity, for example, then such a scheme only makes sense when different representation versions (different bitstreams with checkable integrity) have different PIDs.

So the solution Pfeiffenberger described is that there needs to be user-generated documentation that describes how related data objects with different PIDs are used and related back to an original PID for the source data. There will always be a challenge to maintain all relations between different versions of the same work (temporal versions, representation versions) or between collections and constituents over time. As for other issues it is important for the research community to discuss and decide on how to handle this.

EPOS Description

The EPOS system can be seen as compliant with what we call Kahn-related data models when primary data objects have reached completeness. That is associating metadata and PIDs both having some attributes that store properties of the object is what EPOS descriptions are aiming at. One of the problems is how the community should deal with incomplete or dynamic data that is necessarily used at very early moments after creation and how the community can refer back to these incomplete versions that researchers have used. Again the basic data model organized around data PIDs can function but does not, by itself solve the problem. However, it is important for the community to decide about policies of how and when to assign for example PIDs and associate state information, and how to create and update metadata descriptions.

It should be noted that many communities will have to cope with dynamic data where modifications of bit sequences are not controlled by human actions but just by asynchronous uncontrolled events. These events can produce "gappy" streams of sensor-based data results and the gaps being filled at unpredictable times. Likewise gappy databases are created by crowd sourcing activities where no one knows when human actors will participate in a certain test and when cells in a database are being filled. One practice for this is to have the repository assigning metadata and PIDs execute the strategy and this strategy needs to be made explicit so that users know what the policies are. It is a joint task of the Citation of Dynamic Data WG and the Practical Policy WG to discuss how referring to dynamic data should best be done.

Aggregations and Collections

Aggregations and collections of data objects are essential when discussing data organizations. In one form or another they are explicitly indicated by CLARIN (1.5), ENES (1.7), DataONE (1.10), ORE (1.13) and Fedora (1.14), but they are also of relevance for others. While the models that are influenced by the ideas of digital libraries and semantic web like to use a description style where explicit relations are mentioned, others are not specific about their relation types in their simplified diagrams, but basically these two traditions and their notations can be seen as expressing equivalent intentions.

For CLARIN a collection (i.e. they call a linguistically meaningful aggregation a collection[7]), is a distinct data object described by a metadata object and having its own PID, including selected attributes, so that it can be cited etc. The collection metadata description has attributes that describe all relevant collection properties. Of particular interest is that this metadata description refers to and relates collection components which can themselves recursively be collections. This allows the establishment of hierarchies[8] of metadata descriptions referring to individual digital objects or directly to their bitstreams. It is assumed that these components are each identified by a PID, i.e. the collection description may contain therefore just a list of PIDs.

For ENES "blobs" are aggregations that are identified by PIDs (in their case DOIs) and have a metadata description in a database. Also here the metadata includes a list of identifiers pointing to the individual objects making use of implicit relation types. DataONE is using the ORE model to express aggregations.

In all these models the references can point to anything considered a collection that can be resolved into the current state information of a collection by means of a PID resolver. In CLARIN, for example, use of PIDs means that collection object instances can be stored everywhere. ENES is currently using internal identifiers, so the individual objects must be local to the repository, but they are also changing their system so that for all DOs created during the simulation runs externally registered PIDs will be associated.

Digital Library (DL) and Web-influenced Models

These communities developed models and terminology over many years which are reflected in the ORE (1.13) and Fedora Object (1.14) models. The Europeana model for metadata (1.15) is based on the conceptualization of the semantic web. ResourceSynch is agnostic with respect to how data is organized. So it can be used to synch data that is, for example, organized by the above mentioned DL and Web models. Whatever is the data organization of a repository, and how relations for example are stored in the repository, data synchronization in this view needs to present serialized lists of objects to be exchanged via services such as web services.

The ORE model is meant to describe and exchange aggregations with the help of explicitly named, but light semantics (RDF assertions) contained in a Resource Map. Resource Maps also contain some properties, types and relations. Thus again ORE aggregations are very similar to the ones described in the previous chapters except that support of services makes some of the aggregation relations implicit. For example, a resource map that asserts existence of some aggregated resource X may also assert an ore:isDescribedBy relationship between that aggregated resource X and another resource map describing another aggregation.

The Fedora repository model supports XML structured containers "aggregating" a number of components, each container being identified by a PID and having a number of properties. Essential components are "datastreams[9]" which can encode any content. As indicated in Figure 24 of the Model Overview document Fedora containers can be used to bundle together different presentation/format versions that refer to the same "work", a metadata description and other related information such as disseminators. The container concept also is general and allows one to build other type of aggregations with relations indicating the types of these aggregations.

[7] The term "collection" is very common in social sciences, arts and humanities.

[8] In general these can also be connected graphs.

[9] The Fedora model is using the term "data stream". In this context we do not make a difference with the term used within DFT as "bitstream" in which data is encoded by bits.

As noted in Part 1 (Model Overview) the Fedora repository model bridges between the data object idea, services and abstract models by defining several types of data objects. The type being discussed in this section holds digital content, i.e. data. A second type defines "Service Objects" which store models/descriptions of services. A third type is Content Model Objects which defines a more abstract class of digital objects that are rich information networks. Such abstract digital models seem consistent with the challenges & solutions discussed previously as part of the Pfeiffenberger Model. In particular consideration of the relationship between associated publications on a research topic and various cited data is an important challenge involving complex relations. A small example previously mentioned is the fact that what humans may judge effectively same content such as graphic objects at different degrees or granularity or encoding (Jpeg or GIF) may appear in different formats at different locations and be cited by different sources. Likewise data may be encoded in spreadsheets or DBs.

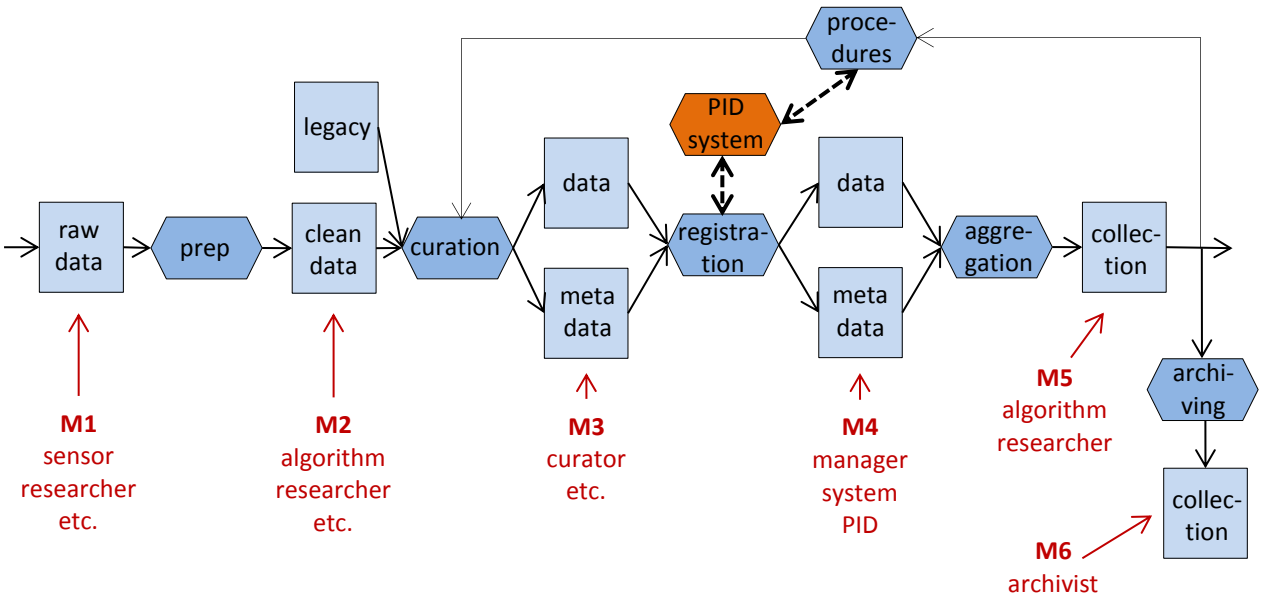
While identity of digital objects has been discussed in this document, identity of services may also be an important component to standardize. At the physical level of bitstreams without further representation-based interpretation there is no difference between data and software. It is the representation and typing information provided in the metadata descriptions (accessible via the PID records) that will help to interpret and process the bitstreams in the right way.

Metadata, PID Attributes, Catalogues

Most of the models speak about attributes included in PID records and metadata descriptions. However, there is much diversity in ideas, usage and terminology. This is not surprising since the role of metadata is largely driven by specific needs that may vary by research domain and over time. The primary intention of the concept of PID record is to give a digital object an externally registered identity which can be resolved into useful current state information such as access paths. This information becomes particularly value if it maintains its integrity over many years and thus answering the question "is this (still) the bitstream which a publication refers to", etc.? This integrity task is a central role for PIDs and may be achieved as part of different formulations such as the binding mechanism discussed by Yin Chen or others. Using external authorities to register data objects with associate information that allows checking integrity is a widely used concept for improving trust in our increasingly anonymous data world. The option to use the PID record to add other relevant information describing non-changing properties of data objects is interesting for many repositories. Yet there is no wide agreement on the type or minimum amount of information associated. A related issue is how tightly to bind such information and which information should be bound. Names and checksums are candidates of such minimal, essential information. The PID Information Type RDA WG is working on the issue of identifying a number of widely agreed core properties and to standardize the APIs. The Type Registry RDA WG is working on a solution to easily register such types.

The area of metadata is much broader since finally creators, managers, archivists and users (social tagging) want to describe the properties of digital objects from their specific point of view. Given that some identity and integrity of DOs can be proven by the PID record, there is still much information necessary to find, interpret and re-use them. This information can become rather detailed in case of workflow frameworks, for example, where the description needs to include all details for follow up processing components to function correctly. In this case of metadata being enriched during workflow execution experts speak about "provenance" information for example that contains information about the genesis and subsequent manifestations of a DO. Similarly repositories that do not use PID records for storing essential information like to speak about "system metadata" that is under the control of the repository managers and which may be close to what some people call administrative metadata. Archivists like to speak about "archiving metadata" that contains information relevant for long-term preservation. For some scientific domains there are, for

example, ISO standards that specify elements or even complete schemas. And of course we need to mention the Dublin Core work on general and specific elements which are widely used for retrieval. For most scientific purposes this is too static and lacks detail to solve all needs.



Thus we observe that there are many points of view without operative consensus on metadata components[10] and metadata elements and their names. Even for meta-models that may work with registered and somewhat standardized concepts and offer flexible syntactic frameworks there are no wide agreements. The semantic web community offers a flexible framework by allowing specifying assertions (triples) using RDF syntax. Formal semantics and be added by means of the OWL semantic language. Nevertheless the meanings are being defined by use of accepted elements. It is widely agreed also for those disciplines that are using XML schema-based approaches with implicit relations that an RDF export with explicit relations should be generated allowing assertions export according to principle of the linked open data framework.

Catalogues are systems that offer typical functions on aggregated metadata such as search, filter, etc. We can draw a generic process flow (see Model Overview 1.21 and diagram above) to investigate when and by whom metadata is being created or manipulated. At different stages metadata (i.e. M1-M6) metadata is being generated that is created by "authorized entities[11]". This can be humans in different official roles or software under various control. It is the task of the Metadata RDA WG to advance harmonization in the area of metadata.

Bitstream vs. Content

In many cases the models just speak about bitstreams that need to be identified, managed etc. On top of these bitstreams are encodings using scientific data formats. These are endless and continuously new ones will be added dependent on the scientific progress. This reflects the fact that bitstreams may play different roles based on their content. Consider, however, when a bitstream' content plays a metadata role. To some metadata descriptions at least from a digital object perspective may be well-described types in so far as they contain key-value pairs or in semantic web terms assertion statements on properties.

[10] It should be noted that some speak about "metadata" packages.

[11] Since metadata can be harvested and thus manipulated by everyone for his own purposes we need to make this difference.

In this view it makes sense to distinguish between internal and external properties of DOs. Internal properties are those that are required to be understood to interpret the content of a bitstream such as its structure and the semantics of the elements being used. Information about internal properties are relevant for the scientists, but not for other actors such as data managers. This is different for DOs that include metadata as part of the object. Here managers, archivists etc. are as well interested in interpreting the bitstream to build integrated catalogues with search functions etc. Integrity information such as checksums are of central interest to a data manager for example. Thus in the case of metadata it is required that we come to cross-discipline (data manager and domain scientist) harmonization that will include components, syntax of descriptions and semantics of categories for this common, "external" metadata.

Metadata descriptions such as representation information are thus required to interpret and re-use the bitstreams, thus they should include attributes describing the internal properties. How to maintain the references between bitstreams and metadata descriptions? With respect to this aspect of data organization we see some differences:

- All models that are variants of the Kahn model require that the PID record points to the data as well as to the metadata description or even to html landing pages which may also include prose information relevant for proper re-use. In addition the metadata description should include a pointer to the PID record of the DO. This is indicated for example in the CLARIN model.
- DataONE uses the ORE aggregation to indicate this close relation between data and (scientific) metadata. Opening a Resource Map would provide links to both object types. In addition they maintain system metadata which also point to the metadata description. Since data and metadata descriptions are objects there will be two different system metadata entries.
- The Fedora object model is similar to the ORE aggregation in so far as it allows one to create containers that bundle a number of data streams such as metadata, presentation versions etc. No repository is obliged to use the container option for this purpose, but then more complicated relation structures need to be maintained.

1.2 Summary

1.2.1 Elaboration on Kahn's Model

If we take Kahn's sketch model as the basis, since it simply describes some independent components and APIs. Then we can indicate the extensions, elaborations or differences by the different contributions:

- Chen elaborates on the PID record and its binding function.
- Pfeiffenberger indicates that there can be different repositories holding varieties of "presentation versions" all being identified with PIDs. There is currently no single solution how relations can be maintained between them, how proper metadata descriptions should indicate provenance etc. So for a proper data organization the basic model described by Kahn is not sufficient.
- Wittenburg adds how the triple between PID record, metadata description and bitstreams is related. His approach may be used to start handling the Pfeiffenberger problem of networked DOs.
- Kahn&Wilensky elaborate on the process in which data objects are being registered and which kind of properties are relevant to maintain.
- The DFT Concept Note basically repeats Wittenburg's core ideas and embeds it in Lannom's canonical (see figure 12 in Model Overview Document) access model where the PID is put central for all access operations – and is thus compliant with the Kahn model.

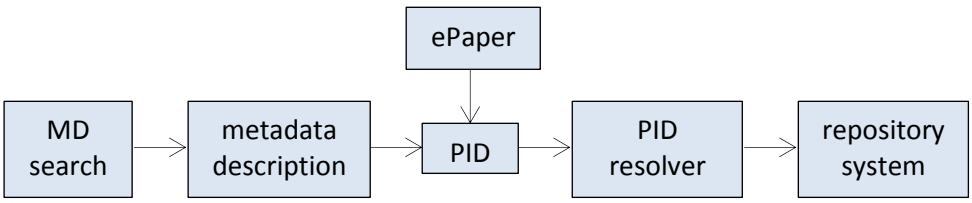
- The CLARIN model uses the same conceptualization and adds a way how collections are being organized.
- The EPOS model also uses the same conceptualization, but focusses on the question how one should deal with dynamic data and refer to it.
- The ENES model also adds to the binding idea of Chen in so far that they see the bundling between data object, information object, metadata object and transaction record as essential.
- EUDAT has to do with many types of communities with highly varying data organization landscapes between and also within discipline communities. Some are using the Fedora repository system as container for metadata and relations, nevertheless registering external PIDs to identify the containers.
- DataONE uses internally generated system metadata including PIDs that refer to the data and metadata objects. Some of the attributes included by various others in the PID record are here integrated with system metadata. It's the Resource Map that establishes the relation between the data object and the corresponding metadata object.
- Earth Science System Data is maintaining a database system that binds data objects and its metadata descriptions by some logical structure and it does not speak about PID registered externally. So it's an internal solution.
- The Clinical Genomics Workflow mainly speaks about automated workflows where at the end of the workflow variant sequences are put together and metadata descriptions are being managed together with the data. But both aspects are bundled together in a specific file format. So the workflow does not speak about separate metadata that could be harvested and aggregated by others and it does not speak about external PIDs.
- The ORE model is used for aggregating any types of data objects and uses ORE relation types to refer to its components, to properties such as a possible external PID, to provenance information and more. In this way ORE defined relations seems to allow us to model bundles and aggregations as they have been described by others. PIDs are properties of aggregations and the attributes that some would associate with PIDs in a PID record could also be modeled as properties. URIs are used to refer to all entities.
- The Fedora model basically is a container format to bundle or aggregate related data streams. It allows users, for example, to bundle version data streams and metadata descriptions, associate a variety of properties with the bundle such as an external PID and include relations. When being used in this way it is similar to other models except that having resolved a PID to the access path of the container it needs to be translated by the Fedora software to access related things like the metadata description.
- The Europeana model does not touch the Kahn model since it is about how to structure and represent metadata in a way that allows optimizing the gateway building between diverse metadata descriptions.
- The MediaWiki model does not fit with the Kahn model, since no layer of indirection is required.
- The DataCite/EPIC models are agnostic with respect to the underlying data organization, since they simply allow users/repositories to register PIDs/DOIs and resolve them to paths and other attributes.
- The cloud model introduced in 1.20 of the Overview as result of a split between physical and logical layer information is transparent to the discussion about data organizations. In the case of clouds the external PID needs to be resolved into a repository address and a hash code which can be used by the cloud system to retrieve the correct bitstream.
- The Reference Process Model is very much compliant with Kahn's model and just adds information about different phases of data.

1.2.2 Access Methods

We can basically identify 2 methods to access a bitstream that encodes the information we are looking for:

1. A user (or algorithm) has found a meaningful PID, resolves it to useful state information amongst which is access path information and requests the bitstream from a repository by making use of the access information. The advantage of this method is that the PID can be seen as an immediate handle since it identifies the DO and will offer all kinds of useful information. The disadvantage is that humans cannot work with such numbers.
2. A user (or algorithm) finds a metadata description of a useful digital object or collection, extracts the PID from the description and then step 1 is carry out. The advantage is that the domain of metadata descriptions is understandable for humans. Disadvantage is that the access is not given immediately to the bitstream, but that an intermediate step needs to be done.

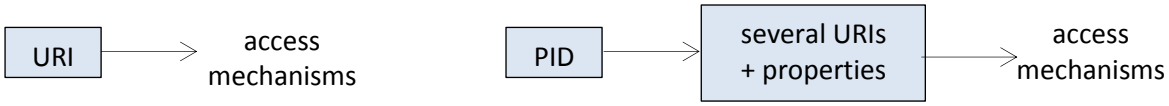
This very much resembles, for example, the Internet situation where a node is uniquely being specified by an IP number and if you have the IP number you can immediately access the node. But humans can only operate smoothly in the domain of "names" that then need to be resolved by the DNS system to IP numbers. In the world of data we need to accept this dualism and systematically apply it.



It is now up to the repository system how they want to give access to information. In the case of a Fedora Commons-based system the access path must specify which container and which "data stream" in the container is meant. It could even happen that content negotiation should take place, i.e. dependent on the user device and network capacity a specific representation version is requested.

1.2.3 Referencing

We can identify two major approaches to conceptualization and terminology building: one is close to the Kahn model although partly being derived independently, while others are inspired by semantic web ideas. We can also see that those who deal with “big” experimental/observational data are in general closer to the Kahn considerations, while the web conceptualization originates from the huge amount of web information. These differences will disappear over time, however we can see that the traditions of these communities still influence conceptualization, terminology, modeling frameworks, etc. To help overcoming these differences we need to accept them first and then bridge where possible.



One issue where these communities differ is in the choice of identifiers. The “web community” (WC) relies on URIs as unique and persistent identifiers which are (so far) straight forward, since the web standards and protocols that know how to deal with URIs are being defined and widely implemented and used. The “data community” (DC) relies on an explicit registration of identifiers for data objects at an accepted registration authority. This choice requires an extra level of indirection which resolves identifiers into paths knowing that there will be many instances of DO's bitstreams at different

locations and willing to add some properties to proof identity and integrity for example like in passports.

This is not the place to argue for one or the other approach; we need to understand the benefits of both and perhaps how to bridge them such as with approaches that add hash values to URIs to allow trust building (Kuhn & Dumontier 2014). However, in the data domain we need to have access to a variety of property information for different types of operations. If no PID record for binding and state information is being used, other mechanisms need to be invented.

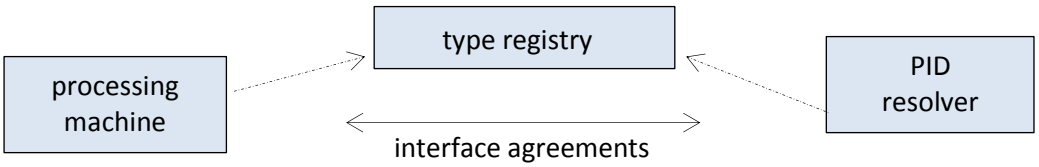
1.2.4 Types and Relations

It is now widely agreed that data management, access and processing should widely be guided by automatic procedures – documented and self-documenting practical policies turned into executable code. In the growing data domain with its increasing complexity only "machine-type" of routines will help us to keep control of what we are doing. Therefore in our considerations about proper data organizations we need to ensure that our machines find their way[12].

Given the schematic access diagram in 1.2.2 where a machine is receiving a PID we can state the following:

- The machine needs to know how the PID can be resolved, i.e. the PID needs to be actionable.
- The machine wants to know the closest location where the bitstream is stored and wants to check whether the bitstream is still the same trusting the information registered with the PID; therefore it sends a request to the resolver specifying the PID and the Information Types it wants to receive.
- The resolver sends the requested information types.
- The machine uses the access path information to contact the repository and also sends some user credentials and the checksum information.
- The repository will check whether it can resolve the access path, whether the user is allowed to access the stored bitstream and whether the checksum is ok. If all is ok, it will send the requested bitstream.

The interaction between the machine and the PID resolver for example will only work, if both are based on using the same types, i.e. if the machine sends "cksm" all resolvers need to know that the checksum attribute is meant. It is completely irrelevant how the resolver system is implemented (whether the values are stored in form of RDF triples or in a relational database system in tables etc.), but it must return the cksm. This can of course be achieved if we start including now such attribute names in a "type registry" so that all software developers can rely on the usage of these types.



In some model diagrams relational type registry semantics is made explicit, in some others not. In general it is absolutely necessary to make relational semantics explicit and formal so as to be machine processable. In the case of an ORE container machines can find information that is being requested. In case of for example a PID record we rely on that the underlying semantics of all

[12] It should be noted that many of these observations relate to the topic of the RDA PIT and DTR WGs.

attributes being very simple in so far as it states that "the DO identified by the PID has properties that are contained in typed attributes of the PID record", thus there is no need to express this in the diagrams.

1.2.5 Very Large Data Sets

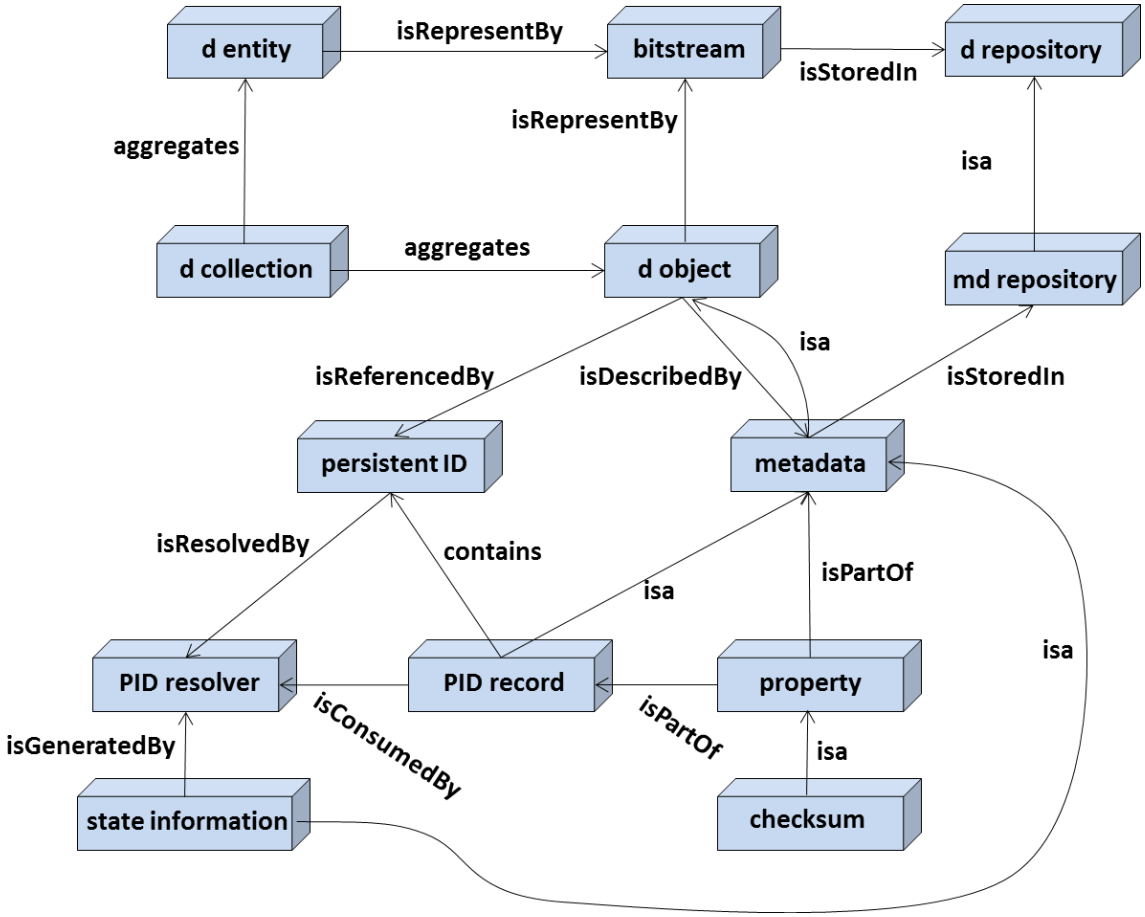
We should notice that for very large data sets and collections algorithms need to operate as efficiently as possible, i.e. all forms of indirections need to be avoided. This is the reason why scalable solutions prefer to work with fast databases containing only few metadata and making use of fast indexes and store their data in file systems or increasingly more often in object stores (clouds) and thus using their efficient access mechanisms.

Efficiency can be achieved in different ways. If a collection has been formed as a virtual aggregation of metadata of course it does not make sense to process all metadata files of a collection in real time. But from the collection definition procedures could for example generate a table with all direct access information before the processing is being started.

2. Synthesis

For a synthesis towards a common conceptualization we can draw a number of conclusions:

- In RDA we need to motivate our conceptualization on the basis of the models that have been suggested to overcome the current deficits in dealing with data.
- We only can deal with the domain of **registered data** that is ready to be accessed by others knowing that there is much data stored on notebooks and servers that may even be shared via traditional channels. However, it will be difficult to generate traceable results from such data as practice shows.
- There may always be special requirements in the area of very large data sets for example which may require special solutions. Nevertheless the methods need to be similar as indicated in the diagram below.



From the analysis we can draw a common model existing of essential concepts and their relationships which fits to most of the use cases as explained in the analysis section.

In the core is the concept of a digital object which

- is represented by[13] a bitstream (instantiations of which can be stored in several repositories);
- is referenced by a persistent ID;
- is described by metadata and
- can be aggregated to digital collections.

The digital collection itself is a digital object. Digital collections can include digital objects and digital entities. Often researchers create large digital aggregations which include digital entities that were not assigned a PID and do not have a proper metadata description, which however can be integrated easily in a resulting collection that is being assigned a PID and has a metadata description. Just think of a large set of photos made from an expedition or a large set of observations created by an experiment with a number of variables[14].

[13] In this diagram we have chosen this relation type to keep it the same for digital entity. With this type we want to indicate that researchers may have stored "metadata" in some form within the file header (if the digital entity is a file), as file type extension, in some spreadsheet etc. Such information allows the researcher to interpret the bitstream. We should note that this relation type "isRepresentedBy" is also being used in the OAIS model.

[14] It should be noted here that the issue of granularity of referencing needs to be decided by the research communities and is widely dependent on possible usages of the digital objects or entities.

Metadata descriptions contain attributes that describe properties of the digital object. Metadata descriptions are types of digital objects, i.e. they have bitstreams[15] that are stored in metadata repositories. However, to prevent recursion metadata objects to not have metadata descriptions which make them special digital objects. Metadata repositories are a type of digital repository. The distinction is made here due to the great relevance of metadata in the data community and the different type of treatment of metadata compared to data[16].

Persistent Identifiers are contained in PID records and resolved by PID resolution systems. PID resolution systems generate state information based on the content of the PID record which they consume. The PID record contains properties which are types of metadata but used for different purposes. One such property which is frequently mentioned is the checksum which is being used for identity and integrity checks.

3. Implications

The implication on a worldwide agreement on such a basic and simple model would be huge. It would widely unify data organizations and give clear guidelines to data practitioners and software developers. Repository system developers would have clear guidelines about how to organize a repository, application developers would know where to find PID information, how to interpret it, would find metadata for objects and collections and would be able to access and interpret digital objects. In addition we would be able to define a generic API for digital repositories to access the data it is storing.

As a recent survey based on about 120 interviews and interactions with data professionals has shown heterogeneity in data organizations is one of the largest reasons for inefficiencies and high costs when dealing with data.

4. References

Kuhn, Tobias, and Michel Dumontier. "Trusty URIs: Verifiable, Immutable, and Permanent Digital Artifacts for Linked Data." *The Semantic Web: Trends and Challenges*. Springer International Publishing, 2014. 395-410.

[15] This relation is not indicated in the diagram to not overload it.

[16] We do not want to define metadata in a more detailed way in the DFT documents, since it is known that "metadata is data" dependent on the view of the researcher, that are various types of metadata, etc. This discussion is left to the Metadata WGs and IGs.