RDA Data Foundation and Terminology DFT 1: Overview

Contributors: Stan Ahalt, Gary Berg-Cross, Jan Brase, Daan Broeder, Yin Chen, Antoine Isaac, Bob Kahn, Larry Lannom, Michael Lautenschlager, Reagan Moore, Alberto Michelini, Hans Pfeifenberger, Raphael Ritz, Ulrich Schwardmann, Herbert van de Sompel, Dieter van Uytvanck, Dave Vieglas, Peter Wittenburg, Yunqiang ZHU, Herman Stehouwer.

Technical Editors: Peter Wittenburg, Gary Berg-Cross
Copy Editor Karen Green (UNC)
December 2014

Version 1.6

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

1. Overview about Models

At the first Plenary meeting of the Research Data Alliance (RDA) Data Foundation and Terminology (DFT) working group, held in Gothenburg, Sweden, the DFT work group decided to collect data organization models that represent concrete use cases, i.e. models that are foundational to running systems or that specific communities especially associated with RDA are considering using as the basis of their data systems. We have so far incorporated the following models, listed here in order of submittance:

1. Gothenburg Sketches
   1.1. Kahn Sketch
   1.2. Chen Sketch
   1.3. Pfeiffenberger Sketch
   1.4. Wittenburg Sketch
2. Kahn/Wilensky 2006 Data Model
3. ResouceSynch Framework
4. DFT Concept Note Data Model
5. CLARIN Language Data Infrastructure
6. EPOS Earth Observation Infrastructure
7. World Climate Data Infrastructure
8. ENVRI Environmental Data Infrastructure
9. EUDAT Data Infrastructure Core Model
10. DataONE Data Model
11. Earth Science System Data (to come)

12. RENCI Clinical Genomics Workflow (to come)
13. Europeana Model
14. ORE Data Model
15. Fedora Object Model
16. MediaWiki Data Model
17. DataCite/EPIC Models
18. OAIS Model
19. DICE Data Model
20. Physical vs. Logical Layer Information
21. Data Processing Model
22. Chinese Earth Science Model

It is of course true that there are more data models in use and the DFT working group remains open and eager to include, discuss and integrate them in this and related documents as appropriate. It is important to note that this document focuses only on the above listed models and in particular "explicitly registered data." We recognize that there are so many other data organization and management solutions out there. The current work also reflects that fact that much temporary data is generated daily through the regular scientific process/workflow. Some of this will never become part of the sphere of managed, registered and thus readily accessible, re-usable citable data. For the purposes of the current DFT work this data is largely outside the scope of what has been attempted. In the Term Snapshot document we briefly discuss 3 examples of local data organizations indicating some of the difficulties to fully integrate such data into the sphere of registered data.

In fact a core of the models examined in this paper discuss Digital Objects (DO) that can easily be referenced by a PID (persistent identifier), which can be resolved into information that supports accessing them. No matter what local data repository system is used, software must be engaged to access data and generate the bitstreams encoding the content looked for. This software stack can include file systems or, for example, database query scripts, etc. Stability and correctness of the software stack applied is crucial. File systems need to rely on an operating system, such as UNIX, and, in the case of a relational database, on the correct functioning of the database management system (DBMS) and on the applications developed by different experts including local ones. In this paper, the challenges that have to do with different software stacks will not be addressed.

1.1 Model sketches at the Gothenburg Meeting

During the interactive session at the Gothenburg meeting, four different models were mapped out on the white board and discussed. These models were only sketches, rather than complete packages and are discussed below.

1.1.1 Kahn Sketch

This basic data model was presented and explained by Bob Kahn (CNRI).

In Kahn's model, the key component in a core data model is the PID (persistent identifier), which has a number of functions. In a canonical access workflow, users might first search for a specific digital object (or collection which is one form of a digital object) using a metadata search. The metadata description includes a PID plus additional attributes that may be used when accessing the object. The PID is submitted to a resolution system that returns location and possibly other information. Location information is used to access the digital object (DO) in some repository, an action that is transparent to the user. From the viewpoint of the user, the PID provides access to a DO. The content of the DO represented by a sequence of bits is then returned to the user. The PID can also be used to request information about the usage of a DO, such as who has received copies of it. The transaction system then returns a transaction record.

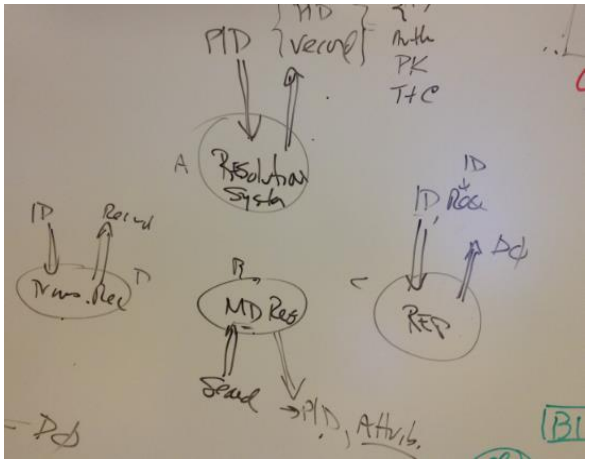


Figure 1: Kahn's sketch (white board)

The canonical access path in the Kahn Sketch is simple and straightforward:

metadata search -> PID -> access path -> DO's bitstream

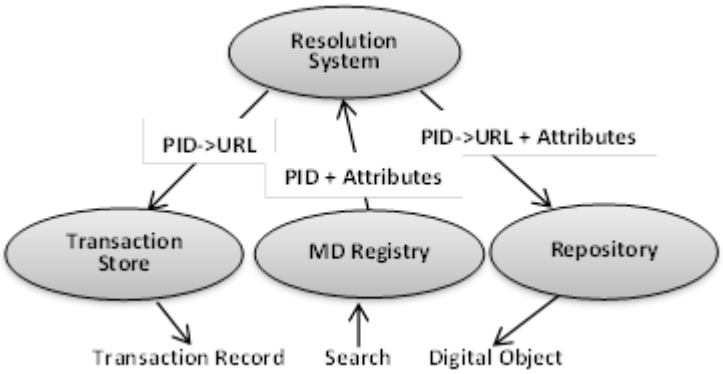


Figure 2: Kahn's sketch

Of course users can access DOs in other ways, using for example cached PIDs, using PIDs that appear in a document or in another DO, etc. In these cases, the user can directly resolve the PID via a resolver to get access to the bitstreams. PID records can also be used to store a variety of resource properties, such as checksums or Repository of Record (RoR) flags, allowing the user to immediately check integrity or to locate the original repository. Implied in this approach is that all these properties (e.g. checksum) are stored in a typed way so that machines can interpret them. That is, checksum is a defined data type whose definition can be stored in a data type registry as under development by the RDA Data Type Register Work Group.

In this view one major task of a data processing infrastructure is to specify the components (PID record, minimal metadata, etc.) and the APIs/protocols that guarantee smooth interaction between them. This model view was the basis of the Kahn/Wilensky paper (2005) and the DFT concept note (see 1.2 and 1.3).

1.1.2 Chen Sketch

A second basic data model was presented and explained by Yin Chen (U Cardiff) who developed the model for her PhD thesis.

This model regards the PID as a "binding object," which creates an association relationship between (a set of) metadata with its data object (or resource). A four-tuple binds together the following: (1) an *ID* as the binding identifier; (2) a reference pointing to the data; (3) a reference pointing to the metadata; and (4) a set of tags to describe other attributes or to annotate the semantics of the binding.

A generic binding service can be implemented, which provides various operations to support binding manipulation (for example, creation/update/deletion), discovery (of binding/metadata/data), communication (get metadata/data object), and validation (the consistency of the binding relationship).

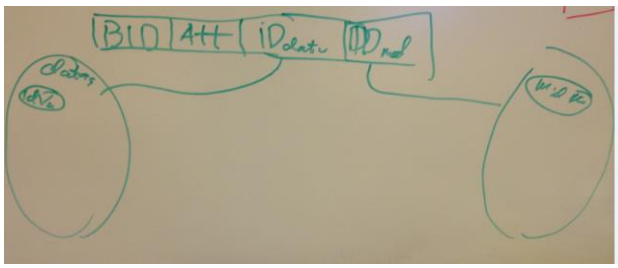


Figure 3: Yin Chen's sketch (white board)

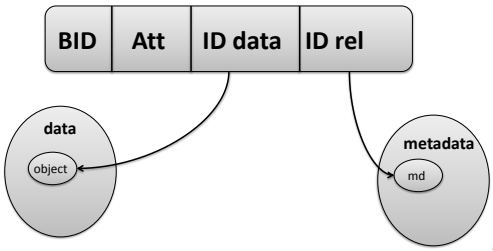


Figure 4: Yin Chen's sketch

Compared with the Kahn model, it is evident that the Chen model elaborates on the nature and context of the PID record, which is one of the essential components when accessing data. In particular it formalizes relations in a particular way. The four tuple could be a defined data type. Employing this data type would make for definable, useful, consistent services.

1.1.3 Pfeiffenberger Sketch

The third basic data model was presented and explained by Hans Pfeiffenberger (AWI Bremen).

Pfeiffenberger's model emerged from experiences in data publication with the data journal *Earth System Science Data*. Pfeiffenberger demonstrated the model using the example of the "Global Carbon Budget 1959-2011," which consists of data stored primarily in one Excel file with many sheets, at the Carbon Dioxide Information Analysis Center (CDIAC, Oak Ridge, TN, USA), a member of the former World Data Center System. Replicating this data into PANGAEA—an Earth and climate science repository in Germany—would require a human curation effort, since no automatic system exists. The data would end up as part of a huge relational database management system, which provides output formats in CSV and HTML. Both repositories (CDIAC, PANGAEA) maintain persistent identifiers (PID)[1] for their holdings, acquired from different PID registration authorities.

This example makes clear that whatever model the DFT working group decides to adopt, that model must be able to handle specific relations such as versions or representations of the digital object. In both given examples, the checksums are of course different, a fact that illustrates the need for different methods for proving identity, or the ability to prove identity at different levels. (The metadata at both repositories are different as well).

One conclusion to be drawn from this example is that the abstract data object has its own identity and should have its own PID and checksum and descriptive metadata. From an identity point of view

[1] We will use the term PID throughout this document. It should be noted that Pfeiffenberger and his colleagues are using DOIs (Digital Object Identifiers) which are one specific implementation of a PID.

the PID could point to a metadata record containing both PIDs using their relations that point to the different representations stored in different repositories. IDs in the record would then resolve to related versions of the data, e.g. the Excel and RDB versions of the data mentioned in the Pfeiffenberger sketch example.

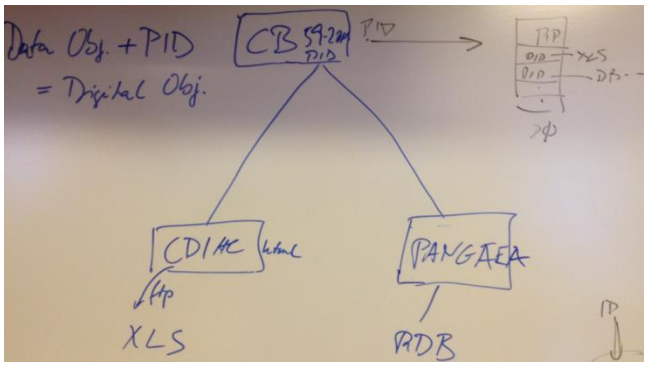


Figure 5: Pfeiffenberger's sketch (white board)

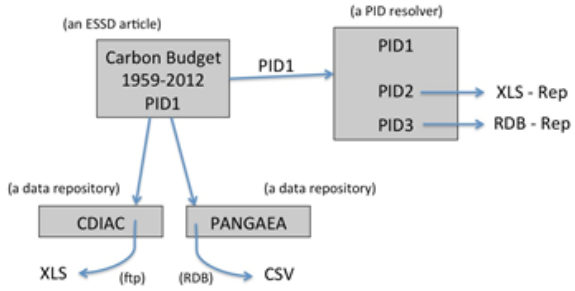


Figure 6: Pfeiffenberger's sketch

Discussion with RDA metadata interest and work groups suggested a key feature to that metadata would be to describe relationships such as touched on in this sketch. Such relations would include those between dataset and various organization (e.g. owner, publisher, rights holder, provider, creator, maintainer etc.) but also between dataset and scholarly publication (based on citation), between dataset and dataset (e.g. a relation of derived from or summarizing which is provenance metadata), between dataset and infrastructure (processed by relation), between dataset and person (creator, editor, user, reviewer…) and so forth. A tentative conclusion is that 'flat' metadata (such as the original Dublin Core -DC) is not adequate and a new standard is needed.

Different approaches to organize the various flavors of metadata to handle operations like data mutation and versioning are used. Further work to enable automated processing of data/metadata and digital objects and the relations between them are needed.

1.1.4 Wittenburg Sketch

The fourth basic data model was presented and explained by Peter Wittenburg (MPI Psycholinguistics).

Wittenburg's model is very similar to the one presented by Kahn. In addition, this model was used to briefly discuss what happens in workflows where enrichments are done.

The Wittenburg model sketch presents a triple of information that needs to be maintained:

- A PID record that points to a metadata description, to instantiations of identical bitstreams and includes additional property information;
- A metadata description that stores a variety of properties about the object, its context and provenance information; and

- A number of instantiations (copies) of the bit sequence encoding the content, which can be hosted in different repositories.

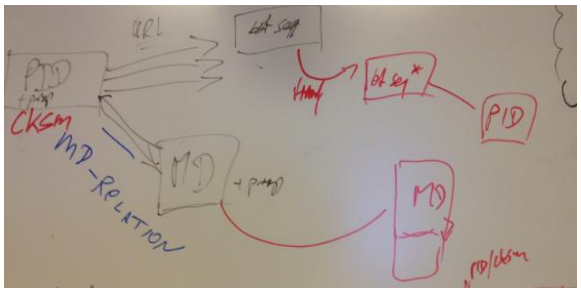


Figure 7: Wittenburg's sketch (white board)

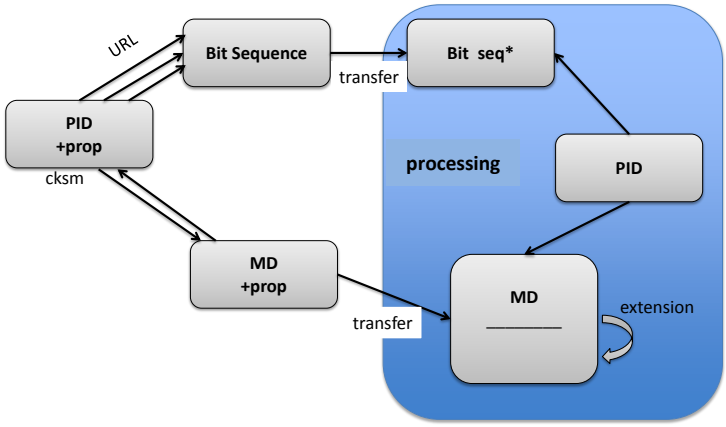


Figure 8: Wittenburg's sketch

For replication operations, the checksum information stored in the PID record is of great relevance. In cases when workflows are used that may change the bit sequence, a new object is created, which requires a new PID to be registered and a new or revised metadata description to be created—again forming a triple. This process is seen as one part of the blueprint vision for the emerging data fabric governed by automatic workflows.

This solution is very much consistent with Kahn's solution and in conceptualizing the idea of a PID record also makes use of something like the binding role as discussed by Chen.

1.2 Kahn/Wilensky Model

In 2006, Kahn & Wilensky wrote an article that references an earlier paper[2] in which they describe the basics of a digital object model. In essence, it is a more fully developed version of what Kahn described in his presentation at the Gothenburg meeting (see 1.1.1). In this model, PIDs are the principal anchor for depositing, managing and accessing digital objects that are stored in a repository.

In this model, the user, or originator, requests a PID for certain "work," which he or she then hands over to a depositor (who may be the same person). The depositor then deposits a DO in a repository by using a standard protocol (RAP = Repository Access Protocol). In doing so, the depositor offers a PID, a bit sequence and some metadata (a minimal Key-Metadata containing the PID and other metadata as understood by standards and data function) to be uploaded. The repository will store the registered DO consisting of both "data" and metadata identified by PIDs. Metadata and PID records may describe other properties by adding information to the metadata and PID records. In particular, these records may capture information about rights, type, RoR flag (originating

[2] http://www.cnri.reston.va.us/doa.html; http://www.cnri.reston.va.us/k-w.html

repository) and mutable flag. They also may contain transaction record storing access information that can be referenced by the PID. Using this model users can access the stored information (data, metadata and transaction record) via the RAP protocol, another action in which the PID is essential.

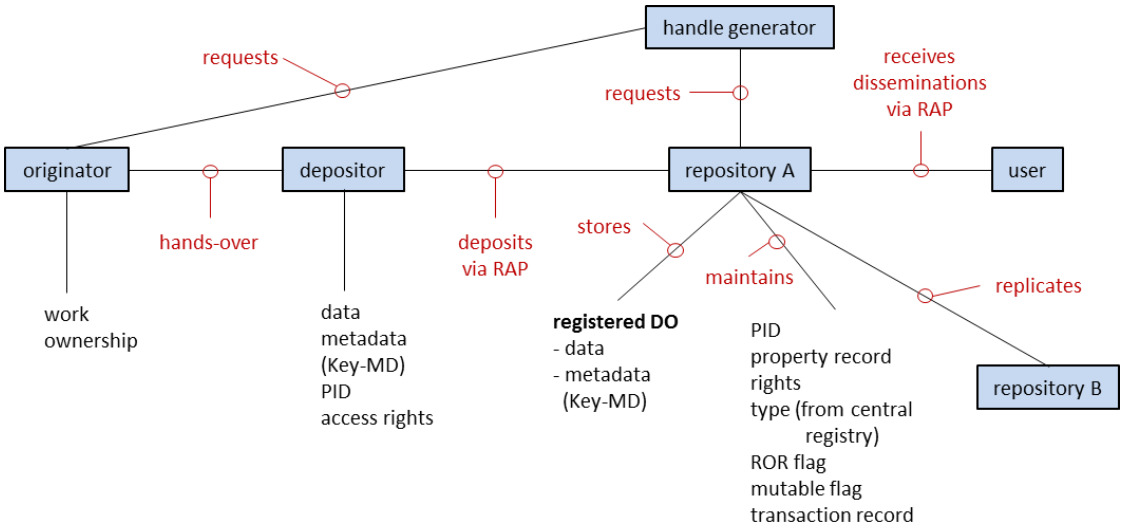


Figure 9: Graphic Representation of essentials of Kahn & Wilensky's model

Below is information about the terminology Kahn & Wilensky are using:

- **originator** = the creater of digital works and owner of the data; the originator can already request handles[3]
- **depositor** = transforms digital work into a DO (incl. metadata), deposits DO, specifies access rights and provides PID if available
- **digital object** (DO) = instance of an abstract data type with two components (typed data + key metadata (as part of more metadata, includes a handle); can be elementary and composed; registered DOs are DOs with a handle; DO content is not considered
- **repository** (Rep) = network accessible repositories that upload digital objects and support access to them; has mechanisms for deposit and access; has a unique name (X.Y.Z) to be registered centrally; stores additional data about DOs; one rep is the ROR
- **RAP** (Rep access protocol) = simple access protocol with minimal functionality required for Digital Object Architecture; repositories can specify more
- **Dissemination** = the data stream a user receives upon request via RAP
- **ROR** (repository of record) = the repository where data was stored first; controls replication process
- **Meta-Objects** (MO) = objects that store mainly references
- **mutable DOs** = indication that DOs can be modified
- **property record** = contains various information about DO (metadata, etc.)
- **type** = data about DOs have a type
- **transaction record** = the recordings of all disseminations of a DO

The Kahn/Wilensky model was used in several European projects as a blueprint for architecture discussions. For presentation purposes at the DFT meeting, a simple diagram (see Figure 9) was created showing the major components. It adds to the model, showing that in certain situations (i.e.

[3] Handles are persistent identifiers as the Handle System is creating/using them, i.e. they adhere to a specific syntax and agreements. Handles are being issued by different authorities using different business models. DOIs are handles as they are issued by the IDF and DataCite for example. But Handles are also issued by many other Handle registration authorities such as EPIC (European PID Consortium) for example.

curation) the repository could be the actor that request PIDs. The diagram also adds a second repository where the same bit sequence could be stored. In this case, the PID record would contain two paths leading to the same bit sequence.

1.3 DFT Concept Note Data Model

The DFT concept note, which includes contributions from 13 colleagues many of whom are engaged in RDA working groups, introduces a simple data model. This model overlaps with the presentations by Wittenburg (1.1.4) and Kahn (1.1.1) and identifies the digital object using the PID and the properties described in the PID record. The PID record includes access information pointing to the locations where instances of bit sequences are being stored and to the original metadata description[4]. The metadata description includes many properties typing the object, including context and provenance. In addition to these the metadata includes the PID (described in Kahn's model as Key-Metadata). Thus like other metadata the PID and its associated information are important for handling the digital object whether for management or access purposes.

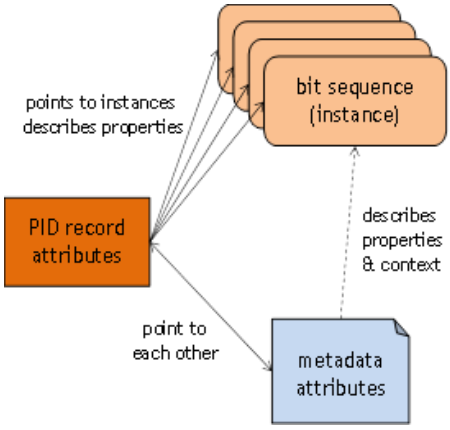


Figure 10: Data Model described in DFT

This data model is illustrated in the hour-glass diagram (figure 11), which was developed by Larry Lannom and is similar to a graphic depiction of the Internet in which the ID number plays a central role. As shown in the lower part of the Figure, digital objects can be stored in many types of containers. A PID resolution system resolves PIDs into access paths. It is the task of the underlying software then to exchange the correct bit sequence. It should be noted here that also metadata descriptions are digital objects. All kinds of services can now access digital objects by submitting a PID to a resolution system (given appropriate access permissions).

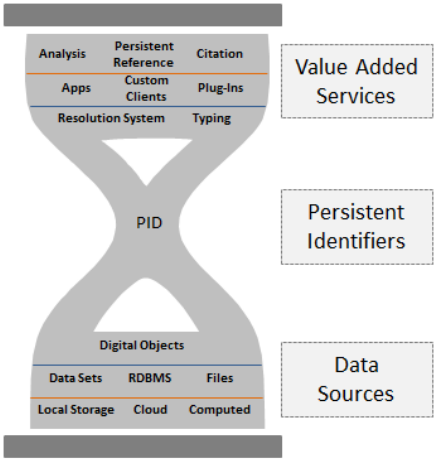


Figure 11: Lannom's hour glass model

[4] Metadata is open and can be reused and modified by anyone, therefore it is important to maintain the notion of "original metadata" created/maintained by authorized persons.

This canonical access mechanism is described in another diagram provided by Larry Lannom (see Figure 12). First, a discovery process may take place to find a suitable DO, which results in a PID. The PID is then submitted to a PID resolver, resulting in a path that can be used to access the identified DO, provided access rights are set appropriately. The received bitstreams, in combination with the contextual and provenance information contained in the metadata, can be interpreted and re-used as needed. Once again, this canonical access model is based on the triple of metadata, PID and bit sequences stored in repositories that are resolved by a resolution system.

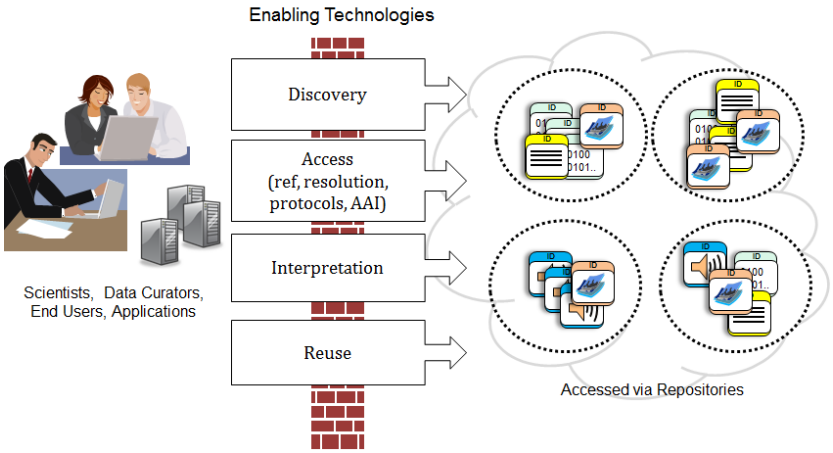


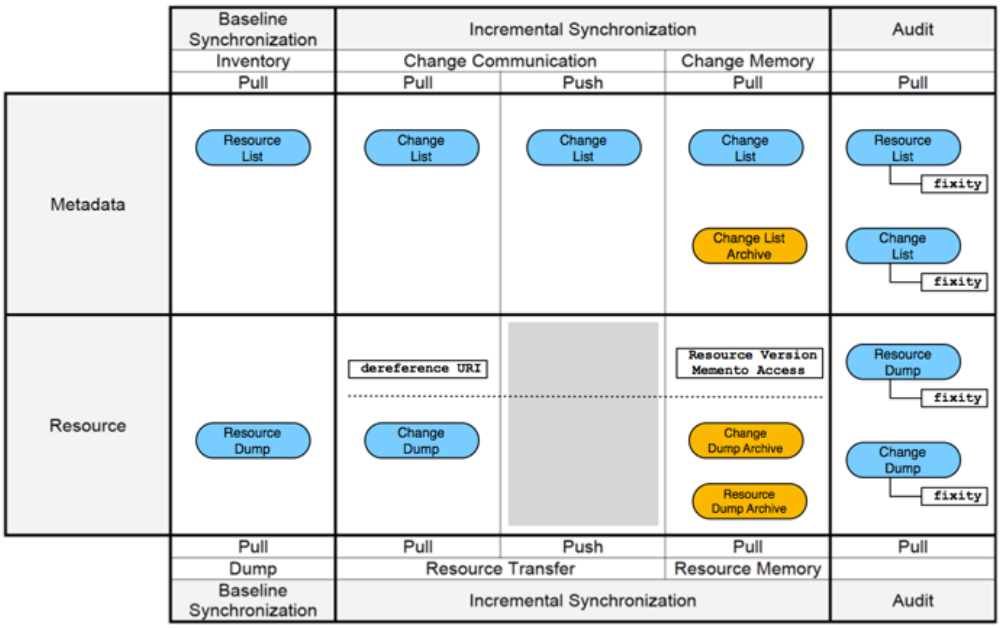
Figure 12: Layered Data Access

1.4 ResouceSync Data Model

ResourceSync is an emerging NISO/OAI standard to support the synchronization of web resources, defined as anything that has a HTTP URI and returns a representation when that URI is dereferenced. ResourceSync specifies a set of modular capabilities that a server (called a source) uses to make resources available for synchronization. These capabilities can be implement to allow systems and applications (called destinations) to synchronize with resources as they evolve. ResourceSync specifies pull-based and push-based capabilities (called notifications). All capabilities are implemented as extensions of the Sitemap protocol that is used by search engines to crawl web sites and is widely supported by web servers. ResourceSync operates in a problem domain similar to that of OAI-PMH, with a few differences: (a) it applies not only to metadata but to web resources in general; and (b) it approaches the problem from a web-centric, not repository-centric, perspective. ResourceSync specifies an exchange protocol rather than a data model; nevertheless it is possible to reconstruct some underlying assumptions that it makes.

An overview of the model from Klein et al (2013) is shown below which organizes what happens when a resource provider, or source publishes a capability list that outlines which of the ResourceSync capabilities it supports. As shown in the graphic these capabilities are Resource List(RL), Change List, Resource Dump, Change Dump, Archives, and Notification.

A resource provider, or source publishes a capability list that outlines which of the ResourceSync capabilities it supports. Those capabilities are Resource List, Change List, Resource Dump, Change Dump, Archives, and Notification. A Resource List (RL) enumerates the resources that are available for synchronization. A Source recurrently publishes an up-to-date RL. The RL contains one entry per resource (consisting of a bitstream identified by HTTP URI), as well as attributes that describe the external properties of the resource, such as last modification, date/time, hash value, size, etc. Resource Dumps (RD) are generated for efficient processing. Each RD contains pointers to packaged content, and each package is a ZIP file containing the bitstreams as well as a manifest containing information about the bitstreams, similar as in a RL.



In addition to RLs and RDs, a Source can recurrently publish Change Lists and Change Dumps that pertain only to resources that have changed within a specified time interval. This increases processing efficiency for a Destination and achieves lower synchronization latency. Change Lists can be made available using a pull and push notification approach.

ResourceSync does not make assumptions about the organization of the data, such as whether there are collections, special relations between data objects, or whether the bitstream contains metadata or data. The specific content that is transferred and the way that this content is interpreted on the destination site determines whether all relations between the different bitstreams transmitted can be re-established by user provided software. ResourceSync, however, does introduce the notion of "linking to related resources," and makes various use cases explicit in the specification, including linking to mirror copies, interlinking content and metadata about content, linking to bitstreams that have differences between prior and current version. Because it covers web resources and linked data that is outside of repositories it potentially represents a more general model of synchronization and not just duplication that may be useful to additional communities using web resources.

1.5 CLARIN Language Data Infrastructure

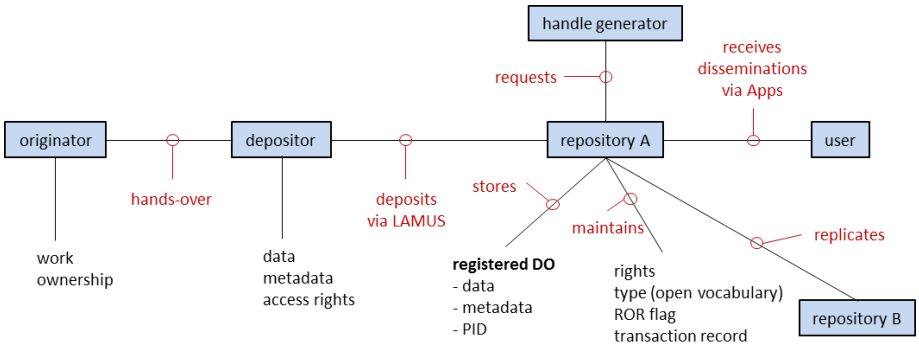


Figure 13: CLARIN abstract model

The basic model for a data organization chosen within the CLARIN research infrastructure[5] is very much influenced by the Kahn & Wilensky paper, as illustrated by Figure 13. The differences are marginal. The repository requests a PID association with a DO once it is ingested, typed checked, etc. However, the depositor must provide metadata and a variety of information about properties to be stored within the PID and metadata descriptions. The repository also notes if the bit sequence is being replicated, and the PID record is then extended to cover the path (URL) to access the new instances.

No standard protocol has been defined to access DOs from a repository, however, it is accepted that PIDs play the crucial role as one part of the process.

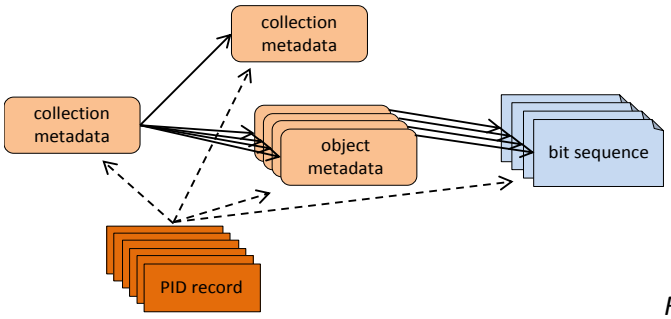


Figure 14: Collection model in CLARIN

Based on its core model, CLARIN also formulated a model for collections. A collection is defined by a metadata object that stores a number of attributes describing properties of the collection. This means it also has an associated PID to make it referable. In particular, a collection contains PIDs that point to the metadata objects of the data objects it contains. The definition is recursive, i.e. a collection can include other collections.

1.6 EPOS Earth Observation Infrastructure

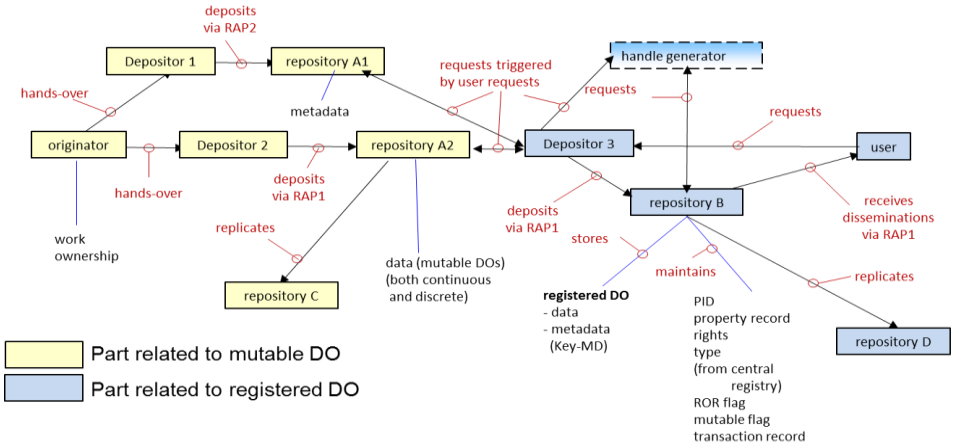


Figure 15: EPOS abstract model

The European Plate Observing System (EPOS) and the solid Earth science community in general, acquire many diverse types of data (e.g., data from sensors installed in the field, geo-referenced field data, experimental data in laboratories, etc.). The most peculiar type of data acquired are likely those obtained by equipment installed in the field that feature real-time connections (e.g., digital seismometric stations, high-resolution geodetic GPS receivers, borehole tensor strainmeters, etc.).

[5] As in many other research infrastructures, we cannot claim that this model has been implemented by all major repositories. The transition will take quite some time, but CLARIN repositories that want to obtain a seal for being assessed using this system need to fulfill basic requirements.

Based on the Kahn & Wilensky model, the EPOS seismological community drew a first sketch of its possible data organization (see Figure 15) contemplating different stages before providing the registered DO. However, and as noted earlier, much is still under discussion concerning these processes, especially how best to deal with the dynamic nature of the data objects. Dynamic data collected in real time often means an initial temporary incompleteness in the data owing to data transmission gaps that are being filled over time. Thus data that is pre-processed into a registered, digital object with an identifier may be mutable in the sense that it is incomplete after additional data having been added to it.

There are several major challenges posed by incomplete real-time data streams, such as how to refer to a specific early and probably incomplete (mutable) version of the data set, how to replicate such data for access, and how to design persistent solutions. For now, the focus is on introducing the digital object model within the EPOS community and matters related to versioning will be sorted out later since best practices have not been established.

1.7 ENES Climate Modeling Data Infrastructure

The data organization model in the area of climate modeling has been worked out in the European Network for Earth System (ENES) modeling infrastructure and has been implemented at the World Climate Data Center. In addition to the normal data objects, ENES distinguishes metadata objects, information objects (with more general information about the data) and a transaction record – and all four are interlinked.

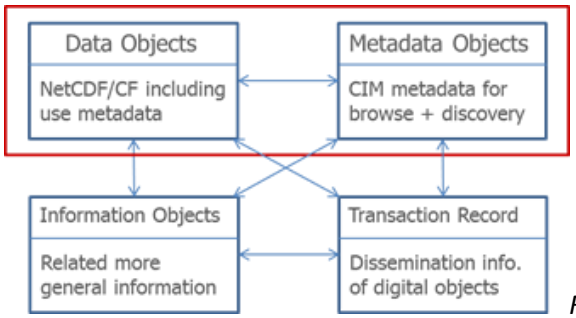


Figure 16: ENES information components

Defining metadata as objects is common practice (see CLARIN, etc.) and transaction records have also been mentioned by Kahn (section 1.1.1). Often, HTML landing pages with broad, elaborate descriptions of the context of data are generated, and these should also be linked (bound) to other data entities. This is very much in line with Chen's elaboration on binding.

In addition, this data organization model can be mapped to the abstract model described in the Kahn & Wilensky model. The climate data community aggregates objects to large collections (blobs) and registers PIDs[6] to make the "blobs" citable. Currently the model is being extended to register explicit PIDs for each digital object created during the simulations. A "blob" can be thought of as a collection where the metadata includes pointers to all included data objects. It is worth noting that the model shares some commonality, at least of interest with the Pfeiffenberger sketch in the relation/registration of digital objects with publications

[6] For this purpose DOIs are being registered.

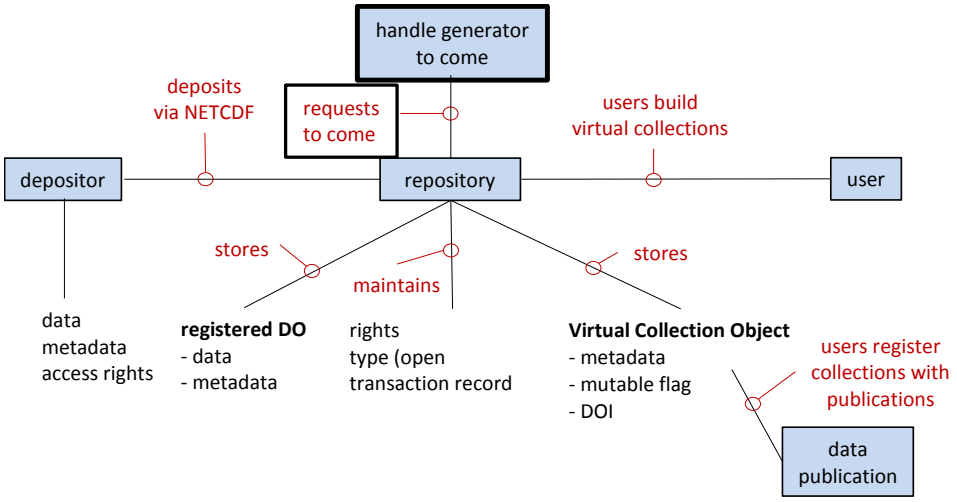


Figure 17: ENES abstract model

1.8 ENVRI Environmental Data Infrastructure

The ENVRI project (Common Operations of Environmental Research Infrastructure) gathers European ESFRI environmental research infrastructures and is working to develop a reference model, which is a community standard that can be used as a blueprint for all ESFRI environmental science communities to build interoperable infrastructures. Based on the requirement analysis, the ENVRI Reference Model captures the full data lifecycle in an environmental research infrastructure, modelling computational characteristics and data structures in data acquisition, curation, access, processing, and community support. At its core, it makes use of the ODP (Open Distributed Processing) framework, an ISO standard (ISO10746, 1-4), as an overall conceptual framework for building distributed computing systems. Based on this framework, the Reference Model describes a principle set of sub-systems that are involved in dealing with the data at various phases.

Using a viewpoint approach, ODP provides a framework for design and specification of a large-scale distributed system, concerning 5 different aspects, including, community requirements, data structures, computational functionalities, engineering mechanism, and implementation methods. ODP helps to specify ENVRI 5-common subsystems at an abstract level, it also defines useful terminology, such as:

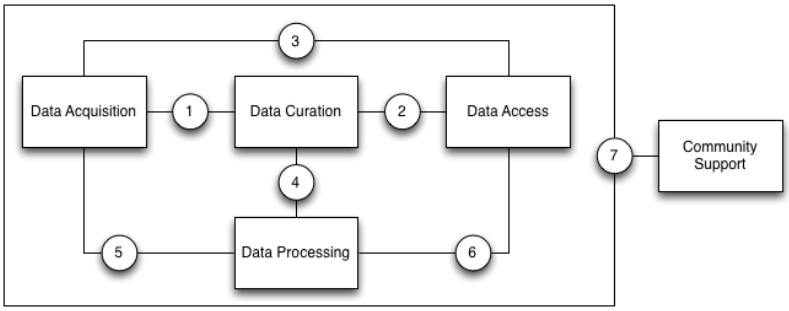


Figure 18: ENVRI process model

***Architecture* (of a system):** *A set of rules to define the structure of a system and the interrelationships between its parts.*

***Interface**: An abstraction of the behavior of an object that consists of a subset of the interactions of that object together with a set of constraints on when they may occur. Each interaction of an object belongs to a unique interface. Thus the interfaces of an object form a partition of the interactions of that object.*

1.9 EUDAT Data Infrastructure Core Model

The European Data Infrastructure (EUDAT) is a European cross-disciplinary initiative to design and develop common data services thus interfacing with about 20 different scientific communities. The services having been launched are: (1) safe replication in different flavors (logical replication based on iRODS installation, physical replication based on gridFTP, and custom replication for repository systems such as Fedora, D-Space, etc.); (2) data staging, where data is staged to HPC pipelines and where result data is pushed back into the EUDAT data domain; (3) a YouTube-like service, which allows users to upload and share data; (4) a DropBox-like service, which allows users to synchronize directories and share data; (5) a semantic annotation service, which creates corrected and extended versions of data objects; and (6) a metadata catalogue service. Despite all the different communities involved, EUDAT needs to have its own core data model to be able to easily integrate the various services.

Therefore EUDAT decided to make its data domain a domain of registered data, i.e. independent of the type of data service it should adhere to the basic data model described in Section 1.3 as follows: (1) A digital object has a bit sequence encoding some content that can be stored at several locations; (2) A digital object has an explicit PID registered with a certified authority that allows identifying an object, proving its integrity and finding its location; and (3) A digital object has a metadata description that contains at least the PID, but usually also describes some external and internal properties of the object to enable interpretation and reuse. The EUDAT model therefore fits with the core model described by Kahn & Wilensky.

1.10 DataOne Data Model

DataONE's basic data organization is based on the following principles:

- Data and scientific metadata are provided by researchers and are separate digital objects ("units of digital content"), each being identified by a persistent identifier (PID).
- Both data and scientific metadata have system metadata associated with them that store properties such as hash value, time stamp, owner, and obsolescence.
- Currently data and scientific metadata are immutable in DataONE.
- The relationship between data and scientific metadata is maintained by grouping them into an Open Archives Initiative's Object Reuse and Exchange (ORE) Aggregation (ORE Aggregations have a PID) and by expressing appropriate relationships between them.
- Any number of data or metadata objects may be contained within an ORE package.
- ORE packages may additionally contain references to other ORE packages.
- PIDs do not have information associated with them.

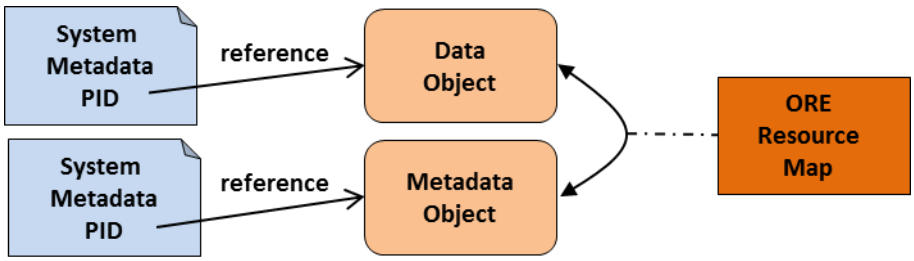


Figure 19: DataONE data model

Data sets[7] within DataONE may be hierarchical since a single data set may contain PIDs that refer to other ORE packages. The relations used in ORE packages include the CITO (http://purl.org/spar/cito) *documents* and *isDocumentedBy* relations, but may include additional types of relationships as deemed necessary by the package creator. The DataOne model as noted separates data and metadata (each is immutable) unlike approaches that bind at least some of these in an identity record. Relationships are handled by ORE aggregations which are discussed in sub-section 1.13.

1.11 Earth Science System Data

A group working on storm surge forecasting led by RENCI and UNC departments is studying, with the help of high-resolution simulations, the effects of storm surge on coastal zones. Model simulations are based on a number of observational data from sensors and from forecasts from other institutions and government agencies - all submitted in well-known standard formats such as netCDF, which includes a header to store metadata.

The modeling software processes the incoming data objects and generates time series data containing predictions of storm surge for areas of interest. The data sets are stored in the netCDF format, which is metadata compliant, with the widely used CF-UGRID convention. All data is stored in a THREDDS compliant federated repository infrastructure so that the data and metadata can be accessed via application programs.

With respect to data organization the netCDF format is key and standard compliant metadata as part of netCDF objects is being separated and stored in a special database which also includes pointers to the bitstreams. ESSD has repositories that store what they call data object but as of now, no PIDs are being used as part of registration in the repository.

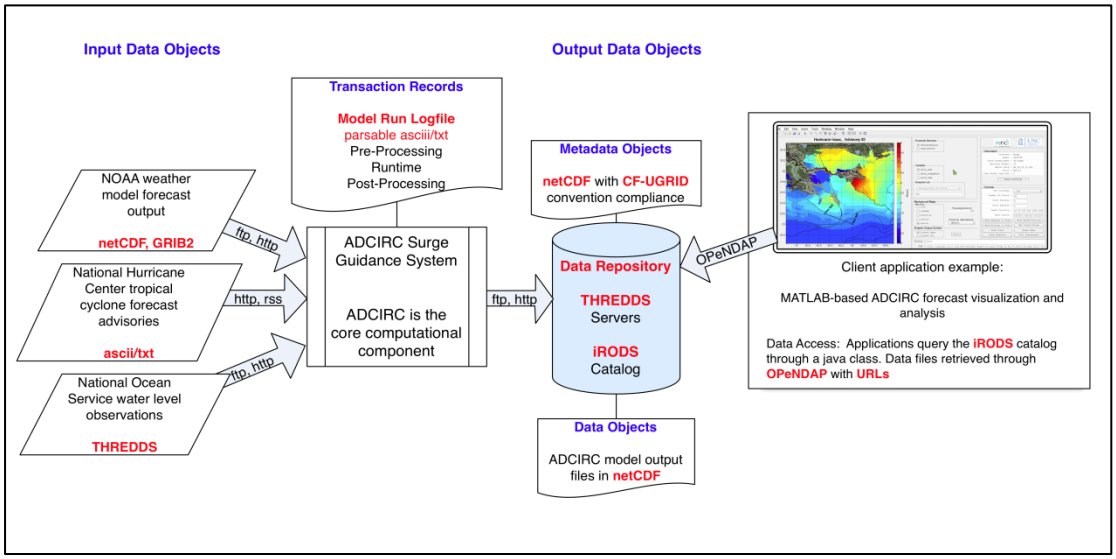


Figure 20: ESSD data flow model

1.12 Clinical Genomics Workflow

A group generating and working with genomic data, led by RENCI and the UNC School of Medicine, has a great interest in proper data organization and workflows to support research in an area where much data is being managed. The group's research involves developing a process that manages and secures genetic data throughout its lifecycle—from obtaining raw genetic material from patients, to analysis by a geneticist, to diagnosis and treatment decisions by clinicians for the patient.

[7] DataONE uses the term "data set" instead of "collection" since it is more clearly defined.

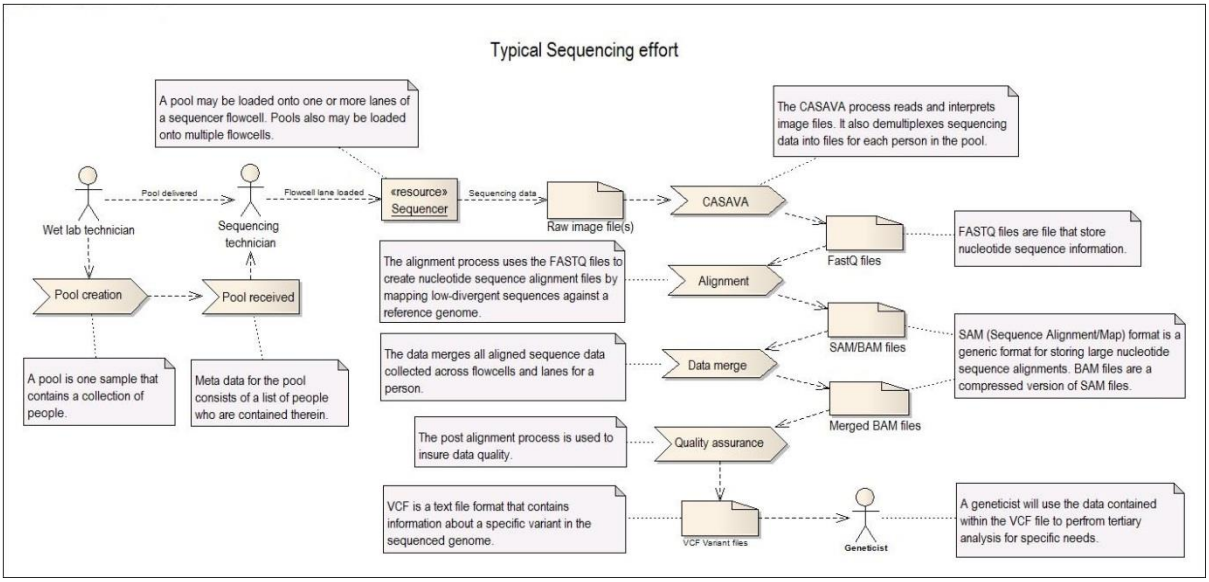


Figure 21: Genomics data flow model (RENCI, UNC)

The process consists of seven steps: (1) preparing a pool of samples from patients and creating metadata consisting mainly of ID lists of individuals; (2) generating raw sequencing data files, which are image files; (3) using specialized software to transform image data into text files, including sequencing information and quality scores in the FASTQ format; (4) aligning sequencing information with reference genome data to understand which sequences are variants that could be markers for diseases. All information is stored in SAM files, which are then transformed to Binary Alignment Files as highly compressed forms; (5) merging more samples of sequenced and processed data; (6) conducting pattern processing to determine the quality of the sequencing and alignment steps; and (7) extracting files that include specific variants for further analysis. These files bundle variant sequences, some header information (metadata) and some position information.

The data organization in the case of sequencing data is straightforward and results from the clear workflow in this kind of research. Management of the data objects is handled locally by IDs such as patient IDs.

1.13 ORE Data Model

The Open Archives Initiative's Object Reuse and Exchange (ORE) is a data model and associated RDF vocabulary to describe and exchange aggregations of web resources. The model is based on the web architecture and leverages semantic web and linked data principles. An ORE Aggregation represents a set of web resources and provides an identity for that set. An ORE Aggregation is described by an ORE Resource Map (ReM), an RDF document that uses the ORE vocabulary and that is published to the web. A ReM is a named graph. The ReM asserts the finite set of constituent resources (the aggregated resources) of the aggregation, and it can express types, properties, and relationships pertaining to the aggregation and its aggregated resources. Both an Aggregation and the ReM that describes it are identified by HTTP URIs, but PIDs can be associated with them using appropriate relationships.

Figure 22 below shows an Aggregation, described by a Resource Map consisting of aggregated resources, and some properties, types, and relationships. The PID for the Aggregation is expressed via the ore:similarTo relationship. In the context of the Research Object effort, which includes ORE at its core (see http://www.w3.org/community/rosc/), the aspect of identification of Aggregations by means of HTTP URIs and PIDs need to be worked out.

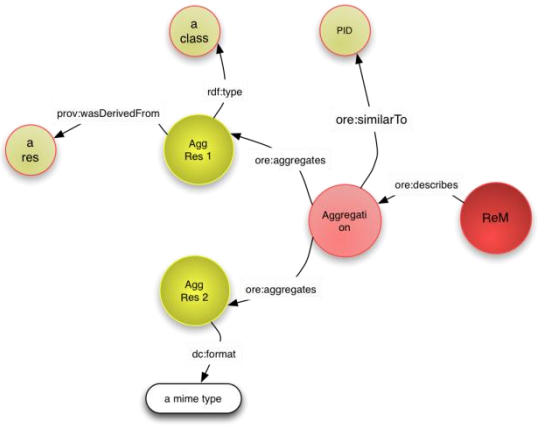


Figure 22: ORE aggregation model

1.14 Fedora Object Model

Fedora is a repository system that supports an explicit and well-described digital object model which can cover different object types: (1) Data Objects containing digital content (data); (2) Service Definition Objects storing models/descriptions of services; (3) Service Deployment Objects describing how services are delivered; and (4) Content Model Objects allowing defining classes of digital objects. In the context of this paper, only Data Objects will be discussed in more detail. It should be noted that in Fedora, digital objects are embedded in XML structures according to the FOXML schema.

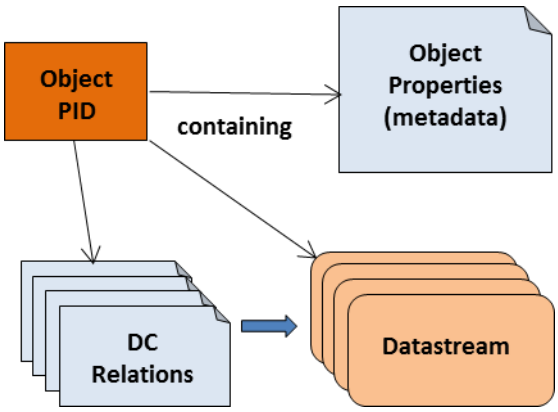


Figure 23: Fedora schematic object model

Basic elements are data-streams representing the raw content (images, text, metadata, etc.) as bitstreams or as references to external locations. Each DO can include several streams. These streams are described by object properties, i.e. the properties describe the characteristics of all streams included - a feature that could be used to bundle versions and presentations. Below are the types of object properties Fedora suggests for describing a data stream:

- **Datastream Identifier:** an identifier for the datastream that is unique within the digital object (but not necessarily globally unique)
- **State:** the datastream state of Active, Inactive, or Deleted
- **Created Date:** the date/time that the datastream was created (assigned by the repository service)
- **Modified Date:** the date/time that the datastream was modified (assigned by the repository service)
- **Versionable:** an indicator (true/false) as to whether the repository service should version the datastream. By default the repository versions all datastreams.
- **Label:** a descriptive label for the datastream

- **MIME Type:** the MIME type of the datastream (required)
- **Format Identifier:** an optional format identifier for the datastream. Examples of emerging schemes are PRONOM and the Global Digital Format Registry (GDRF).
- **Alternate Identifiers:** one or more alternate identifiers for the datastream. Such identifiers could be local identifiers or global identifiers such as Handles or DOI.
- **Checksum:** an integrity stamp for the datastream which can be calculated using one of many standard algorithms (MD5, SHA-1, etc.)
- **Byte stream Content:** the "stuff" of the datastream is about (such as a document, digital image, video, metadata record)
- **Control Group:** pertaining the the byte stream content, a new datastream can be defined as one of four types, or control groups, as follows:
    - ***Internal XML Metadata*** -
    - ***Managed Content***
    - ***External Referenced Content***
    - ***Redirect Referenced Content*** -

See http://www.fedora-commons.org/documentation/3.0b1/userdocs/digitalobjects/objectModel.html for more.

Each DO (as opposed to a data stream) is identified by a PID which is part of the DO. Some data streams are generated automatically such as a DC description per stream containing key object metadata (ID, state, date, version, format, checksum, etc.). Using simple graphics we've represented this model in Figure 23.

The DO serves as a container, is associated with a PID and includes metadata describing some generic DO properties. It contains several data streams covering content, metadata, relations, etc., i.e. all included streams and possible relationships between them are being described. In this graphic, the arrows implicitly mean "containing" and the PID can be interpreted as "representing" the DO.

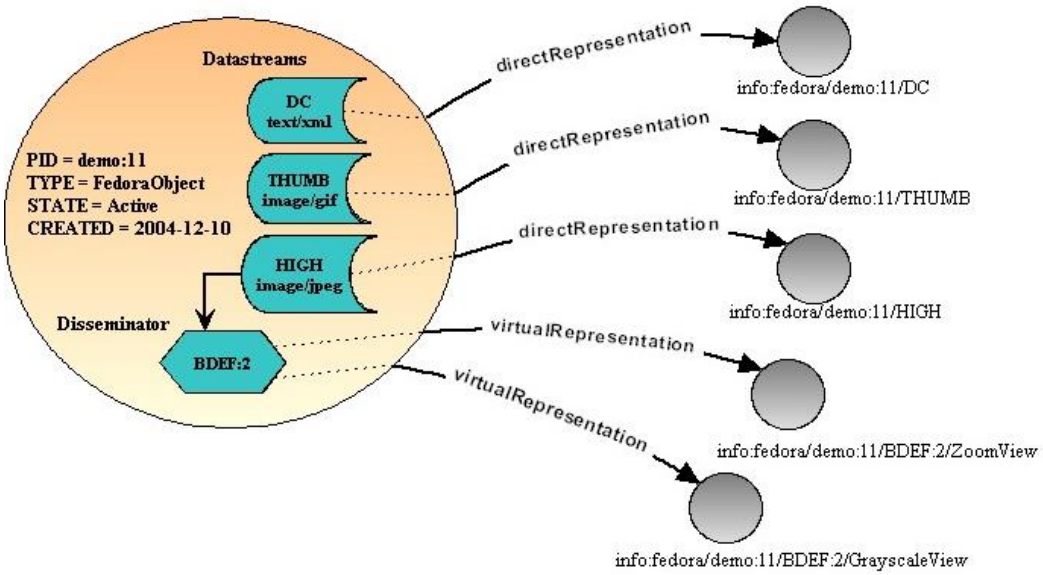


Figure 24: Fedora digital object model

Figure 24 shows an example how a Fedora container can be used to bundle different data streams. It should be noted that Fedora DOs can be expressed by ORE Resource Maps. Typed relations guarantee that machines can find the expected content.

1.15 Europeana Data Model

Europeana is a portal which harvests metadata from a large variety of museums, archives and libraries and thus allows users to search for interesting objects. Because it harvests from many providers, it is confronted with a large variety of metadata standards. Currently the basic standard, often a least common denominator for data sharing, is Dublin Core, which implies an enormous reduction of the rich information that could otherwise be offered at the portal. To alleviate the need to reduce the data, a new (meta) data model was invented called EDM (Europeana Data Model) that offers the flexibility to not only cope with the variety of data, but also to allow enrichments of the data. EDM is based on the principles of the Semantic Web community, i.e. it makes use of RDF, RDFS and OWL semantics where possible, knowing that metadata statements are assertions and thus can be transformed into RDF triples with explicit relational semantics. Thus EDM is not based on a fixed schema. Instead, it provides generic semantic classes and properties and offers an anchor to which other models can be attached. The system makes it unnecessary for metadata providers to change their practices except to be explicit about the semantics used.

EDM makes some very important distinctions about what to include as documentation, which were also present in other models: (1) It includes both "work" or "original creation" as well as "various digital representations" of that work or even of fragments of it; (2) It offers metadata descriptions describing "works" and their "digital representations." There can be several metadata descriptions from different origins describing the same object in different ways; and (3) Objects containing other objects are possible.

Different metadata describing one object and/or its various representations are treated as ORE aggregations with their own identities. EDM provides a number of super-classes and super-properties that allow users to see entities provided by metadata providers as specializations. Also EDM includes a number of classes to represent contextual entities, which allows the data to connect to other knowledge sources. An important aspect of EDM is to establish useful mappings and relations in order to develop a coherent metadata domain that can be queried.

It is not the task of this paper to elaborate on the details of EDM, but to note that it provides useful ideas for minimal metadata. As a mechanism it should be noted that all non-mutable objects, including metadata descriptions, are identified by stable URIs.

1.16 MediaWiki Data Model

MediaWiki and many other content management interaction platforms have a simple model: the Wiki knows about web pages, their versions, and links between different content pages and external information sources, most of which can be easily configured for collaborative purposes. References are in the form of links (i.e. URIs which, in the often short-lived wiki user community, are often no more than URLs without persistency).

All structuring of data is a task the data manager must organize. The Wiki pages are often landing pages with descriptions (prose and/or keyword metadata in tables) and references to file systems or databases where the data objects are stored. Navigation and search tools that index web pages from the descriptions allow users to search for interesting data.

1.17 DataCite/EPIC Data Models

DataCite and EPIC are not initiatives that maintain repositories and thus they do not hold data. However, they have a service that allows resolving PIDs[8] into access paths among other functions. DataCite primarily refers to data collections that are citable in the way publications are citable and

[8] EPIC is issuing Handles, DataCite is issuing DOIs which are also Handles.

that are stabilized results of a completed research workflow. EPIC allows researchers to register any data object that is created as part of daily research workflows, which means the requirements for the persistence of data are less strong.

In both cases, the assumption is that there is meaningful content (data streams) stored at various locations, that the PID can resolve to the access paths of these locations, and that the PID record contains attributes that describe properties, and point to its metadata, provenance descriptions, etc. In DataCite, citable metadata plays an important role, since it is metadata that can immediately be retrieved when accessing the PID record. DataCite has developed a schema for its metadata[9], which has been widely adopted by EPIC. Both EPIC and DataCite can integrate additional attributes, and these might be adopted in the RDA PID Information Type working group. So far neither make assumptions about the availability of metadata or provenance descriptions associated with the data stream.

In conclusion, both models fit nicely with the model presented by Kahn & Wilensky.

1.18 OAIS Model

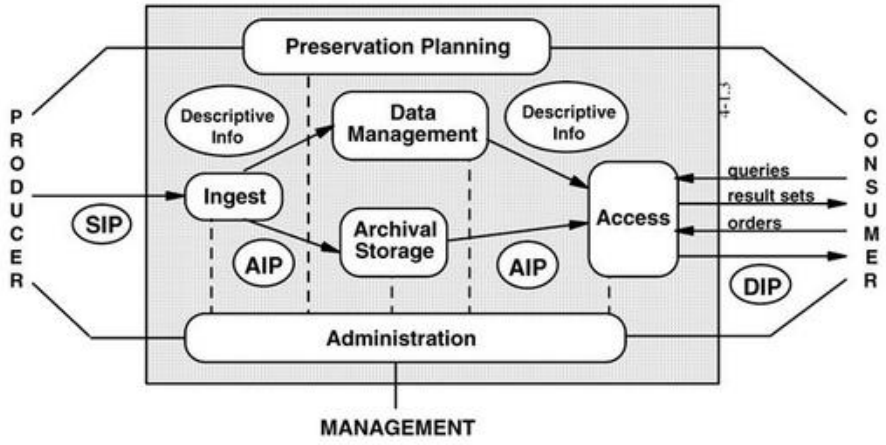
The Open Archival Information System (OAIS) is a well-known reference model for a dynamic and extendable archiving system. An archive is an organization which requires a close interaction between system components and humans to guarantee user access over many years. The model is an abstract specification, introduces relevant and now common terminology, and does not prescribe any specific implementation. The model describes how producers generate and ingest submissions, which steps need to be taken to have a functioning archive system, and how users can access the stored data via distributions.

Producers generate Submission Information Packages (SIP) and ingest them into the archive system. Internally, SIPs are transformed into Archival Information Packages (AIP) which is stored in the archival system.

> *Information Package* is a container that binds the Content Information with associated *Preservation Description Information*. Preservation Description Information is information that is essential to adequately preserve the particular Content Information to which it is bound. An Information Package is serialized using *Packaging Information*. This Packaging Information also provides local hooks (often called 'Fragment Identifiers') into the Information Package to allow accessing each file of which the Content Data Object of the Content Information consists. From ACCESS INTERFACES FOR OPEN ARCHIVAL INFORMATION SYSTEMS BASED ON THE OAI-PMH AND THE OPENURL FRAMEWORK FOR CONTEXT-SENSITIVE SERVICES
>
> Jeroen Bekaert, and Herbert Van de Sompel, 2005

Package transformation step can include many checks (for completeness, integrity, format coherence, etc.). Administration of all AIPs is handled as a data management component. Users (consumers) can access the stored objects by issuing queries. An access component transforms AIPs into Dissemination Information Packages (DIP), which, once access permissions have been checked, is handed over to the user for further processing. A Preservation Planning component takes care of replication, migration, emulation and curation to guarantee long-term archiving and interpretability. An Administration Component keeps track of and logs all processes.

[9] http://schema.datacite.org/



Although the OAIS model is not very specific, it can be a starting point for some terminology and it includes higher level packaging concepts not found in other approaches. It does not seem to rely on a PID approach but have more local identifiers within an Information Package.

1.19 DICE Data Model

In the realm of the iRODS policy-based design and development project, the DICE group came up with a comprehensive data model which has a number of core design goals. The data model focuses on the processes that comprise data management, and the mechanisms that automate the execution of these processes. Example processes include administrative tasks (data replication, format transformation, usage report generation), policy enforcement (integrity, authenticity, arrangement), and assessment (repository trustworthiness, policy compliance).

Complementary to the notions of data objects and collections are operations that need to be carried out to support life cycle management, long term survival, access, interoperability and trust with respect to integrity and authenticity. Explicit policies are designed to govern all operations on officially registered data objects and collections[10], and procedures are functions that implement these policies. The WG on Practical Policy has defined a concept graph detailing the relationship between the purpose for a collection, and the corresponding collection properties, policies, procedures, and persistent state information.

Recently the DICE colleagues stated that for them

- Data are the objects that are being managed, and consist of bits or bytes.
- Information is the application of a name to an object and the categorization/organization of a set of objects.
- Knowledge is the specification of a relationship between named objects through a transformational process.

According to this dynamic perspective, information is a static property that can be stored as a metadata attribute while data are bits deposited in a storage system.

For DICE the term data collection (which is knowledge in the above sense) is central since it will be the set of related data objects sharing some properties that will be the subject of management, curation, transformation, migration and other processes. A comprehensive set of name spaces is used to identify collections and files, and the environment in which the files are managed (users, storage, metadata, procedures and policies). Metadata associated with each name space are controlled by policies that enforce the desired properties.

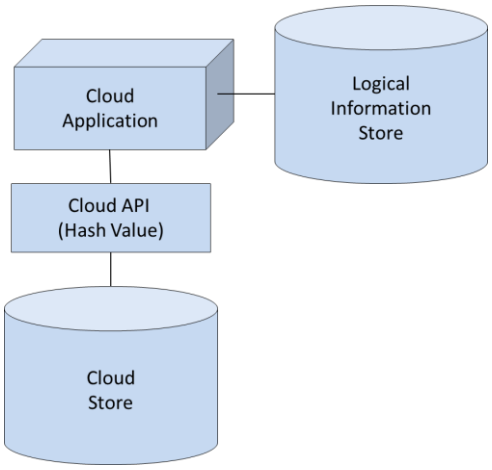
[10] Collections are aggregation of objects.

Policy-based data management systems enable the concept of active objects and active collections. Active objects or collections have persistent identifiers pointing to procedures that will generate information or derived data products. Accessing the active collection causes the procedure to be applied to the digital objects within the collection. Also metadata can be dynamically generated or extended by applying a procedure to digital data. The metadata then consists of the information generated by reification of knowledge relationships encapsulated within the procedures that were applied to the digital objects within the collection.

1.20 Physical vs. Logical Layer Separation

Still most of the researchers are using file systems to store data and include metadata type of information about the content in the file and directory names and their canonical relations by means of directory organizations. The cloud community realized, primarily for efficiency reasons, that it is more optimal to specify a simple interface (API) that basically offers a hash value as a unique key to identify a stored object in the cloud and to access it with the help of underlying parallel operations transparent to the users. In fact no property information describing the object is provided through the API, this information in cloud systems is managed by layered applications.

Therefore we can state that the development of the cloud technology helped distinguish and separate different types of information. On the one hand we have the physical information about a digital object (basically how to access its bitstream encoding the content) and all the logical[11] information which includes the typical metadata information typing the digital object, provenance information saying something about the generation history, access rights information, relational information indicating how the object relates with other objects[12], etc. The hash values used in the interfaces can be seen as internal identifiers but they do not replace the globally available persistent identifiers as discussed in the models by Kahn, Yin Chen, Wittenburg and others.



The conclusion that can be drawn from this development fits with the suggestions of many models that independent of the system that is used to store the bitstream content (file system, cloud system, database system) we need to separate physical and logical information in our complex data domain.

1.21 Reference Process Model

In RDA Europe and EUDAT about 38 interviews have been carried out with research infrastructures and research departments from different scientific disciplines about the way they are dealing with data and about their organization. In addition about 12 interviews were carried out in the German

[11] We use here the term "logical" although we actually speak about all kinds of different types of metadata. It should be noted that the term "logical" is well-defined for example in database technology to describe structures, constraints, etc. The emphasis in on the non-physical and more abstract level.

[12] We can distinguish again many types of relational information. Canonical relations emerge from the creation process (all primary data of a specific simulation or experiment, etc.). Other relations emerge from the objects' embedding in collections that have been created by users to carry out a certain analysis. Other relations can be annotations to fragments of the content, relations between content fragments of different files (semantic weaving), etc.

Radieschen project with the same intentions confirming the findings. Some explicit models included in the interviews as the ones from CLARIN, EPOS, ENES etc. have been presented already.

Although many of the colleagues that have been interviewed do not mention conceptual terms explicitly or are even not aware of the relevance of making certain information explicit (metadata often is kept in the mind of the researchers or in ephemeral spreadsheets, IDs are kept by knowing the path in directory systems, etc.) we can draft an underlying processing reference model that describes the steps that are being taken by the different researcher groups in daily practice.

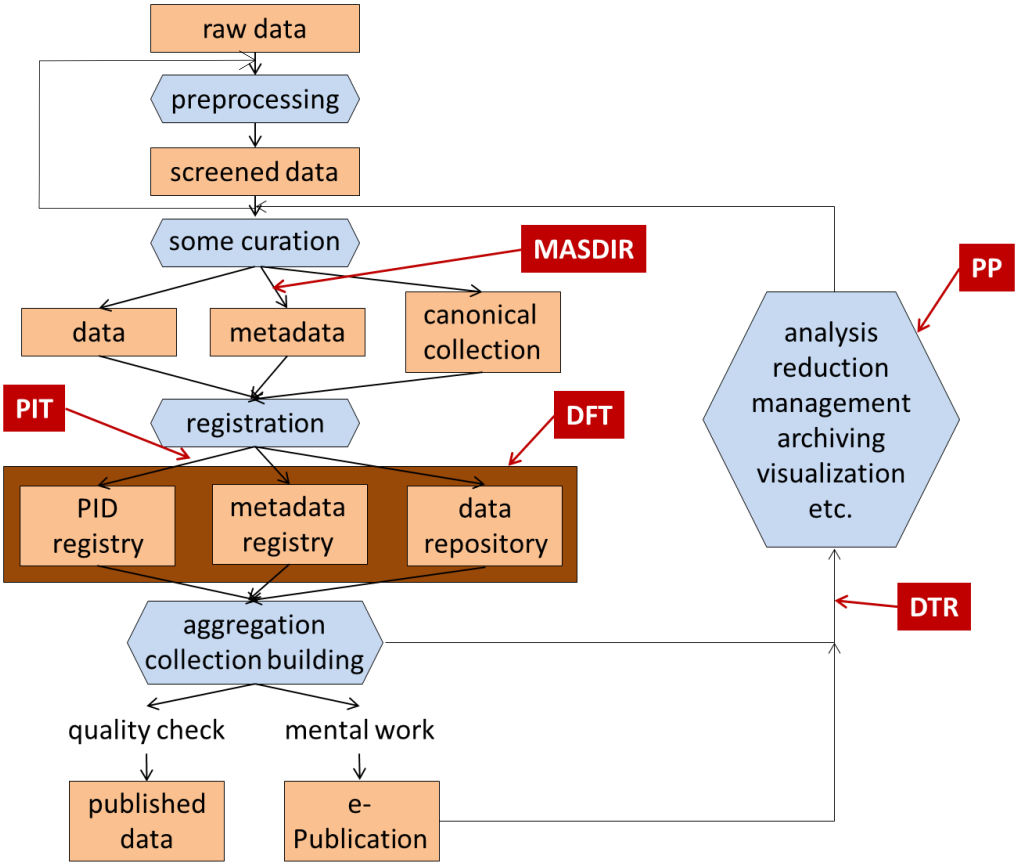
This abstract processing model (Figure below) can roughly be divided in three phases:

- preprocessing phase
- data processing/maintenance phase
- publishing phase

Preprocessing Phase

Raw data being created by sensors, simulations, crowd sourcing and other sources often need to be pre-processed for various reasons (error detection and removal, mapping with reference data, transformations/reductions, etc.), before they are released for further usage. There are some special cases where this reference model is not accurate enough:

- In the case of dynamic data people want to immediately work on the first data fragments being received while additional data fragments are filling gaps and while some preprocessing might still be carried out.
- In the case of huge data amounts as being generated by the LHC experiment at CERN for example data management and distribution activities are being started immediately after some preprocessing as integral parts of automated workflows.
- etc.



When the pre-processing phase is finished some curation tasks are being carried out which typically results in screened data, associated metadata (in whatever form this metadata may exist – file names, spreadsheets, etc.) and a "canonical" (or natural) collection organization represented in some form.

Processing Phase
In a domain of registered data ready for re-use by others a registration process will start that will result in persistent identifiers registered with an authorized institution, in metadata registered in a proper registry (or catalogue) and in data stored in a certified repository. The relations between these are indicated by some models in this document. To prepare all kinds of processing steps (management, analysis, etc.) people in general do some collection building which can include data from other repositories, i.e. it is often not the "canonical collection" formed at creation time that is used as basis for subsequent processing. In particular for analysis purposes (big data) different DOs are virtually aggregated in different collections to carry out specific calculations. The processing step results in new digital objects that again should be registered and described by metadata.

Publishing Phase
The final steps require in general human intervention. e-Publications are the product of scientific theorization and they can be subject of analysis operation. The publication of data collections associated with an ePublication is a step that in general requires additional quality checks etc.

Comments

- As indicated above we have seen during the interviews that often the steps are not being carried out explicitly by the teams, i.e. they do not have a model in mind, but work along established habits. Let's give two examples:
    - Many teams do not use PIDs but they know the paths in file systems for example. It is well-known that after a few months and eventually some system operations users can't recall easily where their versions are stored, which versions they used, etc.
    - Many teams use various ways to manage metadata without storing all relations explicitly. Also here people are losing control about how they are using data.
- It has been indicated in this processing model (see figure above) where the start-up RDA working groups are active.
    - The DFT group describes the basic concepts and terms of the core of the processing.
    - The PIT group is busy to specify the interface for requesting and resolving PIDs.
    - The MD group is registering metadata schemas to facilitate MD creation and interpretation and is working to identify metadata components (packages).
    - The DTR group amongst others specifies a type registry that will facilitate the interpretation of any data object.
    - The PP group is collecting policies that can guide the various processing steps to create reproducible science.

1.22 Chinese National Data Infrastructure for Earth System Science[13]

In the Chinese National Data Infrastructure for Earth System Science (Geodata.cn), we adopt the triple of metadata, data service and data document as the data organization and publishing model which is shown as Fig.1.

[13] Presented by Yunqiang ZHU; Department of Geo-data Science and Sharing, State Key Lab of Resources and Environmental Information, Institute of Geographic Sciences and Natural Resources Research, Chinese Academy of Sciences

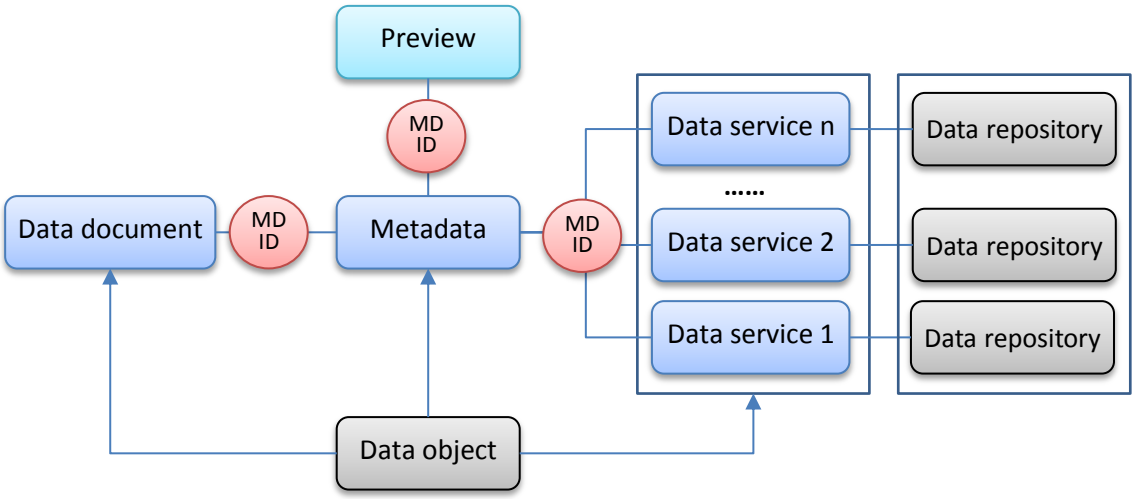


Fig.1 data organization and publishing model of Geodata.cn

Data service is the encapsulation of different access ways of one data object by uniform model. One data object is likely accessed by one or multi ways according to its storage or publish ways. For example, land use spatial data is probably stored in ArcGIS shape file, Oracle database, FTP server or published to web-GIS or WMS (web map service) simultaneously. So we can encapsulate land use spatial data as corresponding file service, database service, FTP service, HTTP service and Geographic Information (GI) service by uniform data service model. Data service model is shown as Fig.2 that includes five elements, i.e. service identifier, name, type, description (optional) and parameters (parameter key and value pairs) and presented by XML.

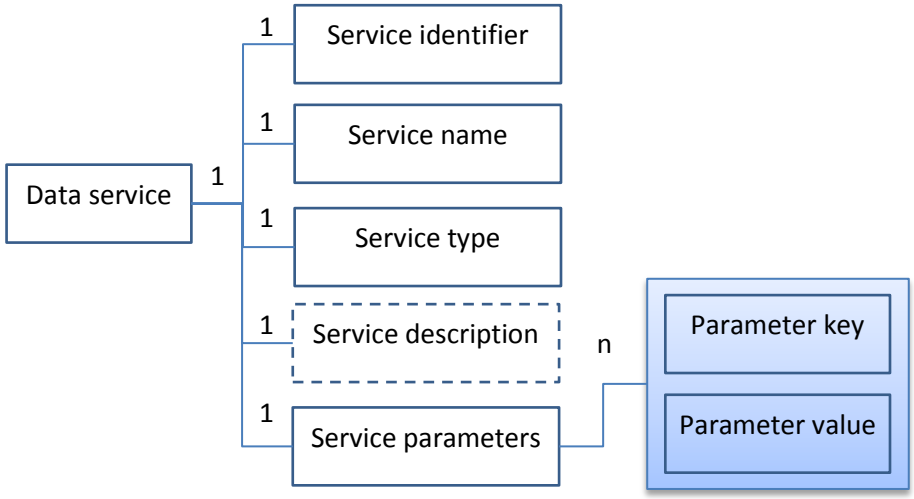


Fig.2 data service model

Different data service has corresponding service parameters, such as name and path in repository of file service; DB server URL, DB SID, user name and password of database service; server URL, path, user name and password of FTP service; name and web page URL of HTTP service; map server URL, map service ID, map name of GI service, that are encapsulated by XML key and value pairs. Metadata is data about data which describes identifier, temporal and spatial extent of data, content, quality control methods and data contributors' contact information and so on.

Data document is the detailed description of data object that covers data attributor fields and their semantic description, such as numerical unit, value domain, data temporal-spatial datum, scale or

resolution and granularity, data provenance and process method, as well as data storage and usage method, data intellectual property right declare and citation requirement. Through metadata identifier, we bind metadata with data services (1:1 or 1:n) and metadata with data document (1:1). Also we can bind preview images with metadata, so that users can get the profile of the data before they access it.

In geodata.cn data model, the purpose of metadata is quick discovering data object, and data service guarantee users accessing detailed data repositories, while data document is the guide of data usage for users. So users can not only easily discover and seamlessly get data resources, but also correctly use data resources. Moreover, we can bind several data services with one metadata, so users can select one or several data services they wanted. As the above land use spatial data (ArcGIS shape format), if the data has been published as web GIS, data provider can publish file service which need upload land use data file to server and HTTP service which need point out the URL of web GIS. After users discover this land use data through metadata, they can browse the data through the web GIS or download the data directly.

The detailed workfolw of Geo-data publish and sharing is shown as Fig.3. For providers, firstly they need publish metadata, data services and data document as well as preview image of data object to Geodata.cn protal. Secondly, they need bind data services, data document, preview image with metadata by the unique metadata identifier. For users, firstly they search metadata and read the metadata description and cooresponding preview image. After that they can decide whether the data meet their requirement. If the answer is yes, they will continuly access data services to download data resources directly or browse data online. They can download related data document to guide their data usage.

1.23 Additional Models

DFT IG will continue to collect models and use cases as they are also being collected in other WG/IGs such as PP WG, DF IG, etc.

2. References

Klein, Martin, et al. "A technical framework for resource synchronization." D-Lib Magazine 19.1 (2013): 3.

Appendix A: Terms used

Here provide a working list of relevant terms that have been mentioned/introduced within the different models. An analysis of how they are being used and how they relate to each other will be made in the next part & version of the document. Some bundling is done here, but this is not meant to replace the analysis.

- architecture
- data model, data organization, data lifecycle
- digital object, data object, object properties, service objects
- registered data, real time data, dynamic data, time series data
- persistent identifier (PID), PID record, binding identifier, reference resolution,
- PID attributes (checksum, data reference, metadata reference, etc.)
- DOI, Handle, URL, URI
- repository
- bitstreams, instances of bitstream (copies, replicas), synchronization
- resource list, resource dumps, change lists, change dumps
- software stack (file system, database query, database management system, RoR flag, mutable flag, type, citation metadata, etc.)
- (canonical) access workflow, processing/curation workflows, provenance metadata
- metadata, key metadata, metadata object, metadata description, metadata catalogue, minimal metadata, property record, metadata attributes, discovery, context information, OAI-PMH, system metadata, descriptive metadata, citation metadata
- fixed schema, flexible schema, metamodel
- transaction record
- information objects, landing pages
- versions, presentation versions
- collections, data sets, aggregations, ORE resource map, container
- relations, RDF assertions, RDFS, OWL
- originator, depositor, users, archivist, curator, steward,
- API, protocol
- dissemination, content replication, content interpretation, content re-use
- data acquisition, data curation, data processing, data access
- identity, integrity, authenticity
- active data, active collections
- data publication, virtual collection
- policies, procedures