Towards an Open Repository Environment

Andreas Aschenbrenner
Goettingen State and University Library
Goettingen, Germany
aschenbrenner@sub.uni-goettingen.de

Tobias Blanke
King's College London
Centre for e-Research (CeRch)
London, UK
tobias.blanke@kcl.ac.uk

Marc W. Küster
Worms, University of Applied Sciences
Worms, Germany
kuester@fh-worms.de

Wolfgang Pempe
Goettingen State and University Library
Goettingen, Germany
pempe@sub.uni-goettingen.de

Abstract

Repositories used to be fairly monolithic systems, with a single object store, a tailored content model, and a dedicated application on top. First steps for searching across repositories (e.g. Z39.50, first drafted in 1988; OAI-PMH for content aggregation, first released 2001) are an exception to this. However, we are still far away from an "open repository environment", in which repositories interact on all levels with other agents (e.g. other repositories, added-value services, registries).

This paper creates a more fine-grained view on repository federation and analyses existing approaches by decomposing them into a physical, a logical, and a conceptual layer for both the object and the system. Among these attributes, the most evident gap pertains to interaction "patterns" between agents. In particular, the notification pattern is more immediate and directed than existing query and harvesting mechanisms, enabling new federation scenarios and laying the grounds for open repository environments. Prototypes of the concepts presented here are being implemented in the scope of the project Dariah, which establishes an e-Infrastructure for the humanities.

1. Introduction

The management of files is a core capability of data-driven science, just like the management of structured data in databases. However, while database systems have received extensive attention in theory and application, the management of semi- and unstructured data in file-based systems still lags somewhat behind. Isolated files are rarely sufficient for capturing and managing information and its context. Metadata are needed to describe the content (e.g. title, author, language, provenance, classification), administrative aspects (e.g. rights, licenses, retention schemes), and to create relations to other files (e.g. isVersionOf, isPartOf). The persistent management of networks of files and their associated metadata is achieved by systems called "*digital repositories*". An aggregation of all the parts that intellectually "belong" together (set of files, metadata, relations and behaviours) is called a "*digital object*" in this paper.

In the mid-90s, universities and research institutions established “institutional” repositories to manage the publications created by its members. [39] Currently emerging repositories have evolved far from these rather static archives containing homogeneous formats in various ways: They have grown to accommodate huge amounts of research data and to embed directly into the daily workflow of research environments. These repository-based environments are employed in a wide range of application scenarios including visualisation and analysis, preservation, and data re-use. Due to this breadth of applications and contexts, the need for federating diverse repositories, distributing repository components, and linking external added-value services into a repository environment is becoming ever larger. An information environment is emerging, in which – rather than monolithic repository systems sitting on dedicated hardware – repository functionalities are part of a larger, open environment of decentralised, collaborative agents (e.g. repositories, registries, re-representation and preservation services). However, existing federation protocols have not yet fully caught up with these developments and are often ill-equipped for supporting these applications.

This paper puts federation of distributed repositories into a broad perspective: The use cases presented in the following section 2 illustrate our notion of an ”open repository environment“. An analysis of existing repository federation approaches in section 3 unearths the attributes of interoperability to be accounted for. Section 4 continues from there and makes a case for pattern-driven design of repository environments and the importance of “*notification*“ patterns. Notification patterns address one attribute of interoperability that is neglected or entirely missing in current federation standards, and its capabilities when building open repository environments are exemplified in section 5, which presents a prototype that informs the development of the Dariah e-Humanities infrastructure [17]. While the use cases and prototypes in this paper are often written from the perspective of a research environment in the humanities, its findings are applicable more broadly in other disciplines and sectors as well.

2. Use Cases

In the introduction we posit that current federation repositories have not yet fully caught up with current requirements of repository environments, in which multiple agents interact in a decentralised manner. This section introduces some use cases, which show that federation mechanisms in such open repository environments is very unlike traditional federations (e.g. Driver/OpenAIRE [27, 64], Dare [56], Europeana [5]). As opposed to these federations, that are often based on the OAI-PMH harvesting protocol [33], an open repository environment (a) may deal with material that changes frequently and may need to propagate those changes swiftly, it (b) includes non-repository agents (e.g. format registries, migration services, visualisation of content networks), and (c) it allows interaction on multiple layers of abstraction (e.g. file, object, re-representation services). [16] As a basis for the analyses in this paper, the following use cases are prototypical and represent a whole class of conceivable use cases: operating on the content, on metadata, as well as on relations between objects.

2.1 Cross-Archive Content Analysis

Research is only in rare cases confined to a single location. Particularly in the humanities, research questions often include material from dispersed locations. Each of these locations may have a institutional repository of their own, from which digital resources are provided in the local format. In this use case, these distinct repositories join together to offer analysis across their collections without changing the underlying technologies.

As a real example, we assume the collaboration of the Oxford Text Archive (OTA) [46] with the e-Humanities infrastructure TextGrid [14]. Both currently offer their users – isolated from each other – XQuery-based analysis [19] of their XML/TEI holdings, and both employ the eXist XML database [43] for this analysis module and duplicate their XML/TEI data from the repository (the master files) into the eXist database (for analysis). A new, shared portal based could similarly offer XQuery-based analysis across all the available sources, while for the master files the OTA sticks to using the repository software Fedora [35] and TextGrid sticks to its approach based on D-Grid [44].

The usage of XQuery on TEI/XML in this use case is purely exemplary. The shared analysis portal could equally implement other mechanisms for analysis such as a SparQL-based approach [4], mining algorithms [24], or others. Essential for the federation mechanisms at the basis of such a shared portal are an immediate link between the repository and the portal to avoid inconsistencies between them, while maintaining the scalability of the overall system. Current federations based on the OAI-PMH protocol (cf. section 4) are restricted to metadata and hence fail to allow for deeper, content-oriented analysis. Also, existing search-engines that harvest the actual data from various sites are not geared to such specialised queries.

2.2 Task Tracking

One of the first steps in research activities often is the collation and preparation of the material to be addressed. This step may involve a variety of tasks, for multiple people, in dispersed locations. A typical research preparation phase in the humanities may involve an actual visit to an archive for a specific manuscript, digitisation of some selected pages, and eventually their transcription, mark-up, and annotation in a machine-readable format. Depending on the size of the project and the availability of the material, this process may take weeks or even years. [63]

Consequently, task management is essential for many collaborative projects, and the particular challenge in this use case pertains to its distributed nature, which may involve multiple independent repository systems. A system supporting task management in distributed teams monitors changes to the material in its distinct sources (e.g. newly incoming digitisations, updates to transcriptions), and allows researchers to annotate the state of the material and to distribute tasks among team members.

Initiatives currently employ a variety of generic software packages. Dedicated solutions are emerging for digitisation workflows [10] or as part of large editing systems [8]. However, we are not aware of solutions that span multiple sources. To enable the construction of such systems in the first place, federation mechanisms are needed to read the metadata of available material from various sources, to keep track of changes to those sources and material, and to integrate the material (without necessarily extracting it from its original source).

2.3 Resources Collaboration Networks

Research is increasingly collaborative, and interaction between researchers from the project team or external is high in all stages of the research process. [15] This trend towards collaboration occurred in parallel to the rise of collaborative software in web environments such as wikis, blogs, and others. [52] However, while collaborative software is being applied in research practice, it is still not adequately linked into research environments.

One conceivable linkage of collaborative software into research environments is inspired by Trackbacks [54], which link related blog entries. Trackback notifications are triggered by the bloggers themselves, and they span virtually all blog platforms. In a research environment, this approach could be extended to scholarly resources (i.e. data, tools, or workflows) and (1) allow notifications from blogs or other discussion media to resources (e.g. ”This blog entry discusses that piece of data”), (2) make users or other machines aware of new or updated resources, and (3) link citations in publications to the scholarly resources for re-use and verification. Thereby, notifications link repositories, tools, and essentially humans in living collaboration networks.

3. Attributes of Interoperability

Many mechanisms for federating repositories have been developed as grassroots initiatives and are de facto standards today. This section focuses on well-known approaches for repository federation. Towards the end of the section, we shall illustrate that federation works on different levels – a physical, a logical, and a conceptual –, which are equally important for successful federation.

Z39.50[1] for querying library catalogues has been around since 1988, and became a US standard in 1992 [13]. It describes syntax and semantics of a metadata query as well as the list of results. However, as it has been around since before the rise of the web, Z39.50 is not using HTTP as a transport layer but defines its own protocol, it is neither RESTful [28] nor loosely-coupled [47], and is hence in many ways not apt to the current web environment. Therefore, an initiative to find a more web-like successor specified the first version of SRU (Search/Retrieval via URL)[2] in 2002. Other than Z39.50, SRU uses HTTP, is RESTful and stateless, and can be used with current web browsers.

The Protocol for Metadata Harvesting, OAI-PMH,[3] [33] was first released in 2001 to connect disparate library catalogues. It is one of the most widely used federation protocol in the repository community to date. In January 2010, OAIster, a self-elected "union catalog" for digital resources,[4] cross-referenced more than 1100 contributors by way of the OAI-PMH protocol and their more than 23 million digital resources. However, OAI-PMH bears various caveats. The close link to Dublin Core[5] is often perceived as problematic. The rather loose definition of the pidgin metadata format Dublin Core leaves various possible interpretations open to the user. Various initiatives (e.g. [57, 62]) therefore found Dublin Core to be ill-suited for resource harvesting, since it "[...] does not possess sufficiently rigorous semantics to unambiguously express the information essential for resource harvesting" [62], and they set out to embed other metadata formats into OAI-PMH. Overall, this leads to the slightly paradox situation that while the OAI-PMH protocol itself is *rather* light-weight ([67, 41]), aggregation initiatives building on OAI-PMH are often quite large and centralised, and some of them enforce additional specifications on top of the OAI-PMH to be able to deal with the heterogeneous data from diverse OAI-PMH sources (e.g. Driver [27], Dare [56], Europeana [5]). Apart from the caveats of its usage in practice, the focus of OAI-PMH on harvesting only metadata is insufficient for many federation initiatives (e.g. [62, 58, 61]).

To address not only metadata but also actual content, the Open Archives Initiative released "Object Reuse and Exchange", OAI-ORE [36]. The focus of the standard are aggregations of (digital) resources, both simple as well as complex ones consisting of multiple distributed items (e.g. text, images, data, video) and relations to other resources [48]. OAI-ORE defines implementations for describing digital objects, e.g. RDF serializations in XML or by embedding RDF statements into the Atom Syndication Format [45]. Version 1.0 of OAI-ORE has been released in October 2008 and uptake is starting to gain speed. As the cousin of OAI-PMH, OAI-ORE has much attention guaranteed, and a series of release workshops has been widely attended. Thus, OAI-ORE can be expected to become an authoritative standard in the future. It must be recognized, however, that – unlike the PMH – ORE focuses on the data, yet does not specify how data are propagated between agents. Therefore, it is a format and does not fulfil all aspects of a fully-fledged protocol and federation mechanism.

But what is a federation mechanism actually composed of? What are the aspects needed to establish a pragmatic interaction between two agents in a repository environment? Answers to these questions are essential for further analysis of existing approaches. Therefore, before we continue, we decompose interactions for repository federation into three layers – the physical, the logical, and the conceptual layer (cf. Thibodeau [59]) – for both

[1]Z39.50. (Maintenance Agency: Library of Congress) http://www.loc.gov/z3950/

[2]SRU, Search/Retrieval via URL. (Maintenance Agency: Library of Congress) http://www.loc.gov/standards/sru/

[3]Open Archives Initiative, Protocol for Metadata Harvesting, OAI-PMH. http://www.openarchives.org/OAI/openarchivesprotocol.html

[4]OAIster. www.oaister.org

[5]Dublin Core Metadata Element Set. `http://dublincore.org/documents/dces/`

the object as well as the system. While these layers alone do not define a technical architecture, each of them requires different organizational and technical measures for their management; the model (cf. figure 1) hence supports separation of concerns, and fosters robustness and simplicity.

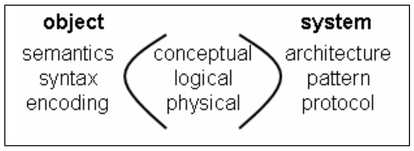


Figure 1. Attributes of Interoperability

- **encoding** (object) – defines the byte serialisation for characters, and is an essential basis for machine interaction. (Please note, Dublin Core also defines "syntax encoding" and "vocabulary encoding" [65], which we address in the following two items.)
- **syntax** (object) – specifies the strings and statements that can be used to express *semantics*. In compilers for programming languages this is often referred to as the lexical rules and the grammar of how statements can be expressed. For example, an XML-document – a prevalent syntax for describing digital objects – that complies with the lexical rules and XML markup grammar is called "well-formed".
- **semantics** (object) – define the meaning of terms and statements in a certain context, for example in a digital object. Semantics are shared, pre-established and negotiated between stakeholders, and expressed in vocabularies (flat lists) or ontologies (network of concepts and their relations). Due to the need for agreement on common semantics between stakeholders, "local" semantics tend to be more expressive than those of larger groups or "global" semantics. Other than *syntax*, which can be captured into a complete machine-readable specification, semantics may always be subject to human interpretation and may need informal definitions alongside the machine-readable ones.
- **protocol** (system) – describes within an information system how one intellectual entity relates to others, e.g. whether they are nested or dependent on each other. [34, 49] Containers such as METS [42] are structural tools to bind closely related entities together as in the case of a digital objects composed of multiple files. Looser relations are often expressed through references between objects that can be meaningful even across information systems.
- **pattern** (system) – identifies recurring design problems in information systems and presents a well-proven generic approach for its solution, consisting of the constituent components, their responsibilities and relationships. [12, 20] Patterns can be building blocks of system architecture, or define the way in which distinct information systems exchange information (e.g. triggers, workflows, conventions, timing).
- **architecture** (system) – specifies the overall structure, capabilities of and interactions between system components to achieve an overall goal. Architectures are tailored towards specific requirements in a specific context; although they may be based on a reference architecture that is relevant in a domain or recurrent application context.

Contracts for these four concepts are needed to enable the linkage of multiple technical environments into a single information system. These four concepts influence both, the overall information system as well as the individual object (cf. figure 1). Please note, that they are only an excerpt of all the aspects that a contract between unlike systems needs. All aspects from the individual bits on the fibre channel up to the pragmatics and user intentions that trigger an interaction in the first place are equally important in achieving interoperability. For the purposes of this paper, however, we focus on the four concepts and their definitions as presented above.

With regard to *syntax (object)*, the repository community has a long history in viewing metadata not as flat lists but rather as interrelated streams and containers. Streams allow multiple parallel metadata descriptions (with or without semantic mapping), whereas containers are hierarchical metadata modules for a single object, potentially with relations to other objects. [34, 49] Object syntax is often in XML with container formats like METS [42] and MPEG21-DIDL [1], as well as the RDF-based aggregation format OAI-ORE. Where objects are not stored as XML, it is simple to export them into a suitable format, since the basic concepts – (hierarchical) lists of metadata with relations to the actual content as well as other objects – remain stable across systems. Therefore, for both the physical and the logical level technical gateways can be established to enable interoperability, where they differ across distinct systems.

This is different when it comes to *semantics*. Vocabularies or ontologies for describing semantics can only be mapped onto each other, if they are describing the same concepts. Therefore, when mapping local standards onto each other, there is often loss of information involved where the standards do not entirely overlap. Consequently, there is no technically *correct* way of mapping semantics, but local compromises need to be found. There is a myriad of available standards for both metadata as well as more operational aspects like describing query strings and others, and adopting one of these standards supports interoperability. For metadata, "application profiles" provide a useful tool for re-using and mixing existing standards to foster *ad hoc* interoperability [29].

When transferring these findings for the *object* to the *system* layers, we see a gap with regards to *patterns*. While relevant for federation, the physical layer - encoding (object) and protocol (system) - is well researched and there are e.g. gateways between different encoding standards and programming libraries for protocols available. On the other hand, the conceptual level above - semantics (object) and architecture (system) - are closely linked to the very application context and hard to discuss on a generic level. Of particular interest to us hence is the logical layer. While there are relevant concepts and tools for syntax (object), we still lack them for patterns (system).

4. Federation Patterns

The previous section defined *patterns* to be generic approaches to recurring design problems. Figure 2 offers a language of federation patterns. These patterns synthesize existing approaches and put an emphasis on *notification*. All the patterns are described in the subsequent sections in detail.

Some of the challenges that are addressed by federation patterns to a different degree include

- **scalability** Efficiency in a federated environment is particularly dependent on the multiple, independent agents. Each additional agent raises the risk that the low performance of one agent impacts detrimentally on the overall performance of the whole federation.
- **consistency** As digital objects are duplicated and passed between independent agents, consistency issues may arise. Particularly in environments where objects change frequently, clients may hence be presented with old versions of an object or with processing results building on such old versions. Likewise, delays in the propagation of a newly added object through the federation may lead to an incomplete state at federated agents.
- **openness** We consider openness to be defining of repository federations that allow diverse, decentralised agents to interact *ad hoc* within a federation.
- **standard** Enabling openness and hence interoperability in decentralised agents calls for a minimum level of standardisation, or also the flexibility to embed standards with regard to syntax, structure, or semantics.

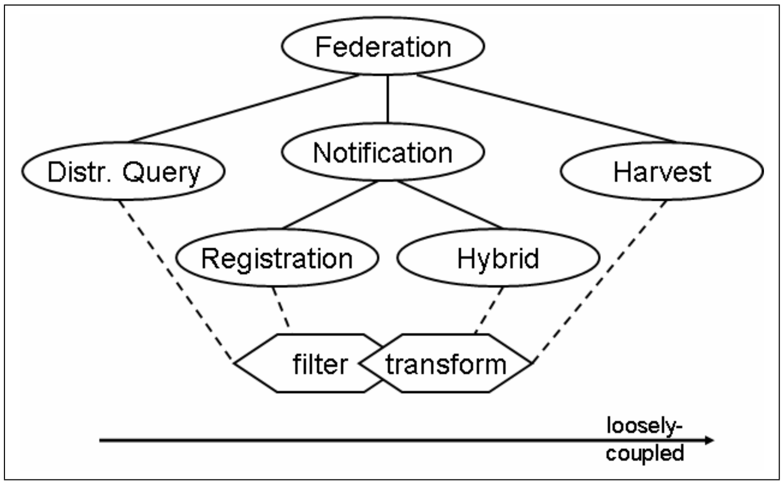
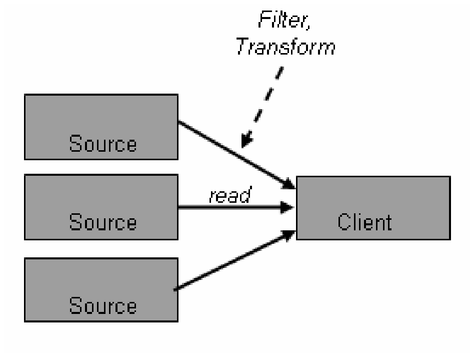


Figure 2. A pattern language for federation patterns. Mechanisms to filter and transform objects can be embedded into each pattern to raise the scalability and manageability of the federation.

4.1 Distributed Query

A Distributed Query essentially is the composition of multiple Client/Server interactions, as a query is sent to multiple sources and the responses are subsequently integrated into a single result set. The client must know all sources, and ideally the sources all provide a single standard interface for the query.



Application Context: A Distributed Query pattern is best used in a setting where objects in the disparate sources may change frequently and at any time. At the same time, however, the client wants to access the very latest object versions, and consistency problems between the various sources need to be avoided.

Another reason to opt for a Distributed Query pattern for repository federation may be technical constraints (e.g. large size) or legal restrictions, as the data remains at the source institution (other than in the case of Notification or Harvest patterns).

Forces: Even with dedicated server interfaces, Distributed Queries often display bad scalability. A Query is often dependent on the slowest server, when clients aim to integrate the various responses into a single result set. Thus, particularly in decentralised environments where clients have little influence on the source's quality of service, slow response times of some sources may be prohibitive for adequate results. Underlining this, the Resource Discovery Network (RDN) was finding that even with only "five subject gateways in its cross search there were problems of poor performance" [21].

♦♦♦

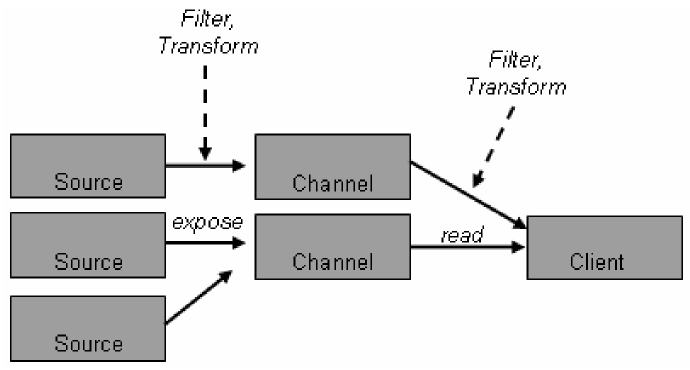
Exemplary Implementations: There are various implementations of the Distributed Query pattern in the repository community. Z39.50 for querying library catalogues has been around since 1988. Z39.50 was widely spread and still is, along with its successor SRU/W that is based on web services respectively REST.

One of the notable implementations in other communities is SDMX, the protocol for Statistical Data and Metadata eXchange supports federations that may span numerous organisations around the globe. [2] SDMX has chosen a Query pattern, since statistical data are often subject to licenses and cannot be hosted outside of the creator's organisational environment. Another characteristic in the statistical data domain, that makes Distributed Query the suitable pattern, are the rigid consistency requirements in the face of frequent update cycles.

4.2 Notification

In a Notification pattern, the source sends out messages on repository events. Triggers for notifications can be e.g. CrUD events – the creation, update, or deletion of an object in the repository –, which allows the client to stay in sync with the current state of the repository.

We distinguish between two sub-patterns of Notification: Notification by Registration, and a Hybrid Push/Poll Notification, which are described below. Both build on the availability of a message channel, which conveys the notifications from the source to the client.



Application Context: Notification is particularly suited for federation topologies where the agents are synchronised in their state, and need information about repository events as they occur. Once many independent agents need to be synchronised, a Notification pattern is more timely than Harvest, and more robust than a Distributed Query pattern by its direct, yet de-coupled communication between the source and the client.

Forces: A Notification pattern requires the setup of a suitable message channel where messages are actively exposed by the source. Particularly in approaches that are by Registration, the reliability of this channel is of key importance. Also and particularly in a Hybrid approach, the latency of transporting the message from source to client must be taken into account.

♦♦♦

Pattern Details: Notifications can be interpreted as the opposite of the Distributed Query mechanism. While in a Query the client requests information from a set of sources in a lower architectural layer, notifications are triggered by low-level events and passed on to higher level services. Notifications can e.g. be used to implement Observer patterns on CrUD events [20].

A Notification pattern builds on a message channel, and we distinguish broadly two approaches of how such a channel can be implemented. The first approach is "**by Registration**", with some messaging frameworks distinguishing between publish-subscribe (one-to-many) and point-to-point (one-to-one) models. [31] Both messaging models require an event mechanism that allows subscription in the publish-subscribe model (which delivers immediately on the occurrence of an event), or the creation of a dedicated queue in the point-to-point model (which delivers on consumption, and hence reliably delivers messages). Because of the registration and since the notifications are passed on without delay, this pattern is often used in more tightly-coupled environments.

In contrast to these registration-based notifications, **Hybrid push/poll** notifications (many-to-many) can be initiated without any communication between the agents and are hence more decoupled. Instead of the subscription process or a dedicated queue, consumers retrieve notifications from a broker. This broker may offer a notification history, such that a client can look up past notifications or it may be offline when a notification is sent and retrieve it later whenever convenient. This increased decoupling and robustness comes at the cost of immediacy, since the consumer needs to actively retrieve the notification, and hence the length of the poll cycle defines the delay.

Exemplary Implementations: Few repositories have adopted message-oriented middleware for coordinating repository-internal processes. Since version 3.0, Fedora implements the Java Messaging Service JMS [26]. Fedora sends notifications on all calls to its *API-M*, which includes CRUD operations on objects, datastreams, and relations. At the time of writing, DSpace is preparing a new event system for the release of its version 2.0. [11] A comprehensive implementation of messaging is also in place in the iRODS rules system that is triggered through events [50].

A tool for implementing a federation mechanism that spans diverse repository platforms and other agents could be the Atom protocol [32], fostering a Hybrid Notification pattern. Since Atom is an XML-based standard, it enables communication across heterogeneous agents with different software bases. Implementations of a Hybrid Notification pattern are considered viable even in a multi-agent environment where timing is an issue and hence frequent poll-cycles are required. [55, 38] Its embedding into the web architecture may be conducive to this, as conditional HTTP GET requests and common caching mechanisms in web proxies minimise the impact of short polling cycles by consumers.

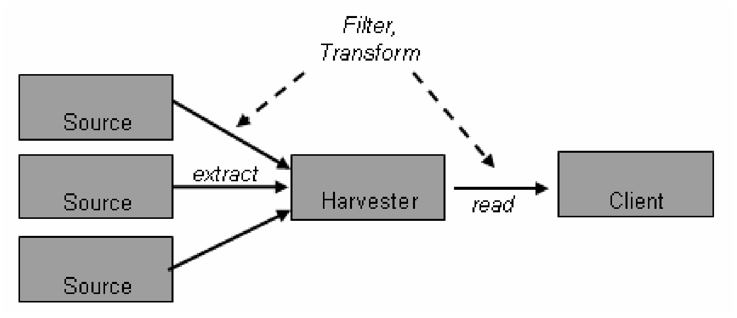
Yet, such an Atom-based Hybrid Notification pattern remains to be tested in a repository environment. In section 5 we suggest a prototype for that, which was developed in the framework of the Dariah e-Humanities infrastructure.

4.3 Harvest

An intermediary between source and client – the harvester – collects all the relevant data from disparate sources, and provides a single, integrated portal to the client. Regular harvest cycles ensure that the data gathered by the harvester remains up-to-date.

The harvest mechanisms may vary as to how the sources are identified, how often harvest cycles are performed, and whether a follow-up harvest cycle only updates changed data (iterative) or re-collects all the data regardless of whether or not it was updated (complete).

Application Context: The Harvest pattern de-couples the client from the server thereby scaling the communication in the federation down from multiple tiers to only two: the client and the harvester. This potentially improves the response time for clients considerably. Therefore, the Harvest pattern is suitable for decentralised environments, in which independent sources may not offer adequate quality of service with regard to their response time.



Furthermore, the Harvest pattern is best used in environments where digital objects change infrequently due to the potential data inconsistencies introduced by the Harvester.

Forces: The redundant storage of data may introduce inconsistencies to the original, which is further aggravated through infrequent updates. Infrequent updates, in turn, may be enforced on the overall system as harvest cycles potentially take considerable time, depending on the size of the federation, server response time, and the size and complexity of the digital objects involved.

♦♦♦

Pattern Details: Harvesters such as those for web search engines are well researched, and there are relevant experiences from this community. However, there are some differences to harvesting mechanisms in repository environments that we will focus on in the following.

With regard to the potential inconsistencies and the load on the harvester, as mentioned above, the key mechanism is data selection: which object should be downloaded, and when? There must be a mechanism for identifying objects in the first place, and in the following we present three conceivable mechanisms.

- Web search engines usually follow-up the links parsed out of the harvested data, thereby establishing a self-referencing network of web resources. This is not feasible in repository environments, which mostly lack such densely linked content.
- In another approach, the server brokers the data to the harvester. In one way to achieve this, the server passes the ID of the next object along with a harvested resource (a "resumption token"). However, this either introduces state between the server and the client which potentially affects the robustness of the system, or it may lead to inconsistencies if the list of objects changes during the harvesting cycle. [47]
- In an alternative approach, the repository or other object source needs to provide a list of its objects. The way such a list is provided may vary from merely a plain list, to a list with details about when the object was last updated, to a dynamic list that can be queried for specific object attributes including last update.

An additional impact on the overall efficiency of the system can be achieved by including information about the last update of an object and other metadata in the selection decision. Metadata about the last update may be useful, in case a harvester re-visits a source to only retrieve the objects that were updated since its last visit – iterative harvesting rather than complete harvesting rounds. More extensive filtering may be applied at this point of selection.

Exemplary Implementations: The Harvest pattern is well known in the repository community due to its implementation in OAI-PMH – probably the most prevalent federation mechanism today. We are not aware of any other implementation of the Harvest pattern or any other Federation pattern that is as widely spread.

OAI-PMH is geared at harvesting purely metadata, not the actual content of an object. However, the protocol has been employed in various contexts (e.g. [40, 22, 51]) and it has also been tweaked to harvest whole objects marked up in METS [58] or MPEG-DIDL [62]. One may argue though that these adaptations on OAI-PMH were mainly driven by the prevalence of OAI-PMH, not because OAI-PMH is really the most suitable technology for use cases other than metadata harvesting.

At the same time, we are not aware of any other significant implementation of the Harvest pattern. The low occurrence of alternative harvesting mechanisms to OAI-PMH in repository environments notwithstanding, it is quite simple to implement the Harvest pattern *ad hoc* using other existing mechanisms. For example, “sitemaps” [3] offer the crawlers of web search engines a standard entry point to the contents of web sites, and it could equally be used to expose repository contents for harvesting by repository services. Sitemaps also offers a *lastmod* field that encodes the object’s last modification date, to support iterative harvesting. In short, the Sitemaps-based harvesting shows that the Harvest pattern is a universal pattern that is not tied to OAI-PMH or any specific technology.

5 An Atom-based Repository Federation

After the previous sections outlined the context and concept of federation patterns based on notification, this section presents an actual environment with multiple repositories and other interacting agents. This environment will establish Dariah, a closely-knit, yet open repository infrastructure for the humanities.

Dariah is a project in the framework of the European Strategy Forum on Research Infrastructures (ESFRI) [23]. ESFRI projects are designed to offer the research community essential infrastructure for decades to come (e.g. telescopes, reactors, ships), and ESFRI projects are currently in their planning phase. For the humanities, Dariah builds a digital infrastructure to share cultural artefacts, re-use existing tools, and collaborate across institutions, cultures, and disciplinary boundaries. Partners in Dariah include humanities data archives, researchers in the humanities, as well as technical partners.

The goals for the Dariah repository infrastructure lie particularly in the combination of two characteristics: the repositories should remain independent, grow and evolve over time, and interact with other agents as described in the use cases of section 2 (hence *open*), while the contents in Dariah and tools provided through Dariah should be accessible to the researcher as if Dariah was a single platform (hence *closely-knit*). As such, Dariah invites institutions and researchers alike to contribute their own content and tools, and to conduct their active research with tools that build on the Dariah platform. Because of this, however, some of its resources may be of varying quality and some of them may be in an early stage of creation and, therefore, private [60]. This is both a challenge and an opportunity, and some aspects of the environment that deals with this and benefits from this, is described in the following sections.

5.1 Semantics in Dariah

Key attributes of repository federation environments include object semantics, and the structure of how repositories interact amongst each other and with other agents. The following description of semantics and structure further defines the context and requirements in Dariah, before we delve into technical implementation.

Linking diversity is at the core of Dariah’s philosophy. Disciplines in the humanities differ greatly with regard to their resources – their data, tools and methodologies. Moreover, innovation is sometimes associated with introducing variations into their data, tools, or methodologies, thereby reinforcing heterogeneity even within a single discipline. Through linking this diversity Dariah aims to build bridges, to enable researchers from different disciplines or cultural backgrounds to collaborate on the same material in a common research environment, and to share their diverse perspectives and methodologies.

A prerequisite to benefit from this opportunity, however, are interoperability and interaction between the diverse resources. But what level of interoperability can be achieved in the face of such a high level of diversity? Experiences from other repository environments such as Driver [37] indicate that – rather than enforcing rich metadata guidelines – encouraging participants to voluntarily provide quality is the more viable route to take. Therefore, initiatives like Europeana [5] and Dare [56] demand only few mandatory fields.

In addition to learning from repository federation initiatives like Europeana and Driver, Dariah also seeks interoperability with them and other associated environments. Both Europeana and Driver are building on standards by the Open Archives Initiative, notably OAI-ORE. While sticking to the same format is not a prerequisite to enabling interoperability since gateway services could translate between different formats, convergence may be useful to keep overall complexity down if viable for all participants.

In this way, Dariah is currently developing its approach to deal with diversity and facilitate interoperability among and beyond Dariah partners. However, the challenges for Dariah are redoubled, considering that in addition to the situation of research data in the humanities, it has to deal with another source of diversity: the diversity inherent in the agents participating in the Dariah environment. Other than the homogeneous repository environments of the aforementioned federation initiatives, Dariah aims to invite diverse agents including collaboration platforms, added-value services, private archives, service providers, and whoever wishes to contribute to an open research environment for the humanities.

In all these aspects, please note that “interoperability” is not related to “uniformity” in any way. The interoperability scheme may differ significantly from the internal metadata management at Dariah sites, as long as the rich internal data can be dumbed-down [66] to the interoperability scheme. Interoperability is necessary to expose and link individual objects, and to enable cross-repository applications where desired (e.g. cross-repository analysis portals, statistics, etc.). The metadata at the individual site is necessary to capture the scholarly context to enable re-use and verification even in the far future, which is a much larger task. Dariah is currently working on enabling an evolutionary metadata approach, where scholars can start with the bare minimum annotation and over time this is extended by the scholar who created the object, by other scholars, or automatically from the usage context of the object.

5.2 Exposing Repository Events

After looking at the context and semantics of Dariah in the previous section, this section describes how they are being exposed for federation in an architecture that is based on a hybrid push/poll notification pattern using Atom as described in section 4.2. An experiment that involves diverse system environments – TextGrid and iRODS/Fedora – vindicates the applicability of the concept for Dariah’s requirements, and that despite the heterogeneity of the actual systems, the concepts resemble each other closely.

Figure 3 depicts the overall design of the federation environment. Reading it from bottom up, it shows how repository events are recorded in an Atom feed. For this prototype, the Atom feed records all specified events in a history to ensure completeness[6]. Atom feed entries are very succinct and only contain the bare minimum metadata needed to sustain the federation[7], the references will then provide more detailed descriptions of the object in an appropriate format (e.g. MODS for resources, OAI-ORE for aggregations).

An Atom-based notification mechanism has been chosen, since Atom is a widely supported standard and Atom-based notification can be embedded into existing systems with minimal effort. The Dariah test environment for exposing CRUD events via Atom spans three different systems: TextGrid, iRODS, and Fedora.

[6]rather than e.g. updating old entries about objects whenever an action is performed on that object

[7]e.g. metadata about the event and date, the URI of the object content and metadata, and the format for filtering

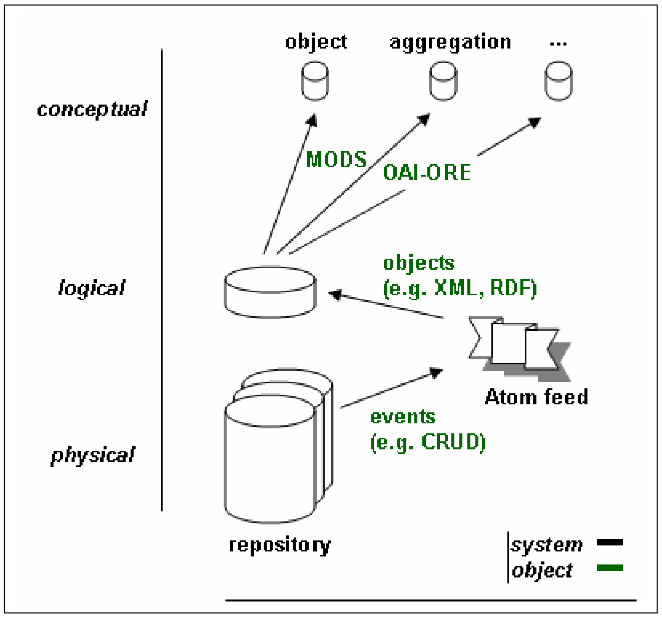


Figure 3. System landscape of the federation, with system components and object layers interacting

Even though the three systems are very unlike, the implementation for exposing repository events is conceptually very similar between them.

The TextGrid infrastructure for the humanities is based on the Globus grid middleware [9], and object management capabilities were tailored into this environment. Its CRUD interface for creating, reading, updating and deleting objects via REST or SOAP (*TG-crud*) conducts tailored functionalities such as normalising and indexing XML/TEI files on ingest. For the prototype we adapted TG-crud to post to an Atom server whenever an object is created, updated or deleted.

Both iRODS and Fedora offer internal event mechanisms which can be used for posting events to an Atom feed. Fedora's event mechanism is built on JMS [26]. For the proprietary event mechanism of iRODS this can be achieved by introducing a new microservice for posting Atom entries that is triggered on relevant iRODS rules. E.g. the following statement sketches this for a `create` event:

```
# actionDef | condition | workflow-chain | recovery-chain
acPostProcForCreate||msiExposeObjectCreationInAtom|nop
```

Event-based notification allows controlled observation of and a direct link to repository internals (cf. section 4). Targeted reactions to events hence offer good scalability, and through their immediacy – in comparison with e.g. harvesting – it raises overall consistency when various (internal) components or (external) agents are involved. Since the Dariah environment will prospectively consist of a continuously growing number of agents that may depend on each other's state, a notification pattern is a suitable architectural building block.

The Atom feeds containing the events of Dariah repositories can be consumed by decentralised agents, decentralised meaning that they only use the data available from the Atom feeds and do not plug into any proprietary repository interfaces. Such decentralised agents could hence be built by other initiatives consuming the event messages from various sources without the repositories being aware of them.

For the test environment, we experimented with replication of digital objects across repositories, as well as an index of all XML/TEI objects in the federation. The federate index provides an XQuery-based interface that researchers use to analyse their XML/TEI texts. Similar analysis portals are offered by both TextGrid as well as the Oxford Text Archive (cf. section 2.1, [16]). The push/poll service that reads the Atom feeds, filters and normalises relevant data, and forwards the data to the relevant agent (in this case, the XQuery index) was hand-crafted for this prototype. Existing tools that may be used for a production-ready system could be based on Jangle [53] or Isidorus [18], yet a hand-crafted service allowed more flexibility when prototyping.

5.3 Discussing the Dariah federation

This section has introduced one of the building blocks for the Dariah research infrastructure for the humanities, the diversity of its collections and the vision of an open environment of decentralised agents. Its design manages diversity, encourages participation and links resources wherever feasible and desired. Just like it accommodates heterogeneity for digital objects, it aims to do so for services and applications as well. As part of this, it provides entry-points for external agents to link in and contribute new approaches to the Dariah environment.

To ensure coherence among emerging decentralised agents as well as in communication with related initiatives, the Dariah federation builds on an Atom-based notification pattern as one of its key design concepts. We have shown an experiment that links TextGrid, an iRODS and a Fedora test server into a single federation, and discussed the potential of the new federation mechanism for advanced analysis portals, collaboration, as well as a host of repository agents and novel functionalities to come.

Atom-based notification is lightweight and deeply embedded into the web environment of HTTP-based, ReSTful services. Web standards offer a gateway into a multitude of available tools, and the foundations of the web also raise the scalability of Dariah as a whole (e.g. through web proxies and Conditional HTTP GET's on the HTTP header 'If-Modified-Since') [55, 18].

However, work on the experiment is not yet completed. Questions pertaining to user and rights management, as well as persistent identifiers have already been alluded to above. Other questions may come up as the system and its constituent parts are monitored over time. For example, with regard to proxy usage, filtering of event notifications remains to be further explored. Filtered feeds would allow the client to e.g. only be notified on specific CRUD operations, or allow for filtering of object metadata (e.g. only notify on Creates or Updates to texts by Shakespeare).

6 Conclusions

Decentralised information environments are emerging, in which a repository is but one agent among a multitude of others. In such an environment, to name just some of the conceivable scenarios, parts of a single digital object are spread over various repositories (e.g. e-Publications [6]), repositories depend upon external re-representation and preservation services [30], and storage is out-sourced to dedicated infrastructure [7].

This paper identified distinct attributes of repository federations in order to guide analysis and implementation in the field. We hope that the perspectives on repository federations presented shows how the (semantic) interoperability of objects and systems interact, and helps to disentangle the lines between tools/standards (e.g. OAI-PMH) and designs (e.g. an architecture based on a harvest pattern) in federation systems. The distinction between the Query, Harvest and Notification patterns has been implicit knowledge in the repository community, yet the explicit analysis of each pattern presented here may be a useful tool when designing a repository federation, and it may equip architects with more flexibility with regard to possible system environments and application scenarios.

Section 2 in the beginning of this paper mentioned three scenarios of an open repository environment. The diversity of these scenarios indicates the opportunities in open repository environments that federate repositories and other agents (e.g. registries, added-value services, applications):

1. **Cross-Archive Content Analysis** – Portals that offer specialised analyses of the content available in a repository federation can be built on both a harvesting or a notification pattern. Rather resource-intense analyses that may take days to compute (e.g. automatic document classification, specialised visualisations) may opt for a harvesting pattern, whereas more immediate, ”live“ analysis can be realised through a notification pattern. The Dariah prototype (cf. section 5) introduced a notification-based analysis portal and its advantages of (almost) immediate updates and increased consistency across the federation.

2. **Task Tracking** – Tracking and associating objects as they occur in any of the repositories requires a high-level of immediacy and consistency across the federation. Therefore, notification may be the suitable pattern of choice: The Harvesting pattern is inadequate with regard to immediacy; and while a Query pattern would offer the necessary level of immediacy, the repeated requests for displaying the various tasks and charts may be detrimental to overall system performance. Please note, that in this case the notification pattern only operates on the metadata.

3. **Resources Collaboration Networks** – Eventually, scenario 2.3 describes a mechanism based on the Trackback protocol for linking all kinds of resources (e.g. objects hosted by a trusted repository, blogs, publications and research data). It basically only transports the references between resources.

Rather than convergence to a small set of concepts and systems, we are expecting diversity and decentralisation to increase in the repository community. The growth of the community to include various application scenarios from data-driven research to enterprise systems; the growing demands of these systems; as well as new players and the integration of new technologies (e.g. clouds as in DuraCloud [25]) in the field seem to point that way. However, this growth and diversity does not necessarily lead to disintegration of and redundancies within the repository community. Eventually, we hope that the framework developed in this paper is one of the pebbles contributing to a coherent discussion while furthering the diversity of approaches and technologies in the field.

7 Acknowledgements

This paper builds extensively on the work previously done in the digital libraries, digital repositories, e-Science, and adjacent fields. This work in progress is partly funded through the “Preparing Dariah” project under the scheme of the ESFRI Roadmap (European Strategy Forum for Research Infrastructures), yet it benefits from discussions beyond the project perimeter.

References

[1] *MPEG-21, Information Technology, Multimedia Framework; Part 2: Digital Item Declaration*, ISO/IEC 21000-2:2003, March 2003.

[2] *SDMX User Guide*, Statistical Data and Metadata Exchange Initiative, January 2007.

[3] *Sitemaps XML format*, Format Specification, February 2008, `http://www.sitemaps.org/protocol.php`.

[4] *SPARQL Query Language for RDF*, W3C Recommendation, 15 January 2008, `http://www.w3.org/TR/rdf-sparql-query/`.

[5] *Specification for the Europeana Semantic Elements, version 3.1*, Report, February 2009.

[6] *Enhanced Publications*, driver Report (Digital Repository Infrastructure Vision for European Research), Viewed August 2009, `http://www.driver-repository.eu/Enhanced-Publications.html`.

[7] *Fedorazon: The Cloud Repository*, JISC Project, Viewed August 2009, `http://www.ukoln.ac.uk/repositories/digirep/index/Fedorazon`.

[8] *Forschungsnetzwerk und Datenbanksystem (FuD)*, Project Website, Viewed August 2009, `http://fud.uni-trier.de/`.

[9] *Globus Grid Toolkit*, Project Website, Viewed August 2009, `www.globus.org`.

[10] *Goobi – Digital Library Modules*, Project Website, Viewed August 2009, `http://goobi.sub.uni-goettingen.de/`.

[11] *Prototype Implementation of New Event System*, DSpace Development Portal, Viewed August 2009, `http://wiki.dspace.org/index.php/EventSystemPrototype`.

[12] Christopher Alexander, Sara Ishikawa, and Murray Silverstein, *A pattern language : towns, buildings, construction*, Oxford University Press, 1977.

[13] American National Standards Institute (ed.), *Application Service Definition and Protocol Specification for Open Systems Interconnection*, NISO Press, Bethesda, Maryland, U.S.A., 1992.

[14] Andreas Aschenbrenner, *Editing, analyzing, annotating, publishing: TextGrid takes the a, b, c to D-Grid*, iSGTW **54** (2008).

[15] Andreas Aschenbrenner, Tobias Blanke, Stuart Dunn, Martina Kerzel, Andrea Rapp, and Andrea Zielinski, *Von e-Science zu e-Humanities Digital vernetzte Wissenschaft als neuer Arbeits- und Kreativbereich fr Kunst und Kultur*, Bibliothek, Forschung und Praxis **31** (2007), no. 1, `http://www.bibliothek-saur.de/2007_1/011-021.pdf`.

[16] Andreas Aschenbrenner, Tobias Blanke, David Flanders, Mark Hedges, and Ben O'Steen, *The Future of Repositories? - Patterns for (Cross-)Repository Architectures*, D-Lib Magazine **14** (2008), no. 11/12.

[17] Andreas Aschenbrenner, Tobias Blanke, Eric Haswell, and Mark Hedges, *The DARIAH e-Infrastructure*, Zero-In Magazin **3** (2009).

[18] Andreas Aschenbrenner, Marc Wilhelm Küster, Christoph Ludwig, and Thorsten Vitt, *Open eHumanities Digital Ecosystems and the Role of Resource Registries*, Proceedings of the IEEE Digital Ecosystems and Technologies Conference (DEST) 2009 (Istanbul, Turkey), June 2009, In Print.

[19] Lou Burnard, Katherine O'Brien O'Keeffe, and John Unsworth (eds.), *Electronic Textual Editing*, ch. Storage, Retrieval, and Rendering [Sebastian Rahtz], Modern Language Association of America, September 2006.

[20] Frank Buschmann, Kevlin Henney, and Douglas C. Schmidt, *Pattern-Oriented Software Architecture – A Pattern Language for Distributed Computing*, Software Design Patterns, vol. 4, John Wiley & Sons Ltd., 2007.

[21] Leona Carpenter, *OAI for Beginners – the Open Archives Forum online tutorial*, 2003.

[22] Churngwei Chu, Walter E. Baskin, Juliet Z. Pao, and Michael L. Nelson, *Oai-pmh architecture for the nasa langley research center atmospheric science data center*, ECDL (Julio Gonzalo, Costantino Thanos, M. Felisa Verdejo, and Rafael C. Carrasco, eds.), Lecture Notes in Computer Science, vol. 4172, Springer, 2006, pp. 524–527.

[23] European Strategy Forum on Research Infrastructures (ESFRI), *European Roadmap on Research Infrastructures*, 2006, http://cordis.europa.eu/esfri/roadmap.htm.

[24] Weiguo Fan, Linda Wallace, Stephanie Rich, and Zhongju Zhang, *Tapping the power of text mining*, Commun. ACM **49** (2006), no. 9, 76–82.

[25] Fedora Commons and DSpace Federation, *DuraSpace – Trust and reliability in the Cloud*, Midterm Report to the Mellon Foundation, February 2009.

[26] *Fedora Messaging Guide*, Fedora Commons Report, Viewed August 2009, http://www.fedora-commons.org/documentation/3.0/userdocs/server/messaging/index.html.

[27] Martin Feijen, Wolfram Horstmann, Paolo Manghi, Mary Robinson, and Rosemary Russell, *DRIVER: Building the Network for Accessing Digital Repositories across Europe*, Ariadne **53** (2007).

[28] Roy Thomas Fielding, *Architectural Styles and the Design of Network-based Software Architectures*, Ph.D. thesis, University of California, Irvine, 2000.

[29] Rachel Heery and Manjula Patel, *Application profiles: mixing and matching metadata schemas*, Ariadne **25** (2005), http://www.ariadne.ac.uk/issue25/app-profiles/.

[30] Steve Hitchcock, Tim Brody, Jessie M.N. Hey, and Leslie Carr, *Digital Preservation Service Provider Models for Institutional Repositories – Towards Distributed Services*, D-Lib Magazine (2007).

[31] Gregor Hohpe, *SOA Patterns - New Insights or Recycled Knowledge?*, Whitepaper, May 2007, http://www.eaipatterns.com/docs/SoaPatterns.pdf.

[32] IETF, *The Atom Publishing Protocol*, RFC 5023, October 2007.

[33] Open Archives Initiative, *The Open Archives Initiative Protocol for Metadata Harvesting*, OAI Protocol, Version 2.0, 2002.

[34] Carl Lagoze, *The Warwick Framework – A Container Architecture for Diverse Sets of Metadata*, D-Lib Magazine (1996).

[35] Carl Lagoze, Sandy Payette, Edwin Shin, and Chris Wilper, *Fedora: an architecture for complex objects and their relationships*, Int. J. Digit. Libr. **6** (2006), no. 2, 124–138.

[36] Carl Lagoze and Herbert Van de Sompel, *Open Archives Initiative – Object Re-Use and Exchange*, Presentation at JCDL 2007, June 20 2007.

[37] Norbert Lossau and Dale Peters, *DRIVER: Building a Sustainable Infrastructure of European Scientific Repositories*, Liber Quarterly **18** (2008), no. 3/4, 437–448.

[38] Heiko Ludwig, Jim Laredo, Kamal Bhattacharya, Liliana Pasquale, and Bruno Wassermann, *REST-Based Management of Loosely Coupled Services*, Proceedings of the International World Wide Web Conference (WWW2009) (Madrid, Spain), April 20-24 2009.

[39] Clifford A. Lynch, *The Transformation of Scholarly Communication and the Role of the Library in the Age of Networked Information*, The Serials Librarian **23** (1993), no. 3, 5–20.

[40] Liz Lyon, Rachel Heery, Monica Duke, Simon J. Coles, Jeremy G. Frey, Michael B. Hursthouse, Leslie A. Carr, and Christopher J. Gutteridge, *eBank UK: linking research data, scholarly communication and learning*, Proceedings of the UK e-Science All Hands Conference, Engineering and Physical Sciences Research Council, 2004, pp. 711–719.

[41] K. Maly, M. Nelson, M. Zubair, A. Amrou, S. Kothamasa, L. Wang, and R. Luce, *Light-weight communal digital libraries*, Proceedings of the 2004 Joint ACM/IEEE Conference on Digital Libraries, June 2004, pp. 237 – 238.

[42] Jerome Mcdonough, *METS: standardized encoding for digital library objects*, International Journal on Digital Libraries, no. 2, 148–158.

[43] Wolfgang Meier, *eXist: An Open Source Native XML Database*, Web, Web-Services, and Database Systems, LNCS, vol. 2593, Springer-Verlag, 2003, pp. 169–183.

[44] Heike Neuroth, Martina Kerzel, and Wolfgang Gentzsch (eds.), *German Grid Initiative D-Grid*, Universitätsverlag Göttingen, September 2007.

[45] M. Nottingham and R. Sayre, *The Atom Syndication Format*, IETF RFC 4287, December 2005, `http://www.ietf.org/rfc/rfc4287.txt`.

[46] Benjamin O'Steen, *The architecture of Oxford University Research Archive*, Proceedings of the Third International Conference on OpenRepositories (Southampton, United Kingdom), 1-4 April 2008.

[47] Cesare Pautasso and Erik Wilde, *Why is the Web Loosely Coupled? A Multi-Faceted Metric for Service Design*, Proceedings of the 18th International World Wide Web Conference (Madrid, Spain), ACM Press, April 2009, pp. 911–920.

[48] Alberto Pepe, Matthew Mayernik, Christine L. Borgman, and Herbert Van de Sompel, *From Artifacts to Aggregations: Modeling Scientific Life Cycles on the Semantic Web*, Journal of the American Society for Information Science (2009), `http://arxiv.org/ftp/arxiv/papers/0906/0906.2549.pdf`.

[49] Andy Powell, Mikael Nilsson, Ambjörn Naeve, Pete Johnston, and Thomas Baker, *DCMI Abstract Model*, Tech. report, June 2007.

[50] A. Rajasekar, M. Wan, R. Moore, and W. Schroeder, *A Prototype Rule-based Distributed Data Management System*, HPDC workshop on "Next Generation Distributed Data Management", May 2006.

[51] Uwe Schindler, Benny Bruer, and Michael Diepenbroek, *Data information service based on open archives initiative protocols and apache lucene*, Proceedings of the German e-Science Conference (GES) (Baden-Baden, Germany), Max-Planck Society, 2007.

[52] Clay Shirky, *Here Comes Everybody: The Power of Organizing Without Organizations*, Penguin Press HC, 2008.

[53] Ross Singer, *JANGLE – Just Another Next Generation Library Environment*, Library Geeks, Podcast 013, November 2008, http://onebiglibrary.net/geeks/episode/013-jangle.

[54] Six Apart, *TrackBack Technical Specification*, Version 1.2, August 1 2004, `http://www.sixapart.com/pronet/docs/trackback_spec`.

[55] David Slik, *Bycast's Cloud Storage HTTP API*, Presented at the SNIA Cloud Storage Group Meeting, May 2009.

[56] Stichting SURF, *DARE use of Dublin Core, version 2.0*, Report, December 2004.

[57] Friedrich Summann and Norbert Lossau, *Search Engine Technology and Digital Libraries – Moving from Theory to Practice*, D-Lib Magazine **10** (2004), no. 9.

[58] Robert Tansley, *Building a Distributed, Standards-based Repository Federation – The China Digital Museum Project*, D-Lib Magazine **12** (2006), no. 7/8.

[59] Kenneth Thibodeau, *Overview of Technological Approaches to Digital Preservation and Challenges in Coming Years*, CLIR Report 107, 2002, `http://www.clir.org/pubs/reports/pub107/thibodeau.html`.

[60] Andrew Treloar and Cathrine Harboe-Ree, *Data management and the curation continuum. how the monash experience is informing repository relationships*, Proceedings of the VALA2008 14th Biennial Conference (Melbourne), 2008, `http://www.valaconf.org.au/vala2008/papers2008/111_Treloar_Final.pdf`.

[61] Herbert Van de Sompel, Jeroen Bekaert, Xiaoming Liu, Lyudmila Balakireva, and Thorsten Schwander, *aDORe: a modular, standards-based Digital Object Repository*, The Computer Journal (2005).

[62] Herbert Van de Sompel, Michael L. Nelson, Carl Lagoze, and Simeon Warner, *Resource Harvesting within the OAI-PMH Framework*, D-Lib Magazine **10** (2004), no. 12.

[63] Virtual Vellum, *Final Report*, JISC Project, February 27 2007, `http://www.ahessc.ac.uk/files/active/0/VV-report.pdf`.

[64] Danielle Venton, *OpenAIRE: archive access anytime, anywhere* , iSGTW **25** (2009), `http://www.isgtw.org/?pid=1002201`.

[65] Mary S. Woodley, *DCMI Glossary*, April 2004, `http://dublincore.org/documents/usageguide/glossary.shtml#E`.

[66] Mary S. Woodley, Gail Clement, and Pete Winn, *DCMI Glossary*, Dublin Core Metadata Initiative, November 2005, `http://dublincore.org/documents/usageguide/glossary.shtml`.

[67] Xiaorong Xiang and Eric Lease Morgan, *Exploiting Light-weight Protocols and Open Source Tools to Implement Digital Library Collections and Services*, D-Lib Magazine **11** (2005), no. 10.