Georg-August-Universität
Göttingen
Zentrum für Informatik

Reference Framework for Distributed Repositories – Towards an Open Repository Environment

Dissertation
zur Erlangung des
mathematisch-naturwissenschaftlichen Doktorgrades
"Doctor rerum naturalium"
der Georg-August-Universität Göttingen

vorgelegt von
Andreas Aschenbrenner
Wien, Österreich

Göttingen 2009

Referent: Prof.Dr. Bernhard Neumair
Korreferent: Prof.Dr. Thomas Breuel
Tag der mündlichen Prüfung: 25. November 2009

Abstract

Our notion of IT infrastructure is changing dramatically. We used to be happy with just a power plug for our desktops – today, many of us can hardly imagine a day without Internet access. Data centres provide storage and other dedicated services to large organizations – today, anybody can access such services in the cloud. Grid technologies offer a complex environment for compute-intense applications . . . however, we still lack a simple, yet trustworthy environment for curating, semantically modelling, and sharing file-based information.

This thesis develops the foundational concepts for such an environment, and it builds on a class of systems stemming from scholarly communications called "digital repositories". Rather than building a new system, we analyse the *interoperability channels* between existing repositories, added-value services, and other agents. The Open Repository Environments emerging from these gateways are *evolutionary* in that repository components can change over time; *anarchic* to the point where they invite decentralised agents to participate; and a *bridge* between existing infrastructures. The infrastructures that were of particular influence to this thesis are e-Infrastructure for virtualising heterogeneous resources, as well as repository technologies for managing reference networks of file-based information, namely "digital objects".

Like any software architecture, the design of Open Repository Environments is a creative process that may be inspired by existing models and experiences, and this thesis creates a framework for such models. Its repository reference architecture derives the minimum consensus on what repositories constitute from an extensive analysis of existing experiences, and specifies two interfaces through which interoperability channels can be established: the Open Storage Interface and the Federation Interface. Subsequently, we present results from a detailed analysis of the functionalities and design criteria for an Open Storage Interface. On the other hand, Federation is a continuum of attributes and techniques rather than a single interface. In our analysis of Federation, we describe this continuum, identify existing federation patterns, and develop a novel Notification-based federation mechanism – probably the first such attempt since the creation of the OAI Protocol for Metadata Harvesting in 2001.

This thesis builds on a number of linked activities that were mostly conducted between 2001 and 2009, and in participation of six national and international research projects in the field. These activities combine analysis and discussion, notably in a series of workshops we organised in Europe and the US to bridge the e-Infrastructure, preservation and repository communities, as well as the architectural design and prototyping of the e-Infrastructure initiatives TextGrid and Dariah based on the findings presented in this thesis.

Kurzfassung

Unsere Erwartungen an IT Infrastruktur ändern sich rasant. Vor nicht allzu langer Zeit transportierten wir Daten im Kilobyte-Bereich über einen Stapel Floppy-Disketten, während heutzutage für viele ein Tag ohne Internet schwer vorstellbar ist. Eine professionelle Speicherinfrastruktur konnten sich bis vor kurzem nur große Organisationen leisten, bis das Aufkommen von Cloud-Anbietern jedem Nutzer mit einem Internetanschluss diese Möglichkeit eröffnete. Schließlich ermöglichen hoch-komplexe Grid-Technologien die verteilte Berechnung von z.B. aufwändigen Simulationen, etc. ... aber es existiert immer noch keine einfach integrierbare, vertrauenswürdige Umgebung zur Archivierung, semantischen Modellierung und Nachnutzung von Datei-basierten Objekten wie z.B. Simulationsergebnissen.

Diese Arbeit entwickelt das Fundament für so eine Umgebung, auf Basis von Erfahrungen aus den Repositorien- und e-Infrastructure Communities. Es ist eine Umgebung, die nicht aus einer spezifischen Software besteht, sondern vielmehr aus der Vernetzung von existierenden Repositorien, externer Dienste, und anderer Module entsteht. Durch diese Interoperabilitätsmechanismen erschafft diese Arbeit "offene Repositorien-Umgebungen", in denen sich einzelne Komponenten unabhängig von einander entwickeln, dezentrale Module untereinander interagieren, und sich unterschiedliche Infrastrukturen mit einander vernetzen. Als solches schöpft diese Arbeit vor allem aus Technologien zur semantischen Modellierung von digitalen Objekten und der Virtualisierung vorhandener Hardware-Ressourcen in Grid-Infrastrukturen, und sieht offene Repositorien-Umgebungen an der Schnittstelle dieser Infrastrukturen.

Systemarchitektur und damit auch der Aufbau von Repositorien ist ein kreativer Prozess, inspiriert durch existierende Erfahrungen und Modelle. Diese Arbeit sammelt existierende Erfahrungen und erstellt Modelle, die diesen Prozess strukturieren und unterstützen. Durch die Entwicklung einer Community-übergreifenden Referenzarchitektur für Repositorien können speziell zwei Interoperabilitätskanäle identifiziert werden: das Open Storage Interface und die Föderationsumgebung. Die Arbeit analysiert detailliert die Eigenschaften und Designkriterien für Open Storage

Interfaces. Weiters strukturiert sie die Attribute und Techniken für Föderationsumgebungen, und entwickelt Entwurfsmuster für unterschiedliche Arten von Föderationen. Teil davon ist die Entwicklung eines neuartigen Mechanismus zur Föderation auf Basis von "Event-basierten Notifications" – vielleicht der erste neuartige Ansatz seit der Entwicklung des OAI Protokolls für Metadata Harvesting (OAI-PMH) im Jahr 2001.

Die wissenschaftliche Basis dieser Arbeit wurde abwechselnd in der Praxis nationaler und internationaler Infrastruktur-Projekte, sowie in Analyse und Diskussion mit internationalen Experten in dem Gebiet gesammelt. Im Speziellen verknüpfen diese Aktivitäten die Bereiche e-Infrastruktur mit nachhaltigem Datenmanagement und -pflege.

Contents

1 Introduction

In 1995 Robert Kahn and Robert Wilensky [207] described a path towards an open environment of distributed digital information services. Their paper triggered the creation of a whole new class of systems, "digital repositories" [52, 92, 257]. Similar systems also emerged in other communities unaware of the "Kahn/Wilensky Framework" (cf. section 2.1). While a lot of ground has been covered since the Framework was written in 1995 and we have advanced beyond it in many ways, current information environments still lag behind the Framework's vision in terms of openness and distribution [245, 289, 291, 310]. This thesis argues that the tremendous evolution repository systems underwent in the last decade still fails to cover the opportunities lost through the lack of openness and distribution in repository-based information environments. Like the multi-faceted approach of the Kahn/Wilensky Framework, the approach to "open up" current repository-based information environments presented in this thesis consists of a combination of interoperability mechanisms geared to interweave repository components and other "agents".

Before going deeper into what we consider to be an *open repository environment* and how we can get there, however, this introduction looks at what digital repositories are and the current state of research and practice. Section 1.1 then pushes beyond the state of the art and presents scenarios that cannot be fulfilled by today's technologies, but for which the open repositories reference framework developed in this thesis provides a generic solution.

Information is rarely self-contained, neither in structure nor in semantics. A digital

text, for example, often contains images or other embedded media, it may consist of template files, or may disaggregate into multiple files for other reasons. The same applies for an image gallery with textual descriptions, simulations with parameter files, or other intellectual entities. Capturing these dependencies, "*digital objects*" can be anything from a simple file to a complex object composed of multiple files, metadata, and relations to other objects and functionalities. Likewise, digital objects are often embedded in a semantic context. One object may be a version of another object, various objects may have been created in the same research or business process, or they may contain related content. The networks formed through these and other relations establish a rich semantic environment that may span several systems.

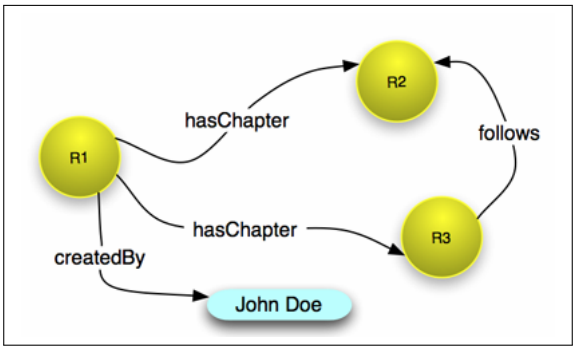


Figure 1.1: Reference graph of a digital object, taken from the OAI-ORE standard that defines an RDF-oriented object format [222]. This figure shows the object R1, consisting of two files R2 and R3, as well as other values (the author) and relations between them.

Digital objects are thus more than localised bit sequences. They are nodes in a semantic network; information nuggets re-usable in various contexts; carriers of functionalities and the ingredients of workflows. [130, 222] The systems managing digital objects in all these facets are called *digital repositories* in this thesis.

The emergence of repository-like systems to manage digital objects as described above can be traced in various communities, although they differ along various lines such as their requirements, technologies, as well as their terminology. As one of these communities, libraries and generally scholarly communications are particularly active in digital repository research and they also founded the much attended OpenRepositories conference series, which is why we call them the "repository community" in the following. Their focus is on the preservation [317] and dissemination of publications,

with e.g. university libraries up until now mainly focusing on the intellectual output of their institutions [233] and national libraries on the cultural heritage of their countries [92]. However, repository systems have emerged in other communities as well. In the context of scholarly research, repositories capture the raw data, derived analyses and other products in the research process to support ongoing research, to facilitate re-use by other teams, and to enable validation of research results [6, 8, 75, 172, 173, 248]. Various other communities and system types (repository-like or related) are mentioned later in this thesis (cf. particularly chapter 2.1).

These communities do not have a common understanding on what repositories constitute. To be more precise, they fail to agree among themselves about the key features of repository systems, let alone between each other. [48, 183, 277, 295] Some argue that the term "repository" does not adequately capture the capabilities of these systems [48], and also that it leads to confusion with other, unrelated technologies (e.g. "objects" in object-oriented programming, "repositories" as in software code and package management, or various other meanings). Nevertheless, this terminology is the currently dominant one.

This thesis does not attempt to follow suit in failed attempts to find a one-sentence definition for tasks and context of digital repositories. Instead, it remains purposefully on an abstract level of repository definition that resonates with a multiplicity of communities and systems. In other words, rather than putting definitions and constraints on individual repository systems to foster homogeneity, this paper on the contrary fosters heterogeneity, while enabling interoperability between diverse repository systems and other *agents* in an open repository environment.

Already the Kahn/Wilensky Framework considers repositories to be merely one *agent* in an environment of information producers, infrastructure services (e.g. identification services, authentication), consumers and added-value services (e.g. identification services, registries, visualisation tools), and others. In order to facilitate interoperability between those agents and amongst distinct repositories, the Framework tentatively defines a Repository Access Protocol (RAP) for (1) accessing objects, (2) depositing objects, and (3) accessing reference services for exploring repository contents. [207] Since then, RAP-like protocols have emerged and, indeed, along with them a multitude

of standard and non-standard data formats, metadata models, protocols, and conventions. [47, 58, 59, 199]

However, these mechanisms for interoperability have been created in an uncoordinated manner and are in many ways incomplete and fragmented. Protocols cluster in the three areas of the RAP – object access, object deposit, and metadata query –, yet other areas remain neglected. The repository community identified this issue e.g. in [18] and calls it the proliferation of *repository islands*, where each island fails to communicate with others, thereby duplicating repository management efforts and reducing contents, features and eventually the benefit for repository users. [245] point out that available protocols are insufficient and only serve "to make each 'island' of data easier to access". Real federation of distributed repositories and decentralised collaboration in an open repository environment remain an unfulfilled vision, despite or as underlined by the various attempts towards that goal. [66, 220, 263]

To re-iterate our notion of an *open repository environment*, "open" in this context refers to extensibility even for external agents, such that new modules can be linked up with existing ones and old ones can be exchanged in an evolutionary manner; new players can join the environment, possibly without knowledge of the originators; and previously unanticipated functionalities can be retrofitted into the environment. In this sense, "open" does not necessarily resonate with the notions of "open source" [270] and "open access" [153]. Components in an open repository environment can be both, either open source or proprietary. Contrary to making all repository contents open access available, emerging repository-based virtual research environments [65] also address management of research data before their publication, and they necessitate private research spaces for individuals and research teams [307].

Brad Wheeler, discussing in [320] institutional e-Research and IT governance, asks what should be part of generic infrastructure and what should be handled on an application-level (cf. figure 1.2). Many of the functionalities he suggests to categorize as 'infrastructure' – including curation, metadata, search and retrieve – are key capabilities of repositories. Inspired by Brad Wheeler's discussion, we consider repository environments as a software infrastructure for managing unstructured data and digital objects, as much as databases are software infrastructure for structured data.

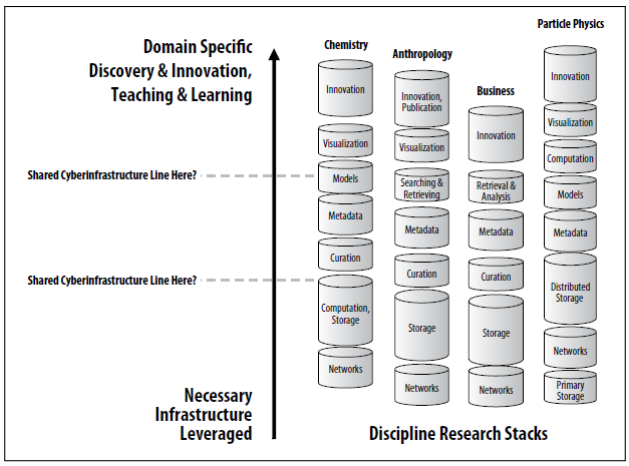


Figure 1.2: Layers of Scholarly Infrastructure in [320]. Brad Wheeler asks where to put the line between infrastructure and the application. He concludes that there is no single answer to this, but rather institutions (providing infrastructure) and researchers (employing applications) have to take this decision together.

1.1 Scenarios of Open Repository Environments

The previous section introduced repository environments and how they can be *open*. "Open" was described to mean accessible and extensible for distributed agents (e.g. repository systems, registries, visualisation tools) in a decentralized manner, yet it does not mean without security mechanisms or anarchic. In order to fill these keywords with life, this section presents scenarios in which open repository environments excel.

These scenarios are currently unsolved or they are tackled by hard-coded mechanisms that are difficult to transfer. They thereby establish a "benchmark" for this thesis, since the concepts presented in this thesis aim to resolve the scenarios in a generic, ad-hoc manner. Moreover, many of the scenarios illustrate prototypical challenges in repository environments, representing a whole class of conceivable situations. Even though many of the scenarios, prototypes and examples presented in this thesis are related to e-Infrastructure in the humanities due to the author's background, the results of this thesis can be generalised to other domains.

The scenarios were developed after surveying a multitude of repository systems, with a

closer analysis of iRODS of the field e-Infrastructure (cf. section 2.1.2); as well as Fedora, aDORe, and DSpace from the repository community (cf. section 2.1.1). In addition to these technical analyses, discussions in a series of international workshops that we organised exclusively for this purpose [83] as well as an international standardisation process for repository infrastructure [90] contributed to creating these scenarios.

The scenarios describe aspects of open repository environments, which build upon interweaving distinct repositories or outsourcing functionalities to external services and infrastructure. Thus, the analysed repository systems generally fail to satisfy these scenarios when viewed as stand-alone systems. At the same time, existing repository federations (e.g. Driver [149], Dare [290], Europeana [21]) fail as well to satisfy the requirements put forth by open repository environments, since – as opposed to those traditional federation mechanisms – an open repository environment (a) deals with material that changes frequently and needs to propagate those changes in a timely manner, it (b) includes non-repository agents (e.g. format registries, migration services, visualisation of content networks), and (c) it enables interoperability on multiple layers of abstraction. [85] The following scenarios display all these features, and they are clustered along two interoperability levels, object storage and federation.

1.1.1 Scenario 1: Repository Storage

Most current repository systems [103, 257] manage their contents in the file system: the constituent files of a single digital object, and often metadata and relations as well. This is displayed by all the repository systems that were surveyed, e.g. DSpace [137], EPrints [292] and iRODS [201] store the files as bitstream into the file system; and both aDORe [99] and Fedora [221] package the files they manage into standards-based, human-readable XML containers – Fedora into METS [238] and aDORe into MPEG-21 [4]. The perceived benefit of this approach is its stability even when building systems to last for long-term periods [111, 174], according to discussions conducted in the course of this thesis. [83]

Today, these storage tasks are hard-wired into those repository systems, and also

storage is often limited to the local hard disk. One of the exceptions is the data grid iRODS, which is designed to manage distributed storage and is capable of dealing with various kinds of storage hardware including online, nearline, and tape storage. [243] Nevertheless, its storage management is tightly coupled into the system core, in that there adaptations need to be hand-coded into the system core.

One of the first repositories that was configurable to employ an external storage handler was DSpace, which starting from its release DSpace 1.3 (August 2005) offered an option to use the Storage Resource Broker SRB [2], a distributed storage system and precursor to the iRODS system. Today storage modules on a plug-in basis are a key topic in repository research, with repository systems including EPrints [187] and Fedora [15] having them on their development roadmaps.

However, as we point out in [85], current activities are often isolated and tailored to a specific repository system. There is no standard for a storage interface emerging that is adopted across the repository community.

Therefore, the following scenarios are only possible with a system-independent storage interface, and they also describe some of the features we expect from such an interface.

1.1.2 Scenario 1.a: On-Demand Storage

Both small and large institutions struggle with the resources and the manpower tied up by effective storage management. Creation and maintenance of storage infrastructure can be particularly expensive when reliable storage and bit-preservation need to be attained, which often involves geographical data replication to ensure reliability and scalability of the service. [111]

Small institutions may be overwhelmed with the long-term costs of establishing and maintaining their own storage facilities. [39] Smaller institutions lack the economies-of-scale displayed by large data centers, [135] even in the face of a single huge collection, e.g. an art school preserving audio and video recordings. [24] Large institutions on the other hand may be able to pool their storage requirements or even have a local data center. However, large institutions often have multiple repositories

where each repository serves a different user group, yet the content overlaps to some degree. [82] For both, small and large organizations, outsourcing storage to external service providers may be an ideal solution.

Those two general situations of small and large organisations apply to a multitude of individual cases. So many in fact, that the DuraCloud initiative [212] aims to establish a commercial storage service that will also provide various low-level preservation services. However, DuraCloud was initiated towards the end of the thesis, and at the time of writing (August 2009) there are no results available. So until there are results from DuraCloud or other initiatives and repository storage becomes an infrastructure service, repositories will continue to store their valuable content locally on the repository server, often using an "insecure, backup-free, under-the-desk rogue server" [324].

1.1.3 Scenario 1.b: Distributed Access

As repository storage is distributed over multiple storage hardware resources, or outsourced to external storage providers as described in the previous scenario, access to a digital object is ideally granted directly through the very storage node it is contained in.

This scenario has two aspects. First of all, access to digital objects that is routed through a central repository server introduces a single point of failure and hampers scalability of the overall system. Therefore, direct communication between the client and the storage node is preferred, without an additional repository component as intermediary. Systems supporting such direct communications often have a small overhead in initiating a communication, in which the right storage node is identified and subsequently the client is redirected to that very storage node. Identification of the 'right' storage node can be based on various criteria: e.g. where the requested file is located in the first place; in case the file is redundantly stored at multiple locations, which is the location closest to the client, respectively which location has the quickest response time at that very moment.

The second aspect ties right into this overhead for identifying the 'right' storage node

and distinguishes between those systems, where this identification process is largely transparent to the client, and those that require active participation of the client. Systems like the commercial storage service Amazon S3 implement this identification of the 'right' server such that it is largely transparent to the client – in the case of the Amazon S3 REST interface [1] by using the HTTP response "307 Temporary Redirect" [150]. Other systems such as the Storage Resource Manager SRM [176] embed more intelligence into this identification process and require the client's activity – in the case of SRM this includes negotiation of the preferred transfer protocol. These two examples may indicate that those systems, for which the redirection process is transparent, tend to be more light-weight, in contrast to systems, which require active negotiation of the client. However, this intuition cannot be taken as a verifiable rule.

1.1.4 Scenario 1.c: Repository Reconstruction

Whenever a repository system crashes without chance of reactivation, or a legacy repository platform needs to be replaced by a novel one, a fresh repository installation may need to be initialized with the objects that were stored in the older one. Software migration or repository reconstruction is an important issue, particularly when dealing with repositories that are used for long-term preservation [317], and also in the general case as recently demonstrated by Google who instituted an engineering team called the "Data Liberation Front" commissioned to support data extraction from (and import to) all Google products. [142]

With regard to existing repository systems, e.g. Fedora offers a dedicated reconstruction mechanism. [116] However, in the absence of a generic, standard repository storage interface (cf. section 1.1.2), such reconstruction mechanisms are tied to the very system and perhaps even to the specific system version in question.

1.1.5 Scenario 2: Federation

"In the future there will be only one (virtual) repository" – this is one of the visions for repository infrastructure formulated at the repository workshop at the Open Grid

Forum Barcelona [83]. In fact, there are today various initiatives striving to federate physically distinct repositories into a single virtual repository, including Driver [229], Dare [133] and Europeana [126]. Content in those cases is dispersed over various locations for historical or for organizational reasons (e.g. each university library establishes its own institutional repository). In their integration efforts, the goal of all these initiatives is to build a single portal that provides access to these dispersed locations.

However, these initiatives predominantly focus on exchanging metadata about publications. In the case of the three initiatives mentioned above – Driver, Dare, and Europeana – all of them employ the prevalent Protocol for Metadata Harvesting of the Open Archives Initiative, OAI-PMH [198]. The limitations of these kinds of federations are becoming apparent as repositories are increasingly managing research data (as opposed to publications), and multiple repositories are exchanging that research data for reuse (as opposed to only exchanging the metadata for viewing). [86, 173, 253] Also, the aforementioned federations usually just take whatever they can get. More fine-grained control over the federation may be required for building thematic collections composed of selected pieces from various repositories, or in the case of inter-disciplinary and inter-institutional projects.

In other words, the requirements on federation in open repository environments is very unlike traditional federation mechanisms. Various initiatives recognised this, including [104], who call for repository interoperability and Next Generation Services, which enable "deep sharing through experimentation with aggregation other than metadata harvesting, resulting in the capacity to move digital objects from domain to domain, along with the ability to modify and re-deposit them in a different location in the process."

The following three scenarios discuss respectively the federation of data (scenario 2.a), sharing metadata of frequently changing objects or collections (scenario 2.b), as well as exchanging data with external, non-repository agents (scenario 2.c).

1.1.6 Scenario 2.a: Scientific Analysis

Particularly in the humanities, research is not confined to a single location but often includes material from dispersed locations. [84] Each of these locations may have an institutional repository with relevant material for a specific research question that bridges all those locations. In this scenario, these distinct repositories federate without changing the underlying technologies, offering search and analysis across their collections in a dedicated portal. With the emergence of more and more repository-based research environments, the requests for scientific analysis of repository contents is likely to increase. [7, 173, 235]

The kinds of analysis conducted by such a joint portal can be manifold. Federation mechanisms should not constrain analysis technologies, and they should not constrain the kind of objects to be shared both with regards to their content and their metadata. In particular, we would like to point out two challenges that analysis functions may pose on the scalability of the overall system. First, an analysis technology could be very resource-intense even when applied to only a single repository, yet should not bring down the performance of the repository. Retrieval or clustering techniques are just two of the fields offering dedicated analysis methods that are very resource-intense, yet may be of interest to repository-based research environments. [74, 298]

The second challenge mentioned here is that fast-changing content should not bring down the scalability of the overall system, even as many repositories join the federation. Fast-changing content requires an immediate link between the numerous repositories and their joint analysis portal to avoid inconsistencies, and may hence increase the communication demand significantly compared to immutable content.

1.1.7 Scenario 2.b: Task Tracking

An early step in many research activities in the humanities is the collation and preparation of the material to be addressed. [296] 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. [312]

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 [35]. Dedicated solutions are emerging for digitisation workflows [43] or as part of large editing systems [41]. However, we are not aware of any existing solution that spans 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, keep track of changes to those sources and material, and integrating the material (without necessarily extracting it from its original source).

1.1.8 Scenario 2.c: Out-Sourcing Preservation Actions

The preservation of objects over long periods of time [100, 255] is a key challenge for repositories, in the face of the rapid advance of hard- and software environments. The importance of taking preservation actions has already shown in many spectacular cases, where important research data has been lost: up to 20 percent of the data of NASA's 1976 Viking mission to Mars have been lost [285]; satellite data recorded in the 1970s, which was to be used to identify ecological trends in South America's Amazon Basin, have been lost; and there are many more such negative examples (also in non-scientific contexts) [178, 226].

Trusted digital repositories [174] are assigned to reliably preserve their contents over time. Preservation of digital objects may involve strategies like migration, where files and metadata are transferred into newer or more stable formats, before old formats run

danger of becoming obsolete. [317] This migration process – or "conversion" as the technical aspects of transferring an object into another format is called – may need to be conducted external to the repository for two reasons. [146] First of all, batch conversion – e.g. of all TIFF files to JPEG2000, or all PDF files to PDF/A [269] – may be compute-intense and hence it should not be conducted directly on the live repository server. At the same time, there may already be external services that offer conversion capabilities, and a repository that receives a myriad of different formats on ingest may not be in the place of providing dedicated conversion services for all of these formats. [268]

1.2 Contribution of this Thesis

This thesis emerged from an iterative process of analysis, technical experimentation and implementation of production systems, as well as discussions with experts in various communities including trusted digital repositories [78, 92], the grid community [249], and e-Infrastructure for the humanities on a national [81] and a European level [86]. Amongst several repository-related activities in various organisational contexts in the years 2001 to 2009, the author is technical architect of 3 initiatives that push towards open repository environments. Those aspects of those initiatives that contribute to the concept of open repository environments are presented later in this thesis.

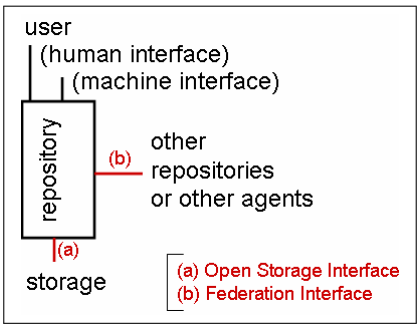


Figure 1.3: Environmental Model of a Repository. Identifies (a) the Open Storage Interface and (b) the Federation Interface, which are the building blocks of Open Repository Environments and a focus of this thesis.

Figure 1.3 depicts an environment model focusing on a single repository and its

interactions with other agents in the environment. This thesis defines a reference architecture for such a repository and then focuses in particular on the storage and federation interfaces. It argues that openness and distribution in these two interfaces are the key building blocks for an open repository environment. This thesis is structured accordingly, with the following scientific contributions:

1. A reference architecture for repository systems (chapter 2.2)

    - Definition of a layered repository architecture that fosters three levels of abstraction – file, object, and application – with two interoperability interfaces between the layers – open storage and federation.

    - The architecture was derived in an analysis of existing systems and experiences, emerged from expert discussions at workshops on repository infrastructure we organised [83, 85], and was evaluated through implementation in three initiatives presented in this thesis.

    - It provides the overall framework, identifies gaps in repository interoperability and decentralisation, and is the basis for further analysis in this thesis.

2. An open storage interface [interface file / object layers] (chapter 3)

    - Specification of the central attributes of on-demand storage infrastructure that is independent from a specific repository system or other agent, is stable as agents evolve, and is capable of serving multiple agents.

    - The specification of the infrastructure was derived from a series of experiments with diverse distributed storage systems offering dissimilar interfaces: a virtual file system (Cleversafe), a dedicated storage handler (iRODS), as well as a RESTful interface (S3-like).

    - Implementation of a storage infrastructure that translates grid technologies (the Storage Resource Manager, SRM) into a Cloud-like storage interface (S3-like). Evaluation of S3-like through experimental deployment of a DSpace repository and other utilities.

3. Federation attributes [interface object / application layers] (chapter 4)

    - Mapping of existing federation activities based on attributes pertaining to both, the digital object and the overall information system: syntax and semantics (object), structure and patterns (system). Identification of gaps, particularly with regard to patterns.

    - Profiling of a notification pattern, which allows more immediate and direct federation of repositories than current federation patterns and thus enables federated applications that are more interactive than previously.

    - Development of an Atom-based federation network, which allows a hybrid push/pull notification pattern that is robust, light-weight, and serves all aspects of a digital object (data, metadata, relations).

4. TextGrid, a live environment built on open repository concepts (chapter 5)

    - Implementation of the reference architecture from chapter 2.2 in environment based on grid technologies.

    - Creation of an environment that is open on three levels: as an infrastructure (primarily focused on virtualised storage), as a platform (where new functionalities can be added and re-used), as well as a software (targeting scholarly research in the humanities).

    - Discussion of organisational and social aspects coming to bear in an open repository environment.

2 Repository Architecture: Identifying Interoperability Channels

This chapter develops a layered repository reference architecture. Rather than being a construction plan for a new repository system, this reference architecture aims to establish a framework for connecting repositories and other agents in an open repository environment. Its three layers are hence designed to prescribe as few as possible and to enable heterogeneity, yet to ensure a minimum level of interoperability between the agents.

The repository architecture is mainly based on three activities: a survey of related fields for precursors to open repository environments and an analysis of existing repositories [87]; intense discussion in the e-Infrastructure [83, 90] and repository communities [89, 91]; as well as its validation in the e-Infrastructure environments TextGrid and Dariah as presented in chapters 5 and 4 respectively.

2.1 Related Fields and Technologies

Two fields are of particular importance to this thesis: trusted repositories and e-Infrastructure. Despite their diversity, both fields share common goals and are complimentary when it comes to establishing repository-based research environments, as the following sections describe. However, interoperability between them, let alone their convergence is only just starting to emerge, which makes solving the scenarios a

particular challenge.

When it comes to digital infrastructure in research it often seems the world is evenly split into two groups: those who care about high-performance computing [252], and those who care about trusted data management and data-driven science [73, 228, 247, 254]. On the one hand is the simulation of cosmic phenomena [249], and on the other the collation of national census data [55]. On the one hand deciphering the human genome, on the other preserving 3D models of the Buddhas of Bamyan statues, a former UNESCO world heritage site in Afghanistan destroyed by the Taliban in 2001.

These perceived distinctions are, of course, purely artificial. A researcher may move back and forth between the data collected in experiments, simulation and analysis, collaborative environments, and publication platforms. The integration of these two disparate digital environments is thus a concern for all stakeholders: for researchers and generally anybody moving between environments; for information technologists who can learn from each others experiences; and for funding bodies who strive for efficiency and quality in their funding subjects. As the following sections show, the existing experiences and available systems in those communities as well as in related fields are complimentary and provide a good starting point for achieving this integration.

2.1.1 Institutional Repositories

Ever since the move to digital processes, scholarly communication is changing dramatically. [234, 237] Institutional repositories are an offspring of that domain combining diverse research topics in scholarly communication including open access [153] and digital preservation [317]. This section looks at the evolution of institutional repositories and finds that repository research still focuses too much on research products rather than the whole research process. Being at the end of the information process chain repositories into a restricted scope, monolithic design, and hence a limited added value for its users.

Institutions always had the need to capture and manage their research output, and e.g. the Oxford Text Archive founded in 1976 can be seen as one of the early precursors to the current institutional repository landscape. [94] However, a steep rise in numbers of

institutional repositories and the emergence of a coherent community only happened at the turn of the century. [92] Today, the number of repositories is soaring and can hardly be tracked: The OpenDOAR directory [53] – a portal for voluntary registration – refers to more than 1.400 operational repositories by October 2009.

However, repository systems are still very local phenomena. While they are advanced when it comes to sharing metadata about digital objects, [197] they are only beginning to explore distribution, sharing functional components, and more complex architectural patterns. Repository research to date has predominantly dealt with how to get published material into the repository (e.g. communication with depositors, copyright and rights issues in general) and how to disseminate it (e.g. RSS notification, personal publication pages), thereby labelling repositories as static archives of archived work. Only slowly repository-based environments like the one at NSDL ([215], [219]) with 2.5 million educational resources and the multi-disciplinary research environment of the Max Planck Society, eSciDoc [136] are emerging. Rather than focusing on the dissemination of publications, these repository environments promote re-use of research data, collaboration, and they integrate into the whole research life-cycle.

While we have been looking at various open source software packages for institutional repositories ([257], [103]), we will focus on *Fedora* [221], *DSpace* [281], as well as *aDORe* [99] in the further analysis of this thesis. We chose these three repository software packages, since they belong to the most prevalent systems in the repository community [52], and they display some interesting concepts with regard to openness and distribution, as the following mapping of their functionalities to the scenarios shows.

(The following listing maps features in Fedora, DSpace and aDORe onto the scenarios described in section 1.1, using a rating of (+) – poor coverage, to (+++) – good coverage).

1. **1.a On-Demand Storage** (+) – Fedora [123] and DSpace [2] have both experimented with an integration into the Storage Resource Broker (SRB) [112]. While DSpace provides SRB-integration as part of their stable distribution, there are hardly any live DSpace instances known to us that benefited from that. This may be the case as there is some overlap in functionalities between SRB and

DSpace and the integration is not based on open standards (cf. the detailed discussion in chapter 3.2).
Currently, the major repository packages are looking into generic storage interfaces: Fedora in [15] and together with DSpace in [12], as well as EPrints in [187]. However, at the time of writing (October 2009) there are no results from these activities available.

2. **1.b Distributed Access** (+) – As mentioned in the previous item, storage of institutional repositories is localised for the most part. aDORe is the only repository system, which addressed distribution in its core architecture. This distribution is achieved through federating disparate OAI-PMH providers, a standard mechanism that could be extended to other repositories and which is discussed in section 2.2.3.

3. **1.c Repository Reconstruction** (++) – Both, Fedora and aDORe are capable of completely reconstructing a repository from file storage. Since they both employ standards for object markup – being METS in Fedora [238] and MPEG-DIDL in aDORe [4] – a failed repository could be reconstructed by an agent that builds on another platform than Fedora or aDORe with little custom coding. However, there is no completely generic mechanism for repository reconstruction.

4. **2.a Scientific Analysis** (+) – Scientific Analysis is not a key objective for institutional repositories. To our knowledge there is no dedicated support for conducting dedicated analyses on repository contents by Fedora, DSpace, aDORe or others.

5. **2.b Task Tracking** (+) – We are not aware of any significant applications across multiple, heterogeneous repositories apart from federated search. Likewise, while there are Task Tracking applications tailored to specific platforms [43], they fail to integrate disparate repositories.

6. **2.c Out-Sourcing Preservation Actions** (++) – Preservation is a key incentive for repositories and there are numerous respective activities. However, they are mostly tailored to a specific platform just like for EPrints [186, 293], or

more generic services have failed to overcome prototype status and lack adoption. [146, 194] A reason for this may be the lack of standards that enable administrative or scientific workflows with external services.

2.1.2 e-Infrastructure

There is an increasing number of exemplars where research is driven by teams that span communities, countries, languages, and whatever other traditional classifications may exist. [65, 84, 249] Rather than disseminating the products of their activities *post factum*, these teams work collaboratively throughout the entire research process. [97] This section looks at some of the e-Infrastructures supporting such collaborations, particularly at grid technologies. It finds that these technologies are – despite their promise to be generic – in many ways tailored to their user communities, and often focused on short-term processes rather than sharing and re-use of intermediate results.

In 1998 Ian Foster and Carl Kesselman [158] envisioned a digital infrastructure, in which existing hardware is shared despite geographic dispersion and heterogeneous technical platforms. Grid technologies clearly had many predecessors and other founding fathers. Ian Foster himself acknowledges Len Kleinrock for his metaphor of "computer utilities" in 1969. [156] The massively distributed computing infrastructure of high-energy physics, with the CERN as one of their epicentres, became the first production grid infrastructure. The latest instalment at the CERN, the Large Hadron Collider, produces about 15 petabyte of data annually, which is stored and processed by tens of thousands of computing elements at computing centers around the world. The LHC may be the classic example, yet today the grid is applied in astronomy, climate research, medicine and a host of other academic disciplines and commercial companies as well. Those discipline-specific applications are often mounted upon national grid infrastructure such as UK Grid [60], the German D-Grid [249], or the US Cyberinfrastructure [97].

Technologies like the grid promise to provide hardware resources on-demand, something not only needed in research collaborations but in many digital environments. Other, also commercial offerings are therefore emerging promising sheer unlimited availability

of hardware, to take the burden of hardware management off the user, and to raise robustness while lowering overall costs. For instance with regard to storage, commercial systems for Information Lifecycle Management align the value of information with the most appropriate and cost effective IT infrastructure from the time information is conceived through its final disposition. [154] Storage clouds enable outsourcing of storage to remote service providers. These offerings go beyond the original vision of the grid. Foremost, grid environments expose much of their inner wiring to the user, turning configuration and using grids into expert activities. The pros and cons of these platforms notwithstanding, the philosophies of virtualizing hardware resources resemble each other in many ways.

Moving on from virtualizing storage and computational hardware, the grid community is currently exploring virtualising service components as well. The Next Generation Grids Expert Group envisions a convergence of grid technologies with semantic technologies, in which grid services know their capabilities and specialised semantic services are capable of dynamically matching service capabilities with tasks. [145] However, early experimentation (e.g. [119]) remains to be deployed in production systems.

While the community has always been particularly active on computation, services and workflows [131], much ground remains to be covered with respect to metadata management and repository-like features. It is symptomatic that the longevity and semantic annotation of data has not received nearly as much attention as semantic grid services. This may be changing though, as data-driven science increasingly recognises the importance of data stewardship, and services like ANDS, Datanet, and others are emerging.

This lack of attention also reflects in the availability of systems. While there is an enormous variety of grid middlewares, they are mostly focused on computational applications or operate on a low hardware level. For example, the Storage Resource Manager (SRM) [176] standard is capable of mediating between various hardware platforms, yet does not include any metadata management that goes beyond file attributes. The only system that displays repository-like features are the Storage Resource Broker (SRB) [112] and its successor *iRODS*, the Integrated Rule-Oriented

Data System [267].

(The following listing maps features in iRODS and the SRM onto the scenarios in section 1.1, using a rating of (+) – poor coverage, to (+++) – good coverage).

1. **On-Demand Storage** (+++) – e-Infrastructure activities have distributed and scalable storage infrastructure as part of their core concepts. Pertinent examples are SRM and iRODS, which at their core devise storage virtualization and management. SRM is an interoperability protocol and geared to provide on-demand storage by integrating partner sites with heterogeneous storage resources. In iRODS, however, all nodes – or "bricks" in iRODS terminology – have to run a dedicated iRODS instance and, hence, there is no interoperability with non-iRODS partners.

2. **1.b Distributed Access** (++) – The dedicated Java library 'Jargon' [45] allows client-access to iRODS that may be distributed to the iRODS brick accommodating a specific file. However, any search for a file or metadata-related requests are rooted through the central metadata database iCAT [261]. Furthermore, iRODS does not offer any native consistency mechanisms in the case of file replication or concurrent access. Contrary to that, the Storage Resource Manger SRM [176] offers virtualisation, data replication and locking mechanisms that are conducive to ensuring consistency in storage infrastructure.

3. **1.c Repository Reconstruction** (+) – There are no mechanisms for reconstructing an iRODS repository from file storage. The SRM cannot be considered a repository system in the first place, as it is purely file-oriented and does not offer basic metadata management like iRODS [250], let alone more sophisticated object modelling capabilities.

4. **2.a Scientific Analysis** (+++) – Computational analysis is a key focus of e-Infrastructure environments and one of the foundational reasons why the grid came into existence. However, there still is a curious disconnect between computational grids and trusted data grids, let alone repository-like systems. Storage resources in grid environments are typically used for staging files to computational resources rather than for reliable long-term storage of digital

objects, [177] though this is slowly changing with the increasing importance of data-driven science. For example, there is no official gateway between iRODS (data grid repository) and SRM (data interoperability for computational grids). [22] In a nutshell, while grid environments are a fertile ground for scientific analysis, there interoperability and policy issues remain to be overcome.

5. **2.b Task Tracking** (+) – While services for monitoring hardware resources [62, 160] and services [63] are prevalent in e-Infrastructure environments, there are no prevalent services for monitoring and federating digital objects.

6. **2.c Out-Sourcing Preservation Actions** (+) – In principle, computational grids offer an ideal place for distributing batch preservation tasks such as format conversion. However it is not yet clear, how a trusted repository could reliably and in a generic manner interoperate with a grid environment. Initiatives like the D-Grid project WissGrid have identified this as an unsolved field, [122] yet results from their research are still pending.

2.1.3 Other Precursors

Even though this thesis was created in the context of academic research and institutional environments, managing files and their context is a prevalent back-end activity in diverse communities. This section briefly touches upon some of those fields, to convey an impression of the breadth of the field and to also show where repositories found inspiration and experiences to benefit from. As discussed in section 2.2.1, for example, *re-representation services* have a predecessor in delivery workflows such as those to be found in content management systems. Without identifying each individual tidbit, the following list thus gives a brief overview of further related fields and technologies. Each item is contrasted against our notion of repositories to further sharpen this notion. It also underlines how repositories could be – like databases for relational or semi-structured data – a generic infrastructure technology for file-based data of relevance to all of those fields.

- **Knowledge management** enables organisations "to collectively and

systematically create share and apply knowledge". [232] Ideally, in a Learning Organization [278] tacit knowledge is recorded and spread across the organisation to grow and evolve.
Unlike Knowledge Management Systems, repositories are not linked to project management, organizational learning and processes within an enterprise, although they could accommodate all that.

- **Data Warehouses** collate an organization's structured data to facilitate reporting and analysis (potentially across various database sources and their heterogeneous models). Data Warehouses are often separate from an organization's operational systems and geared towards performance. [165, 211]
  Unlike Data Warehouses, repositories are at any time more open to possible ways of processing their collections, yet (because of that) repositories fail to match (and never will) the high performance offered by data warehouses. On top of that, data warehouses usually build on structured data in databases, whereas repositories focus on unstructured data.
  (Please note, Data Warehouses are not considered in the following, since they are a database rather than an file-based technology.)

- **Content Management** supports processes within an organization and the management of unstructured information generated as part of these processes; e.g. Enterprise Content Management [67] supports business processes, Web Content Management [68] helps maintaining, controlling, changing and reassembling the content on a web-page.
  Unlike Content Management Systems, repositories do not solely focus on the business context or publishing process of documents (e.g. web pages) but support all conceivable sorts of application environments. In fact, content management systems could be built using repository technologies.

- **Records Management** Systems are intended for the management of electronic and physical records from creation to their disposal, which provide evidence of an activity through their content, context and structure. [167]
  Like Records Management Systems, repositories have a stake in modelling and preserving digital information. However, repositories may support other lifecycle

stages (rather than just tapping into them), and they are more flexible as to the diversity of information and the type of metadata they allow. [213]

- **Digital Libraries** manage and provide long-term access to digital collections. In some broad definitions the field also comprises information retrieval, digital preservation and various others. [11]
  Like Digital Libraries repositories aim to disseminate content and offer it for re-use. However, digital libraries usually collect existing information from various sources, whereas repositories may also be concerned with earlier life-cycle stages and support the re-use and revision of existing information. So while the concepts do not fully overlap, the field of digital libraries is more comprehensive those of repositories.

All these fields – despite dissimilar terminologies – share similar ideas when it comes to repository-like systems. Due to the breadth of all of those communities, there is overlap between them. For example, e-Learning environments are being built on top of repositories [272], digital libraries [152, 239], content management systems [196], as well as custom-made [102, 195], open source [56] and commercial products. Despite their different backends, all these e-Learning environments follow similar goals and share similar requirements.

Also there are some indicators that the fields are increasingly overlapping in terms of technology, although that is difficult to attest globally. To name just some, preservation features have always been relevant for repositories, as much as they are relevant in digital libraries or records management systems. The ISO standard for an Open Archival Information System (OAIS) [111], which establishes a reference model for preservation systems, is therefore equally relevant for all of them. In another context, the combination of a content management system (Plone) and a repository (Fedora) has been tested in the open access e-Journals project DIPP [273]. Last to be mentioned, there are also ideas for integrating the content management standard JSR 283 [283] into repositories to enable interoperability between heterogeneous Java-based systems. [20]

2.1.4 Facets of One Environment

e-Infrastructure and repositories were originally created with disparate goals in mind: e-Infrastructure initially aimed primarily at creating a computing environment for collaborative research, whereas repositories were primarily dedicated to capturing research outputs (cf. table 2.1).

However, the lines between them are blurring, as symbolised by the CERN – a lead institution in the development of both grid and repository technologies. [168, 192] The grid community is increasingly aware of the challenges and opportunities in data management, interoperability and preservation to foster collaboration. [13, 122] Vice versa, repositories are increasingly present in all stages of object life cycles, facilitating active collaboration in research, companies and other environments. [65, 136] Moreover, the positions seem to switch as the repository community discusses with fervour whether long-term preservation should or should not be part of a repository's mission. [48] Preservation has become unpopular due to the perceived barriers it puts on depositing material, thereby forgetting that preservation is a key value proposition for institutions and depositors alike that used to be a trigger of the repository movement in the first place. [233] Ironically, it is – amongst other – the preservation capabilities that caught the grid community's interest in repositories. [90]

In addition to the ongoing convergence between the two communities, their concepts and technologies are complimentary when it comes to tackling the scenarios that establish an open repository environment. Table 2.2) summarizes the findings of the discussion above and shows that e-Infrastructure excels on two lines: when it comes to the storage of digital objects, as well as in task-oriented analysis and processing of the digital objects. While e-Infrastructure ensures the availability of data and of tools to work with them, repositories enable re-use and sharing through data management tasks like modelling and recording the context of a digital object, provenance, version management and preservation. Repositories therefore link various components and technologies and are themselves essential components of e-Infrastructure for collaboration.

Despite these opportunities, interoperability between e-Infrastructure and repositories

	e-Infrastructure	**repositories**
	computing infrastructure	object management and dissemination platform
supports	collaboration, through reuse of data and high performance computing	dissemination, through publication of research results
resources	storage and compute (virtualized hardware)	meta/data (linked information)
time unit	immediate, or asap	long-term
features	high performance, scalable	stable, rich (functions, content)
user group	research community (within a discipline)	research community (of an institution)

Table 2.1: e-Infrastructure vs. repositories – Originally, these two fields cover distinct objectives, yet they address a similar user group and that user group is expanding. Due to this overlap in target community, the fields are starting to converge (cf. section 3.2).

remains an open issue. Some initiatives ventured towards such combining the two fields [2, 39, 123, 202], yet those activities remained largely tailor-made to a specific e-Infrastructure and a specific repository system, and they were closed to other agents.

Figure 2.1 aims to capture this convergence and complementarity as it was also described in sections 2.1.1 and 2.1.2, where institutional repositories used to be rather monolithic (e.g. OPUS); then they allowed for machine interfaces or a plugin architecture that facilitate system adaptation and external clients (e.g. Fedora); and they also progressed with regard to federation of heterogeneous repositories (e.g. aDORe). The evolutionary graph also indicates that – if the communities fail to agree on common standards – the storage virtualisation capabilities in e-Infrastructure are slowly being duplicated by the repositories. If the two fields eventually converge towards open standards on a storage as well as a federation layer, we may eventually see combined systems emerge.

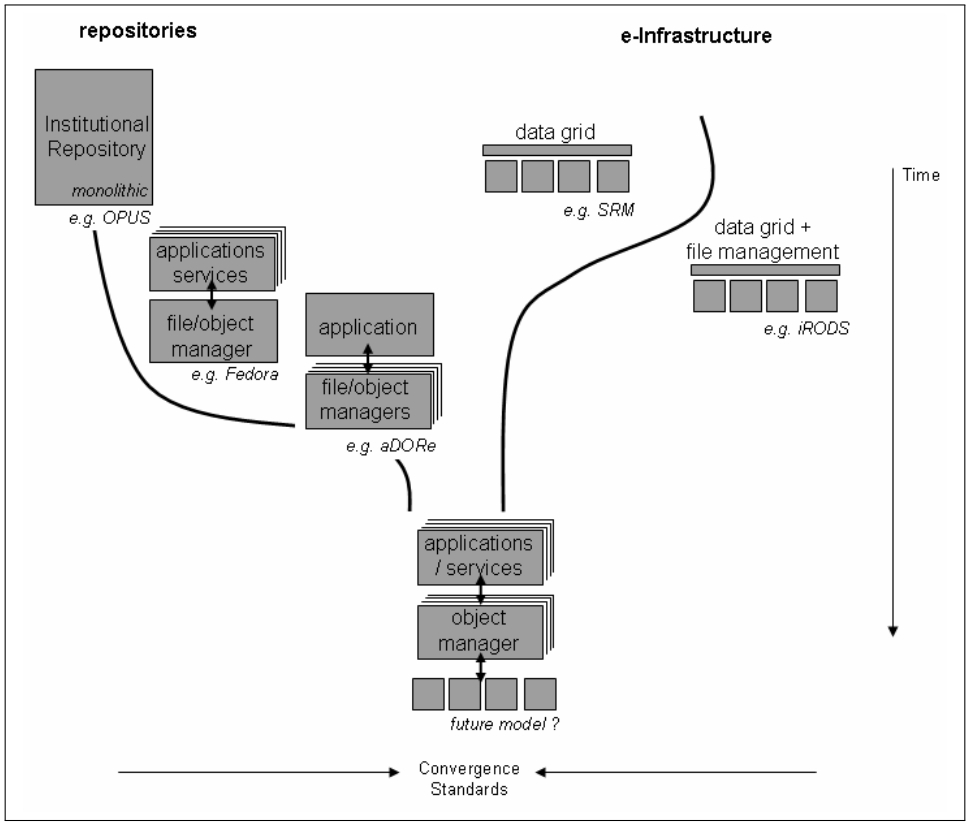


Figure 2.1: Tentative evolutionary tree of systems in the repositories and the e-Infrastructure communities. While repositories are increasingly decomposing individual functions into re-usable components, e-Infrastructure is increasingly offering management capabilities for files and metadata.

	e-Infrastructure	**repositories**
1.a On-Demand Storage	+	+++
1.b Distributed Access	+	++
1.c Repository Reconstruction	++	+
2.a Scientific Analysis	+	+++
2.b Task Tracking	+	+
2.c Out-Sourcing Preservation Actions	++	+

Table 2.2: e-Infrastructure vs. repositories – Coverage of scenarios defined in section 1.1 on a scale from (+) – poor coverage, to (+++) – good coverage. This table underlines that both fields are conceptually complimentary, disregarding interoperability issues.

2.2 A Repository Reference Architecture

This section defines a three-tier repository reference architecture, where each layer rises in the level of abstraction from file, to object, to application. Novel in comparison to existing repository architectures are the open interfaces between the layers – the Open Storage Interface and the Federation Interface –, which allow the combination of distinct repository components vertically along these layers, but also horizontally by integrating external agents. The following chapters then focus on these interfaces in detail – chapter 3 analyses the Open Storage Interface, and chapter 4 analyses the Federation interface.

Before specifying the layers and the interfaces between the layers, however, the following sections define our notion of "digital objects" and of an "open environment", which are prerequisites to the layer model.

2.2.1 Features of Digital Objects

In the introduction (cf. section 1) we succinctly described *digital repositories* to be management systems for *digital objects*. Turning this perspective around, we can hence derive the requirements from repositories from the notion of digital objects. This section therefore reviews and refines the notion of a digital object, and puts them into perspective with the repository survey as outlined in the previous section.

With some minimal variations in scope and terminology across the various systems and communities, the following features are the constituents of a digital object.

- **files** – Digital objects consist of a single or possibly multiple files, either packaged together into a single container or tightly linked together by reference. This includes e.g. small XML files as well as 100 megabyte images or even larger videos. All the reviewed systems also internally stored file-based representations of the objects, where some systems store the files as they are (DSpace, iRODS, Tupelo), while others use container formats like METS [238] (Fedora) or MPEG-DIDL [4] (aDORe) to package (multiple) files and metadata into a single

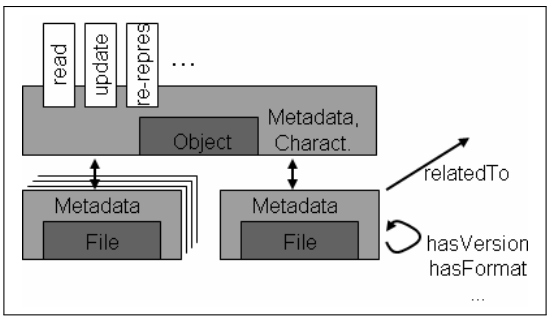


Figure 2.2: Object Features – an object provides a map of potentially multiple files, metadata attached to files and other referable entities, and potentially relations, as well as service stubs attached to the object. An object may span multiple systems, although the manageability of the object may then be limited.

logical file. The advantages of file-based storage are mainly simplicity and stability – even if no other data is available, repository systems can be rebuilt from the information contained in the files.

- **metadata** – Object metadata is essential for a variety of aspects including repository management, retrieval, as well as for conveying the context of the object for later re-use. Existing repository systems vary in their flexibility of their metadata approaches from defining a fixed metadata schema (DSpace, aDORe), to allowing the customization of the metadata schema (iRODS, Fedora, Tupelo). In the former case, both DSpace and aDORe build on Dublin Core [264] – an open standard nevertheless a fixed schema with defined meaning. While Dublin Core may be suitable for many publication environments, it may be insufficient for many repositories containing research data or other more complex digital objects. [306]
  Customizability of metadata schemas can be on various levels, ranging from the possibility of adding new metadata fields (iRODS) to supporting sophisticated modelling capabilities through e.g. semantic technologies based on XML, RDF, and OWL (Tupelo) [188]. In between those two poles, frameworks for object modelling (Fedora) [148] allow the definition of templates for digital objects, which define the structure of an object comprising potentially multiple metadata and other datastreams.

- **relations** – Relations link objects, services, or just anything with a suitable identifier. References have become almost universal as e.g. underlined by the Dublin Core Abstract Model [264], and the pertinent activities on defining "persistent identifiers" that are geared to increase the reliability of references over time, including the Uniform Resource Identifier URI, the Digital Object Identifier DOI, the Archival Resource Key ARK [185]. However, to our knowledge only Fedora and Tupelo use relations as part of their object models and internal architecture.
  Overall, while relations to link internal and external objects and to embed *re-representation services* into the repository architecture potentially increase the flexibility of the system, they are not a core requirement for repositories.

- **re-representation services** – Re-representation services are services that are attached to digital objects to convert them on-access into other representations. One example is a digital object containing a 80 megabyte high-resolution image file that is converted on-the-fly into a small thumb to be displayed on a gallery site, or of which only a small defined area is transferred. [114] Since a digital object may consist of both the master as well as a small thumbnail, it is a question of the underlying model whether the thumb is created on-the-fly (which is compute intense) or whether the thumb is stored additionally (which is rather storage intense), however the user does not recognize the difference. Such re-representation capabilities work with any data format or object metadata, and they are offered e.g. by Fedora in its Content Model Architecture (CMA) [148], and in aDORe where they are called Digital Item Method (DIM) [99]. Apart from repositories, web content management systems like Cocoon have been offering a similar mechanism for many years, in which e.g. the web server merely stores an XML representation of a web resource and the Cocoon publishing workflow delivers HTML, PDF, or any other format as requested. [265]

- **characteristics** – To manage digital objects over time, they must be described with suitable metadata. While other forms of application-oriented metadata can be opaque, these characteristics must be processable by the repository. There are various aspects to this, including significant properties for preservation [182],

rights, and repository handling.

The value of a digital object is one of the characteristics, that define the handling of the object in the repository. Unique objects or objects that were particularly expensive in their creation may receive more attention by repository management than objects that can be recreated at minimal costs. Redundant storage in geographically distributed places or storing them in particularly trusted data centers are some of the conceivable measures to ensure retention of an object. iRODS calls these kind of handling instructions "policies" [282] and suggests the standardisation of policy metadata. However, storage policies may vary between repositories: one repository may create six copies of a high-value object on commodity hardware and check their integrity once in a while, whereas the other repository may opt for two copies on high-end RAID systems and check them once every Saturday morning. Therefore, we argue that rather than describing the policy ("redundant storage, 3 copies") it is the value that needs to be described as a distinctive object characteristic. Overall, however, whatever the actual data fields are, they should be recorded into the object characteristics.

We are not aware of standards for object characteristics, although all repositories deal with them – implicitly or explicitly. The administrative metadata section in Dublin Core – an obvious place to look for system-independent metadata – is a good starting point, yet it is targeted towards web resources that do not need to be managed. For example, rights for publicly available web resources differ from the rights typically encountered in repository-based research environments with private data. Or the field *handling* in Dublin Core focuses on web harvesting.

As a requirement this means that repositories need to be capable of processing digital objects for redundant storage, integrity checks, and others. Of those repositories surveyed iRODS is currently the only one, which offers an extensible framework for implementing such low-level processing. It remains unclear whether the repository community will agree on a shared understanding of object characteristics, or whether this remains to be dealt with on an individual level. It is mentioned here as an open challenge for its organizational implications on open repository environments.

These features of digital objects as listed above can be identified in repository systems,

however not all today's repositories serve all these features. While there is a general consensus that repositories manage files and their metadata, not even that is a given as some repositories (Fedora, aDORe) allow objects without files or with only a reference to an externally managed file. However, even though these characteristics are not minimum requirements for repositories, they give a good idea of the overall capabilities repositories are likely to serve.

Ideally, a digital object has all these elements stored in its body rather than dispersing them to various locations (e.g. to file storage, metadata database, reference system, etc). Such self-contained objects are building blocks of robust, distributed architectures, where the objects are always the primary reference, rather than having pieces spread to an intricate network of services creating a puzzle that only a specialised higher-level service is in place to resolve. This is particularly of interest for initiatives with a long-term view, since even highly reliable systems are likely to fail or need to be exchanged against novel technologies when planning for a time frame of decades. [242, 317] Self-contained objects potentially facilitate a migration between different repository platforms.

2.2.2 Attributes of Open Environments

A key objective of this thesis is establishing a repository environment that is "open", meaning accessible and extensible for distributed agents (e.g. repository systems, registries, visualisation tools) in a decentralized manner. This section looks more closely at the attributes of "open environments". Eventually, the reference architecture for digital repositories must enable the implementation of those attributes.

Tharam Dillon et al. identify the following three features as general architectural design principles for open environments: loosely-coupled, simple, and decentralised (cf. the "Evolutionary CUBE", [134]). Frank Buschmann et al. support this with two similar attributes for quality interfaces (with respect to Interface Partitioning): "expressiveness and simplicity", as well as "loose coupling and stability". [106]

1. *loosely-coupled* – "The core principle behind loose coupling is to reduce the

assumptions two parties (components, applications, services, programs, users) make about each other when they exchange information." (cf. [191], page 10) Reduction of assumptions comes in many interrelated facets ([260] identified 12 such facets). When optimising along these facets and thus raising the degree of loose-coupling, systems become more extensible and have the potential to grow and scale rapidly – characteristics displayed e.g. by the RESTful architectural style. [151]

2. *simple* – Simplicity manifests in a focused set of capabilities and stripped-down interfaces. This may be achieved e.g. by pruning complexity, by taking assumptions between the two parties (which works against the previous point on loose-coupling), or by decomposing complexity into simple modules moving complexity from a single service to the overall system.

3. *decentralised* – Both, loose-coupling and simplicity further the independence of individual components, avoid lock-in into a specific component and enable the components to evolve independent from each other. This applies for interaction between specific components in a designed system, and it equally applies for external components. It is the link between internal and external components that is changing as repositories embed external infrastructure and added-value services. Vice versa, an open (i.e. loosely-coupled, simple, and decentralised) design allows repository-based applications and other components to interact with an existing system, thus enabling its anarchic growth. Following the mantra of the Common Repositories Interface Group (CRIG) [47]: "The coolest thing to do with your data [and services] will be thought of by someone else."

These three values are the basis for moving from a single integrated repository system to a larger, open repository environment, since they facilitate the interaction of multiple, decentralised agents (repositories, added-value services, repository-based applications, etc.). However, the attributes are guiding principles rather than "absolute" prerequisites. For example, even the REST architectural style – which is considered to foster loose-coupling [231] – fails on some points in being loosely-coupled. [260]

2.2.3 Layers of the Repository Architecture

This section describes the repository reference architecture. This architecture represents a common denominator with regard to the functionalities provided by existing approaches, and identifies the interoperability channels of the Open Storage and Federation Interfaces. The following chapters build on this architecture and analyse the interoperability interfaces in detail. As we will argue, this interoperability architecture is capable of resolving the scenarios presented in section 1.1, and establishes an Open Repository Environment.

The repository layers and their tasks introduced in the following are designed to be a generic reference and common denominator between disparate repository systems. Likewise, the architecture at hand is not designed to prescribe the "right" repository architecture, and individual systems will have their own ways of looking at repository architecture. To put it differently, we are not designing a new repository but aim to better connect existing ones.

The architecture was derived from a comprehensive survey of existing technologies summarised in section 2.1, as well as discussions in the repository [85, 90], the digital library [89], as well as the grid community [90].

Among various approaches to software architecture in distributed environments [106], a layered architecture, which clusters key architectural concerns and capabilities into layers and defines interaction protocols between layers, is well suited for ensuring separation of concerns and it allows individual layers to evolve separately. At the same time both the service-orientation paradigm [16] of grid and environments ("Everything is a service", [161]) as well as resource-oriented architectures [151, 274] of web environments are compatible with a layered architecture. Practitioners underline the robustness, scalability, and flexibility of such a combination. [5, 303] Most of all, this approach allows the combination in separate layers of approaches from e-Infrastructure and the repository community where they excel most: virtualization of storage infrastructure and virtualization of information resources respectively.

Based on these considerations, the repository reference architecture (cf. figure 2.3)

consists of three layers: virtualized storage at the bottom, upon which a layer for digital object management mediates to end-user applications. Note that none of the layers is called "repository", since the repository really is distributed across the layers. Each layer adds another level of abstraction to the content named as *physical*, *logical*, and *conceptual*, which is inspired by Thibodeau [304] and is reflected in the Federation interface as well (cf. section 4.2).

The interfaces between the layers are more than merely conceptual borders of an architectural concept. It is these interfaces that enable mixing various repository components and external services in a decentralized manner to form a single repository environment. This means that an infrastructure for repository storage could serve multiple object management layers or vice versa, and equally any external service or end-user oriented application could build on one or many infrastructures and object management layers. This is essentially the kinds of scenarios presented in section 1.1.

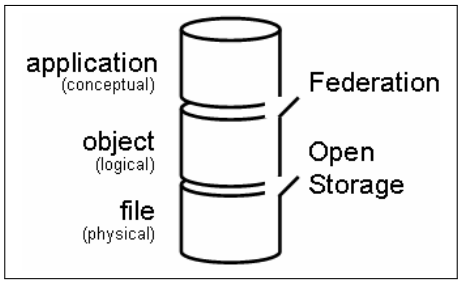


Figure 2.3: Schematic Repository Reference Architecture consisting of 3 layers (*file*, *object*, and *application* in rising abstraction), as well as two interfaces between the layers – the Open Storage and the Federation interfaces, which are the key interoperability channels of Open Repository Environments.

Layer 1: File

The bottom-most layer provides a trusted infrastructure for storing digital objects serialised into files over time.

In addition to the technical interface, a contract with storage infrastructure is characterised by service level agreements that address features like reliability, scalability, and availability. Redundant storage of the data and regular integrity checks

are essential for ensuring the stability of the data (i.e. "bit preservation"), and redundant storage on different types of storage media may also impact on access times. To support this, digital objects may be annotated in their *object characteristics* with information about their value or usage patterns. These metadata help infrastructure decide about e.g. the level of redundancy to ensure file integrity, whether files need to be available for fast online access in hierarchical file management, and other potential aspects of lifecycle management [154]. Eventually data needs to be recoverable and exported in case of disaster, decommissioning of the infrastructure, or whenever the infrastructure provider needs to be changed.

An aspect that may be supported by infrastructure is integrity of data operations. However, inspiration by ACID-compliance in database systems may be infeasible. ACID is an acronym for four properties – Atomicity, Consistency, Isolation, Durability – that transactions in database systems need to display, to ensure integrity even in the face of multiple parallel database accesses and even in the face of failure of some of these transactions. [315] To achieve these properties, database systems provide locks on information and logging that enables roll-back in the case of failure. All this is infeasible in repository environments, where documents could be open for editing by an author for a whole day, or the infrastructure may distribute pieces of a video over various locations to enhance scalability. Eventually, consistency mechanisms need to be tailored to the specific context. In a text-based environment it may be sufficient to inform whenever another user is editing the same document or provide mechanisms to merge distinct versions. However, infrastructure may provide mechanisms to log parallel accesses (with time-outs) or to even temporarily lock specific files for write or read operations (cf. locking and pinning in SRM [176]).

While storage infrastructures like data grids or clouds strive to be as generic as possible to serve just any kind of data, the technical setup of the infrastructure may entail specific reactions on an application level. For example, storage may be only writable once or may be very slow on re-write, may be geared towards files of a minimum size, or the like. These storage constraints need to be dealt with on a higher layer.

Open Repository Interface: Storage

The Open Storage Interface enables access to stored digital objects in a storage-independent and management-neutral manner. It offers a CRUD interface for creating, reading, updating and deleting object serialisations, yet does not go far beyond that level in terms of functionalities, in order to remain generic and scalable. Chapter 3 analyses the Open Repository Interface in detail, and shows its capability of resolving the storage scenarios presented in section 1.1.1.

Logical file management can be organised along various lines and existing repositories differ considerably in that. Aspects of logical file management include (1) storing the files in a hierarchical directory structure, (2) encapsulating the files together with the object's metadata in an XML container (e.g. Fedora uses FoXML/METS, aDORe uses MPEG-21 DIDL), or (3) binding various files together in a single package for immutable mass storage (e.g. aDORe uses the ARC file format). These idiosyncrasies are detrimental to openness, as only a specific repository software is capable of interpreting the files in the storage infrastructure. Also it affects negatively the scalability of the overall system, since any access needs to go through the repository software, even if software infrastructure is distributed.

The Open Storage Interface aims to provide a generic interface to digital objects for CRUD operations on a storage-level. CRUD operations are the minimal set of operations required for handling objects, both internally or by an external agent [262, 325] A suitable standard format to expose digital objects is e.g. OAI-ORE. OAI-ORE has been conceived as a *lingua franca* for object representation and is suitable as an open storage format even though originally conceived for federation-level activities. Also its Atom representation is conducive for enabling some of the patterns discussed in chapter 4.

In addition to the object interface, the open storage interface may provide some Federation interfaces to facilitate low-level virtualisation of repository contents. This includes e.g. a list of all the collections and objects contained in a storage vault, to facilitate harvesting and reconstruction of repository contents; as well as an event mechanism that exposes OAI-ORE-based Atom-feeds on CRUD events (Hybrid

Notification). Chapter 4 discusses these and other Federation mechanisms.

Layer 2: Object

The object layer manages the digital objects from one or more storages, and is capable of handling object workflows adequately: e.g. ingesting, updating and versioning, and (re-)packaging digital objects.

One of the key tasks of the object layer lies in binding the various aspects of digital objects together (i.e. metadata, files, relations, characteristics), ensure their consistency, and to enable the user to retrieve and process objects. This may entail that e.g. metadata and relations are indexed in specialised databases, which provide efficient access and additional functionalities such as filtering for metadata elements. However, the object remains self-contained and holds all the relevant information adequately linked within file storage; any index data is a copy and the object layer ensures its consistency in the case of updates.

Ideally the object layer is in the place to redirect many of its tasks to (external) services rather than passing requests up and down the layers. For example, reading files from storage could be redirected directly to the Open Storage Interface, potentially to an off-site data center. Re-representation behaviours could be triggered by the object layer, yet conducted externally at the server hosting the re-representation service. This measure not only distributes the tasks of the object layer, thereby fostering scalability, also many such services could be re-used by various repositories or other agents.

Open Repository Interface: Federation

Federation mechanisms lie in between the object and the application layers. The Federation Interface actually is a cluster of mechanisms to achieve interoperability between diverse agents in an open repository environment. These mechanisms are capable of interweaving multiple repositories, respectively of enabling interaction between repositories and other agents.

Digital objects are often of interest in multiple contexts: publications may be disseminated through institutional as well as thematic repositories; research data may be created in a specific project and later re-used in another, maybe inter-disciplinary or inter-institutional project; and many other such situations are conceivable.

A more detailed discussion in chapter 4 analyses Federation and develops novel federation mechanisms. In this section, we list three federation activities to give an idea of the kinds of environments enabled by federation.

- The most prevalent use case for federation as yet is search across multiple repositories. Today many universities have their own institutional repositories. Initiatives like Dare [290] and Driver [149] establish central portals to search for publications on a national respectively European level. Other than the federations of research publications in Dare and Driver, the Europeana [21] initiative addresses research data and aims to pool all digitisations of cultural material in Europe.

- SDMX, the Statistical Data and Metadata eXchange, [125] is "an initiative to foster standards for the exchange of statistical information", sponsored amongst other by national statistical offices, the World Bank, and the United Nations. Amongst the challenges for SDMX is the requirement to accommodate partners in remote areas (e.g. Africa), and to ensure that any updates even in remote countries are immediately propagated throughout the whole federation of global partners. To enable interoperability, SDMX includes metadata schemas as well as guidelines for web services [124] to interconnect statistical databases around the world.

- The project TIPR, Towards Interoperable Preservation Repositories, federates preservation repositories including three university repositories based on heterogenous software platforms. As part of this federation, digital objects are replicated and distributed to the dispersed repositories. TIPR's goal is to ensure the longevity of the digital object. [108]

An analysis framework in chapter 4 distinguishes three levels of interoperability that need to be covered for both the object to be exchanged between two distinct agents as

well as the overall information system that accommodates the interaction (cf. figure 4.1). In addition to this federation framework, the chapters 4 and 5 show the capabilities of Federation techniques to resolve the federation scenarios presented in section 1.1.5.

Other than in the case of the Open Storage Interface, there are various attempts for federating repositories on an object level. However, as the scenarios (cf. section 1.1) underline, current approaches are fragmented and often insufficient for contexts other than open access publication repositories. While there will never be a single, final solution to repository federation, we discuss an extended federation model in chapter 4. At this point we would only like to mention the two orthogonal types of federation and two prototypical and popular federation protocols.

Federated content – Combining multiple repositories in a single application increases both the exposure of the objects as well as the value of the application. Therefore, federation protocols have been created independently in various communities, including the following. Apart from protocols, metadata sets like Dublin Core [264], encapsulation formats like METS [238], schemas like PREMIS [17] or other standards are of relevance when federating repositories. The relation between protocols, formats, and others is discussed in chapter 4.

- Z39.50 [200] for querying library catalogues has been developed in 1988, and became a NISO standard in 1992. The protocol was widely spread and still is. Its successor SRU/W [132] better suits the current web environment, and it is embedded in ongoing work for extending search/retrieve interfaces.
- The Protocol for Metadata Harvesting, OAI-PMH, [198] was first released in 2001 to connect disparate library catalogues. Spurred by the open access movement [3] it quickly became a *de facto* standard. In September 2009, OAIster, a "union catalog" for digital resources [50], cross-referenced more than 1100 repositories by way of the OAI-PMH protocol and their more than 23 million digital resources. Apart from harvesting publications, OAI-PMH has been employed in other contexts as well ([113, 235, 276]).
-

Multiple applications – Embedding objects in multiple applications environments – the orthogonal federation mechanism to embedding objects from multiple repositories in a single application – has not found as much attention as its counterpart. Many repositories today offer interfaces or programming libraries to build custom applications on top of the repository infrastructure. However, there are as yet no standards that would enable an application to move from one repository platform to another. It may be argued to which extent that is useful and there will likely always be custom interfaces, yet some aspects may be covered by standards to ensure portability where needed.

The newly issued OAI-ORE standard covers one aspect of this: object representation. OAI-ORE is a format specification for serialising digital objects expressed in RDF-based Resource Maps. Version 1.0 of OAI-ORE has been released in October 2008. Being the cousin of OAI-PMH, OAI-ORE has much attention guaranteed. Some of the future use cases it mentions include "applications that support authoring, deposit, exchange, visualization, reuse, and preservation." [222]

Layer 3: Application

Applications can build on all aspects of a digital object, alone or in combination: its metadata, files, as well as its relations and re-representation services. This section only briefly underlines the versatility of the conceivable applications in a repository environment, since essentially the Federation interface has to provide the adequate techniques and sockets for this. This is reflected by the outline in the previous section, and is further analysed in chapter 4.

Applications can be as diverse as an image archive; an office platform for collaborative editing; or a teaching environment. Ideally a user does not recognise on what kind of information infrastructure an application is built, be it a single repository, multiple repositories, or other agents. Actual scenarios of federated applications are presented in section 1.1.5.

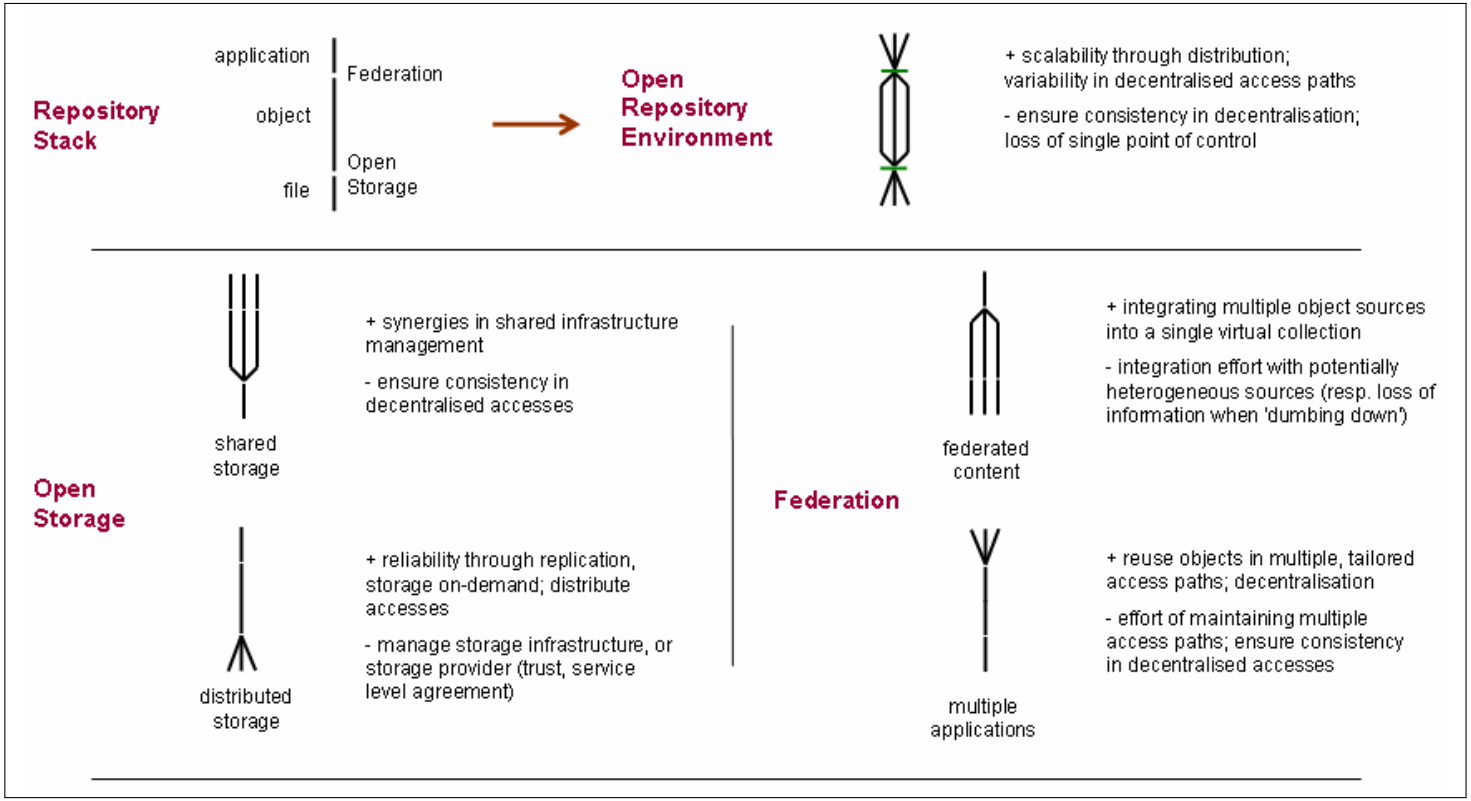


Figure 2.4: Repository interfaces: analysing pros and cons for each interoperability scenario (Open Storage = file/object layer left; Federation = object/application layer right), as well as combined in an open repository environment (on top)

2.3 Discussion

This chapter has looked at precursor systems and related fields to digital repositories. Section 2.1.3 mentions a broad range of systems related to repositories in one way or another. Repositories have never been subsumed in a single coherent field with a sharp profile, and there are no indications that this will happen (in the near future). However, the ongoing convergence between repository systems and e-Infrastructure defines this thesis.

The repository reference architecture presented here aims to establish interoperability channels between existing repository systems as well as other agents, with regard to all features of digital objects including files, metadata, relations, and characteristics. The architecture does not attempt to define individual system components or any more fine-grained functionalities in order to remain generic. As a common denominator, it identifies two interoperability channels: the Open Storage Interface as well as the Federation interface. These two are the glue for the three-layer repository reference architecture, and as we argue in the upcoming chapters, these two layers are the fundamental building blocks of Open Repository Environments.

This architecture is the basis for the analysis and experimentation presented in the following chapters. Chapters 3 and 4 look at the Open Storage Interface as well as the Federation interface respectively. The TextGrid project described in chapter 5 implemented these concepts, and embedded them into an organizational and social context. Furthermore, these chapters also describe how the scenarios presented in section 1.1 are implemented by way of the Open Storage and Federation interfaces, thereby illustrating the rol of the repository reference architecture presented in this chapter as a blueprint for Open Repository Environments.

3 S3-like: A Model for an Open Repository Storage Interface

This chapter analyses the open storage interface of the repository reference architecture as defined in section 2.2 above, thereby vindicating it. In a series of experiments also discussed in [88] it analyses differing technical approaches regarding their capacity for implementing the attributes for "open environments" defined in section 2.2.2: loosely-coupled, simple, and decentralised.

The experiments were not defined by the specification of the open storage interface (cf. section 2.2.3), although the findings of this chapter resonate with that specification thereby further supporting the reference architecture. The analyses in this chapter also refine the conceptual specification of an open storage interface with a more technical analysis and hence guidance for implementation.

The series of experiments described in the following tests the integration of the prevalent repository software packages DSpace and Fedora with storage virtualisation employing the grid software packages Cleversafe [33], iRODS [267], as well as a RESTful abstraction [151, 274] upon the Storage Resource Manager SRM [176] grid standard.

Each of the storage virtualisation mechanisms employed works on a slightly different level. The analysis framework employed in this chapter is illustrated in figure 3.1. It combines the functional capabilities from the software packages Fedora, DSpace, iRODS, and Cleversafe, which were identified in section 2.1 as representatives from the repository and e-Infrastructure fields respectively: the application-orientation in

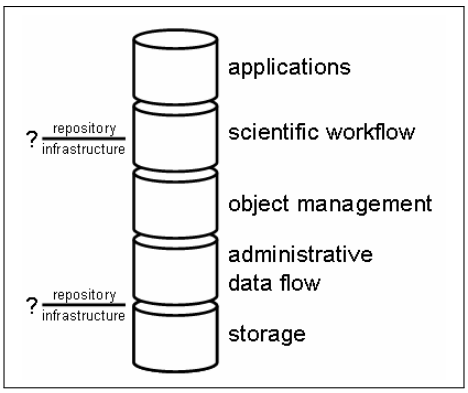


Figure 3.1: Layer model of the shared functional components of repositories and storage infrastructure, as a framework for the analysis of the open storage interface. The illustration follows the model of figure 1.2.

DSpace [281], Fedora's approach to embedding scientific workflows [130] and object modelling [148], the administrative 'microservices' in iRODS [267], as well as storage virtualisation in both iRODS and Cleversafe [33].

Like Brad Wheeler in figure 1.2 we ask where to draw the line between shared infrastructure and higher-level applications or, as in this case, dedicated repository services. Each experiment varies as to where infrastructure and repository services interface, including (a) the minimum level of a storage infrastructure mounted as a virtual file system (cf. the Cleversafe experiment, section 3.1); (b) a high-level solution that includes administrative services, and to some extent also metadata management and higher-level services (cf. the iRODS experiment, section 3.2); as well as (c) a novel approach that remains between those two former experiments (cf. the S3-like experiment, section 3.3).

The analysis finds that in order to fulfil the attributes 'loosely-coupled', 'simple', and 'decentralised' (cf. section 2.2.2), integration needs to be on an intermediate level – not merely a virtual file system, yet neither a monolithic block that includes all the functional components. In the following, along with the analysis of each of the approaches an icon of figure 3.1 indicates the level at which the experiment puts the infrastructure line. We also discuss how an open storage interface relates to other existing mechanisms such as WebDAV [171] or commercial cloud offerings including

Amazon S3.

Please note: the RESTful abstraction is inspired by the interface of the Amazon S3 storage cloud service [1]. Amazon in its 2009 license agreement explicitly disallows re-engineering of the S3 API. "You may not, and may not attempt to, reverse engineer, disassemble, or decompile the Amazon Properties or the Services or [...]" (Amazon Web Services Customer Agreement, May 20, 2009). The experiment described here was conducted earlier, with a license agreement in place that did not have this paragraph. However, our future work will not build on the Amazon API, but may build on one of the emerging cloud standards [51, 64, 286].

3.1 Cleversafe: Low-Level, Transparent Storage

This section describes an experiment, in which a Fedora repository is installed upon a Cleversafe data grid. Distributed over 5 nodes, Cleversafe is mounted as a virtual file system, thereby offering a low-level interface with minimum exposure. Since the repository does not detect a difference whether the storage is local or remote, it can be installed without any adaptation. However, the section concludes that while very *simple* from a system administrator's point of view, the approach fails with regards to the attributes '*loosely-coupled*' and '*decentralised*', and it consequently is insufficient for an open storage interface.

Cleversafe dubs itself a 'dispersed storage' network, which was available in version 1.1 at the time of writing (April 2009). Cleversafe is open source software developed by a company as their key product. At the basis of the Cleversafe software is an algorithm, which chunks data into pieces, spreads them over distributed data nodes, and performs error correction for fault tolerance (a Reed-Solomon Code). This algorithm displays stability in the face of failing nodes, good read performance from redundant nodes, and increased security as single nodes only merely accommodate encrypted data chunks (cf. [287]). A Cleversafe storage 'vault' can be mounted via iSCSI [241] and hence works as a virtual file system.

We used an early version of Cleversafe (Version 0.7.8, November 2007) and installed six

virtual CentOS 5 nodes for the storage network. This purely experimental setup, in addition to the fact that we used an early version of the software resulted in rather slow access rates – hence, performance is no criterion in this experiment. Cleversafe was used to store the digital objects of a Fedora 2.2 repository, running on Ubuntu Linux. No adaptation of the Fedora repository was necessary for this.

Regarding the functionalities defined in figure 1.2, Cleversafe offers virtualised storage in that it promises to scale to any storage size and it enables distributed storage management. However, a single Cleversafe network can only be shared read-only, since multiple iSCSI initiators writing data at the same time could compromise data consistency. Hence, a read-only journaling configuration with fail-over can be installed easily, yet it is impossible to provide multiple read/write interfaces to enable distributed ingest on an infrastructure level. In other words, a specific repository installation is tied to a specific Cleversafe deployment.

Cleversafe fails to offer any functionalities above virtual storage, such as administrative workflows or metadata management. At the same time, since a specific repository and a specific Cleversafe installation are coupled together, these functionalities must be offered by the repository software. Due to the distribution algorithm and the subsequent dispersing of the files in small pieces, it is not possible to adapt Cleversafe accordingly.

Overall, this approach offers a storage interface that is tightly coupled and is just above the virtual storage without administrative or any higher level functionalities. Consequently, it deviates from the reference architecture in various ways: it neither satisfies the attributes to be '*loosely-coupled*' and '*decentralised*' (cf. section 2.2.2), nor does it fulfil the functionalities specified by the open storage interface (cf. section 2.2.3).

3.2 iRODS: High-Level Storage Infrastructure

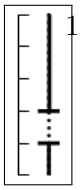
iRODS combines in a single, integrated architecture functionalities of a data grid, a repository, as well as various user application environments such as a web client. This section demonstrates the various levels, on which repositories such as DSpace and

Fedora could interact with iRODS. In order to obtain the scalability of e-Infrastructure (iRODS) and on top the versatility of repositories with regards to semantic modelling and user applications (DSpace, Fedora), we identify a level as suitable that is just above storage and administrative infrastructure tasks. However, all levels analysed fail to enable multiple, decentralised agents to interact with minimum integration effort and without causing potential inconsistencies between the heterogeneous systems involved.

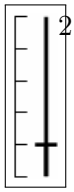
Where Cleversafe focuses on a single functionality, storage distribution and virtualisation, iRODS is functionality-wise clearly at the other end of the spectrum. iRODS goes beyond the functionalities offered by storage virtualisation in a data grid. [266, 313] So-called *rules* for triggering microservices allow comprehensive adaptation of administrative workflows and hence of tailoring the data grid to the respective requirements. In addition to low-level data grid and administrative functionalities, iRODS offers specialised graphical applications such as an AJAX-based web interface. As such the iRODS data grid aims to offer the whole continuum of repository functionality, from low-level data management to the user interface.

Despite this broad spectrum of activities, iRODS has not yet comprehensively addressed typical repository processes such as ingest procedures for authors, more sophisticated object modelling capabilities, or embedding in scientific workflows and other repository-based applications. For example, the DSpace repository offers a comprehensive user community model that has not even a rudimentary equivalent in iRODS, and Fedora offers more advanced metadata and content modelling mechanisms. Several projects are hence looking into combining iRODS with higher-level repositories such as DSpace or Fedora. [96]

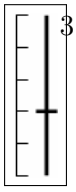
Various approaches on how to connect iRODS as an infrastructure with repositories on a higher level are conceivable [88, 91, 326]). The following paragraphs describe these approaches, whereby the wrapped images are icons of the functionality levels defined in figure 1.2:



1. **iRODS objects as external datastreams** – some repositories including Fedora are capable of managing the metadata of digital objects that are outside of their stores. While this allows for referencing objects stored in an external iRODS data grid, the repository has no means for managing the object (audit trails, versioning, etc).
– **Therefore,** iRODS fulfils the requirement of scalable, distributed storage. However, administrative control and object management capabilities are lost, and the coordination between iRODS and the repository requires additional overhead.



2. **using iRODS as a repository storage module** – instead of storing data locally on the repository server, iRODS is plugged into the repository as a storage component. The Jargon Java API [45] for iRODS is used for directly implementing the storage handler into the repository. DSpace offers support for this approach as part of its general release. Furthermore, in a joint effort the DSpace and Fedora teams aim to develop a generic storage handler with a plug-in mechanism [12] to accommodate iRODS and just any other storage handler.
– **Therefore,** despite the fact, that this form of integration is configurable for out-of-the-box DSpace and Fedora repositories, it fails to use iRODS rule management and its administrative control flow. Essentially, iRODS is used as a virtual file layer with the same caveats as displayed by the Cleversafe experiment (cf. section3.1).



3. **iRODS microservice and rule support** – one step further from a simple virtual storage, iRODS could define rules for parsing a newly deposited object on ingest, extract the metadata into its ICAT database, and hence activate its low-level rule support. This does not demand any adaptations in either iRODS or the repository, yet demands a higher level of coordination between them, since metadata management is partly redundant and must be synchronized. Behaviours on the iRODS- and the repository-level must not interfere with each other.
– **Therefore,** this integration approach certainly makes the best use of what functionality is readily available in both iRODS and the repository. The redundancy necessitates a level of coordination between the two systems that may

not be feasible in all usage scenarios, e.g. scenarios where objects change frequently. Before this approach can be implemented, the minimum metadata that needs to be duplicated must be identified.

4. **integrating iRODS into the repository landscape** – since iRODS is offering the complete stack of repository functionality up to user interfaces, iRODS could fully integrate into the repository standards landscape and connect with other repositories using federation standards (cf. section 4.2). – **However,** iRODS is not currently offering standards-based interfaces. Even if standards-based interfaces for iRODS were on the horizon, shifting the infrastructure/repository line up to the application level may have negative implications on the scalability and consistency of the repository storage infrastructure.

These four approaches exemplify integration levels of iRODS-based infrastructure and repositories. Please note the dramatic difference between the first and the last pattern with regard to openness, with the last clearly being the most open approach. However, as the evaluations beneath each approach indicate, the approaches 2 and 3 are the most feasible ones. Which of them to choose depends on the complexity of the usage context: is it sufficient to revert iRODS to a virtual file system (approach 2), or are administrative functionalities needed to e.g. enable data replication or preservation functions [316] (approach 3).

Both, approach 2 and 3 build on a client/server communication between the repository and the infrastructure using the Jargon library [45]. However, the iRODS interfaces are largely undocumented (though the code is available) and the Jargon library ties them into a Java-based software environment. Even if somebody would re-engineer the interface to iRODS to enable diverse repositories or other agents to plug into an iRODS infrastructure, the file and metadata management in iRODS cannot be tapped such that diverse agents can work on the same objects in a decentralised manner. While ways to adapt the iRODS interface accordingly are conceivable, previous attempts to establish gateways between iRODS and other systems have lacked support from the iRODS developers [22]. In other words, currently existing interfaces to iRODS are neither sufficiently loosely-coupled and decentralised, nor are they organisationally

open. For these reasons iRODS/Jargon fails to fulfil the requirements put forth for open repository storage.

Apart from this it should be noted, that systems stemming from the e-Infrastructure (iRODS) and the repository communities (e.g. DSpace, Fedora) are starting to overlap functionality-wise. Overlapping technologies are not problematic as such. However, the two communities lack common concepts and standards, thus failing with regard to systems interoperability and also duplicating efforts.

3.3 S3-like: A RESTful Intermediary for Open Storage

The previous experiments have failed to satisfy the requirements for open storage with regard to *loosely-coupling* and their support for *decentralisation*. This section develops such an interface based on the model of cloud storage, specifically Amazon S3. The interface abstracts from the implementation of the storage infrastructure: e.g. local storage, SAN storage of a data center, or a distributed data grid. In the actual experiment, we use the SRM grid middleware as storage infrastructure, abstract it through a cloud-like interface that is inspired by Amazon S3, and deploy the DSpace repository software on it.

This section also shows that open repository storage differs from generic storage infrastructure (e.g. WebDAV, Amazon S3) in its support for file and metadata management that is specific to repository environments.

3.3.1 A concept for SRM-based repository storage

The Storage Resource Manager (SRM) [176] stems from the high energy physics community where it serves as one of the grid middleware components to distribute the massive data influx from experiments such as the Large Hadron Collider (LHC) at CERN [30]. SRM is a grid standard in development, and already employed in huge systems (e.g. SRM/dCache [143], SRM/Castor [110]). As a standard protocol for initiating transfers between storage resources, the SRM is capable of mediating between

storage components of various types, transfer protocols, and other grid components. To sustain the large amounts of data from experiments in high-energy physics, SRM is geared towards performance and works on a block level, providing low-level handles on files and storage space. Specialized functionalities include pinning of files (i.e. reservation for a defined time), storage space reservation, and others.

We first evaluated whether SRM could be plugged into a repository as a storage handler, similar to how iRODS can be plugged into DSpace (cf. previous section). However, this turned out to be impractical due to the tight link between the SRM and the overall grid environment. This coupling between SRM and its environment shows in several ways. Since SRM is not a transfer protocol itself but a mediator, transfer protocols such as GridFTP, DCache, or others are required as well. When operating in a grid environment, the user needs a grid certificate as well as suitable authentication mechanisms based on the Grid Security Infrastructure (GSI) [159] to authenticate. Hence the certificate as well as security-related protocols have to be available at the client. The officially supported client library called GFAL – Grid File Access Library [44] – is based on C and Python, and the package '*lcg_utils*' provides convenient command-line tools. However, it expects software packages from the grid distribution gLite [223], which is ultimately best installed on a Scientific Linux operating system [23].

As an interface between a grid node (SRM) and the web-based repository (DSpace), we looked for a more lightweight interface that is capable of translating between the two worlds. Usage patterns for repository-based applications clearly differ from those needed in scientific infrastructure. Repository-based systems are often targeted at non-expert users and tie into their common usage environment – currently the web. Performance requirements are absolutely central in the latter, whereas web-based tools may compromise on some of the tuning parameters for the benefit of simplification and efficient communication over heterogeneous systems.

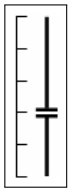
Existing cloud services are a premier model for translating between storage infrastructure and web applications. Cloud services often – like in the case of Amazon [25] – offer both, REST- [151, 274] and SOAP-based interfaces [175]. Despite their simplicity – or rather, because of their simplicity, REST-based protocols often

satisfy the needs of web applications. We are therefore looking for a RESTful interface inspired by cloud services, which is capable of translating between storage infrastructure and repositories.

So why not use a cloud service like the commercial Amazon S3 from the outset? Using a commercial cloud service is an option, of course. Yet, most of the research data we are holding is unique and valuable. While Amazon promises multiple copies of each file and an uptime of more than 99 percent, we do not know whether Amazon will still be there and offer on-demand storage in 20 years. To overcome this, e.g. the DuraSpace project [117] or the EPrints storage layer [294] aim for replicating data between multiple clouds and other storage providers thereby spreading the risk. For those institutions with substantial storage resources available locally or within their national computing infrastructure, however, creating their own storage cloud may be more feasible. The additional control over the infrastructure, both organizationally and technologically, is conducive to long-term costs, trustworthiness, and opens up the opportunity for adding other functionalities into the infrastructure.

3.3.2 Implementing a storage cloud based on the SRM grid protocol

This section describes the implementation of 'S3-like' cloud interface, which establishes the link between the repository and its storage infrastructure. The hardware and operating systems of those two components are completely separate, communicating via HTTP.



As a proof-of-concept, we decided to re-engineer the REST API of the Amazon S3 storage service [1] as an interface between the grid and the repository. Our experimental implementation of the S3 interface uses Python WSGI (Web Server Gateway Interface) [139]. Existing S3 libraries like Jets3t [46] and respective tools can be re-rooted from the Amazon cloud to our re-engineered interface. This includes e.g. the DSpace repository, for which a storage handler implemented upon the Jets3t library has been developed [36].

The *S3-like* service can be launched on any WSGI-enabled web server, and was tested on the CherryPy [31] standalone server for development as well as the Apache web

server for deployment. S3-like was designed with a plug-able storage interface. Once the local service was created and tested, we embarked on plugging in the SRM grid protocol as well.

The SRM storage module for S3-like employs the *lcg_ utils* package. Therefore the S3-like experimentation server has the grid middleware gLite installed for certificate management and all the components required by lcg-utils, in addition to the WSGI-enabled web server. It is physically located in Goettingen, Germany, while the data is transferred via SRM to test servers in D-Grid [249] or See-Grid [57] project. [88]

While the implementation of S3-like is fairly stable and flexible, the SRM storage plugin is merely experimental. Nevertheless, the experiences gained from this experiment were promising. File transfer through the S3-like interface onto an SRM site works, however some S3 features have been disabled, optimisation is still outstanding, and the translation of the security mechanisms – the simple keys in S3 and the asymmetric public key infrastructure in the grid environment (Grid Security Infrastructure, GSI) – remains to be resolved. These deficiencies, however, do not reduce the following positive lessons learnt.

First of all, we were doubtful whether SRM accommodates the specific requirements of repositories. SRM generally deals with large, immutable files. Small files, on the other hand, may lead to an inefficient use of data grid capabilities. [318] To be able to translate between the typical repository content being numerous, small, mutable files typical for repositories and the huge, immutable files expected by SRM, we considered adopting a storage concept like it is used in the aDORe repository. The aDORe repository collates a number of objects into a container and stores the whole batch in permanent storage. The container format is a combination of XMLtape and the ARC format. [309]

While the aDORe-approach is conceivable, it turned out not to be necessary, since the SRM – apart from the initial communication overhead – proved to be sufficiently efficient in serving the repository. In fact, measurements on the test server showed comparable results between SRM, S3-like and Amazon S3. Optimisations are of course conceivable for a production environment. This includes the aDORE-approach, as well

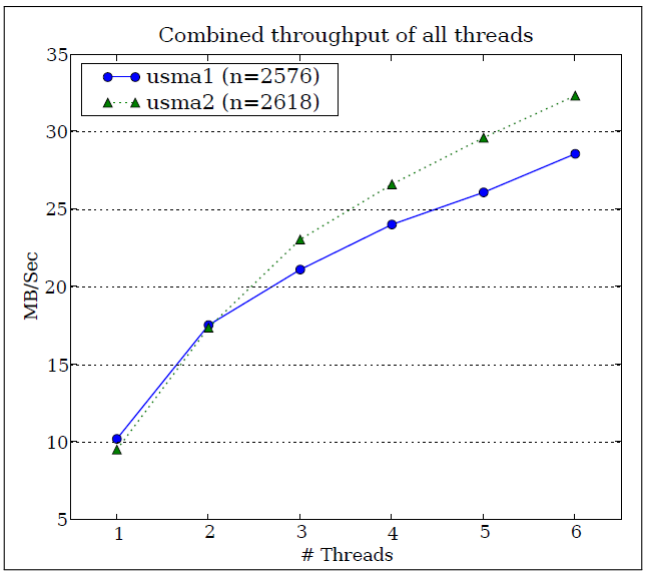


Figure 3.2: Performance of 100MB GETs from S3 for concurrent threads from EC2 servers usma1 and usma2 respectively. [170]

as optimising the communication within SRM. However, that was out of scope for this proof-of-concept-

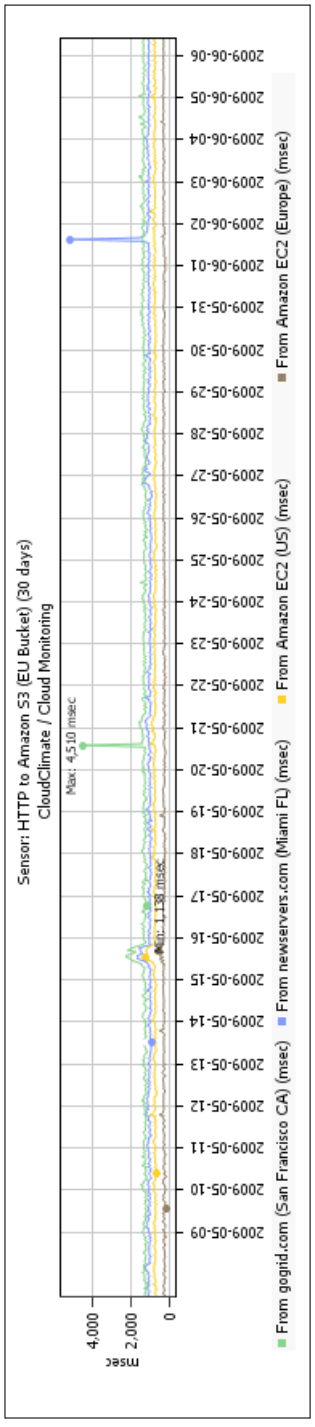


Figure 3.3: Performance of the Amazon S3 service when downloading a 65 KByte file every 30 seconds from the US and EU locations. Taken from CloudClimate, June 7, 2009.

Also in production environments, measurements indicate similar performance characteristics of SRM and Amazon's S3, with a small edge for SRM. For internal data transfer, EC2 indicates a maximum bandwidth of 250 Mbit/s (30 MByte/s, cf. figure 3.2 with measurements between Amazon S3 and EC2 clusters). For SRM the expected data rate between Tier 1 and Tier 2 centres should sustain 300 Mbit/s and more (more than 37 MByte/s). [120]

The performance bottleneck really is not internal processing but rather Internet latency, which is more of an organisational issue than a technical one (i.e. location of data centres, redundancy, etc.). The Cloud Climate [34] shows overall stability of Amazon S3 (cf. figure 3.3), and [193] measures a transfer rate with a fixed cost when storing a file of around 140 ms and a variable cost of about 10 to 12 MByte/s. While 10 to 12 MByte/s may be too slow for some interactive applications as underlined by [76], it is sufficient for object storage in a repository environment. Note that Amazon in its "Design Requirements" [25] did not detail the transfer rate it aims to deliver. Amazon states rather vaguely on its website: "Amazon S3 must be fast enough to support high-performance applications. Server-side latency must be insignificant relative to Internet latency." And Amazon makes use of this vague contract with the user, as e.g. exemplified by an unannounced infrastructure re-configuration in early 2007 and a permanent "marked decrease in available bandwidth from EC2" [170].

This shows clearly that SRM as a backend for a cloud-like interface technically comes up to quality of service requirements, with the main bottleneck for transfer being Internet latency.

3.3.3 Discussion of S3-like

This section described the implementation of a grid-based storage cloud, and demonstrated its implementation based on the SRM data grid standard and a re-engineered version of Amazon S3 as cloud API. The concepts are transferable to other systems, and e.g. a cloud-like interface could equally be layered upon Cleversafe or iRODS.

The loosely-coupled and light-weight cloud interface ensures a separation of concerns between the storage infrastructure and the repository, thereby allowing evolution of the respective systems independently. This covers two of the three attributes for open environments (cf. section 2.2.2), only leaving 'decentralised' to be discussed. Concurrent access of multiple agents to a shared S3-like infrastructure is in principle possible. However, in order to fulfil scenario 1.c (cf. section 1.1.4), those agents must be capable of interpreting the objects, even if they were deposited by other agents. The OAI-ORE format [222] enables a minimum level of semantic interoperability, linking all constituents of a digital object – content, metadata and relations (cf. section 2.2.1) – together at a single location. Thereby, heterogeneous agents can access digital objects in a decentralised manner, and this is also in line with the specification of the open storage interface in section 2.2.3.

S3-like or any open storage infrastructure needs to be adapted accordingly and expose OAI-ORE annotated objects rather than just any file. Storage infrastructure thereby ceases to be generic to just any application, and is tailored to repository environments. This also means that commercial clouds or similar storage infrastructure can only be embedded into repository environments by way of an intermediary. However, to really unlock open repository storage and enable interaction across heterogeneous agents, this shared storage format is necessary.

While not part of the experimental setup, authentication and rights management are important issues that remain to be solved. Existing approaches between grid and repository systems are incoherent, e.g. the simple keys approach in Amazon S3 is incompatible with the PKI-based security mechanism in grid environments, and does not allow for mechanisms like delegation [259], which would be required for scenario 1.1.8.

However, the experiences from the S3-like implementation indicate that the presented bridge between grids, clouds and repositories is in principle feasible. Many more features remain to be explored, such as the configuration options *location* and *storage_class* in the S3 protocol, which could give other starting points for optimisation. Furthermore, giving the user the chance to define the physical location of its data potentially raises the user's trust into the infrastructure. One possible scenario

may be a storage class 'confidential, valuable', which triggers the infrastructure to replicate the asset in three distributed data centres with particularly high security measures, rather than a simple tape backup of a 'standard' storage class.

There is intense discussion in the grid community about offering RESTful interfaces. [204] Ian Foster (the "inventor" of the grid [158]) rightly points out [157] that the unique selling point of commercial clouds goes beyond their simplicity and includes e.g. the billing model (pay by credit card, have it delivered in no time with hardly any administrative overhead).

Apart from huge national and international grid programmes, we may also see institutional clouds emerge in the future, tapping into the opportunities in cloud-like services to simplify an institutions digital infrastructure and for transforming the link between infrastructure management and users. [76] identifies opportunities and challenges in establishing "private clouds" and – adding to its more technological view – we would also like to point out the potentially transformative nature of institutional/private clouds for the user.

On a different note, there has been much discussion about grids versus clouds. The S3-like experiment shows that data grids and storage clouds may interact smoothly. After all, they both follow similar goals in virtualizing storage resources, even though they address different usage patterns (high-performance computing versus general-purpose, interactive applications). From this it seems that merging grids and clouds is but a small step away, and indeed grid resources can be mixed into the world wide pond of mash-ups.

3.4 Discussion

The three experiments presented in this chapter analysed distinct models of storage infrastructure for repository environments, each at a different level with regards to the functionalities they support as part of the infrastructure.

The Cleversafe experiment did not fulfil the requisite attributes for openness identified

in section 2.2.2 – loosely-coupled, simple, and decentralised. iRODS offers a multitude of functionalities out-of-the-box in a monolithic architecture, yet is neither loosely-coupled nor does it foster decentralisation. Our approach, S3-like, realises a light-weight interface as an intermediary between various kinds of storage infrastructure and repositories, and it fulfils all required attributes.

Eventually both iRODS and S3-like drew the line between infrastructure and repository between the 'administrative data flow' and the 'object management' (cf. figure 3.4). The repository storage should expose objects rather than files – e.g. in the OAI-ORE format – to foster decentralisation across heterogeneous agents, yet the actual management of the objects is accomplished outside of the infrastructure (cf. section 2.2.3).

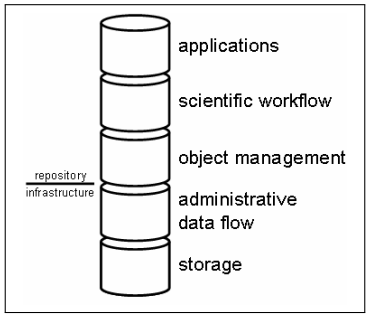


Figure 3.4: Defining the Infrastructure/Repository line for the Open Storage Interface, following the same layer model of functional component defined in figure 3.1.

But how do the results of this chapter relate to the storage **scenarios** put forth in section 1.1.1 – On-Demand Storage, Distributed Access, and Repository Reconstruction? (cf. figure 3.5)

First of all, all three experiments offered storage virtualisation, in that they equip the repository with remote, distributed storage resources that are managed independently from the repository core and can be extended on-demand. How this storage infrastructure is managed is largely an organisational question – e.g. whether repository staff also manage the storage infrastructure themselves, or whether it is provided by the institution, the community, or a commercial provider.

With regard to Distributed Access, both iRODS and S3-like can in theory be accessed

through multiple agents, yet at this stage neither ensures consistency in face of concurrent accesses that may interfere with each other. TextGrid (cf. section 5) and e.g. Dropbox [61] follow a simple yet effective strategy, in that they do not prevent interference on simultaneous 'writes' to the same object, but they tell the user that possible inconsistencies may have occurred. Whether or not that is a suitable strategy depends on the usage context. Since SRM allows locking of files, S3-like can be adapted to prevent inconsistencies in the first place [225].

There is another aspect to Distributed Access apart from parallel access through multiple agents. Up until now, external agents only had access to repository contents through the top-level interfaces of the repository. By way of decoupled Open Storage, external agents can access repository contents through both, top-level repository interfaces as well as the low-level storage interface.

The scenario Distributed Access assumed distributed agents using a homogeneous storage method – i.e. using the same storage hierarchy, file names, and storage format. However, once heterogeneous are involved as in the Repository Reconstruction scenario, the abstraction from the file to a dedicated object format such as OAI-ORE is needed. Therefore, iRODS is incapable of dealing with heterogeneous agents and the Repository Reconstruction scenario in particular, whereas S3-like is an intermediary that may translate between distinct formats and it is hence in the place of dealing with heterogeneous agents.

Overall, this chapter has shown where storage models like Cleversafe and iRODS fall short of fulfilling the requirements outlined in section 1.1.1, and has also shown how a RESTful intermediary could resolve that by being open, loosely-coupled, and by fostering decentralised interactions. While S3-like is only an experimental such intermediary, it is in principle in the place of solving all the scenarios for Open Repository Storage.

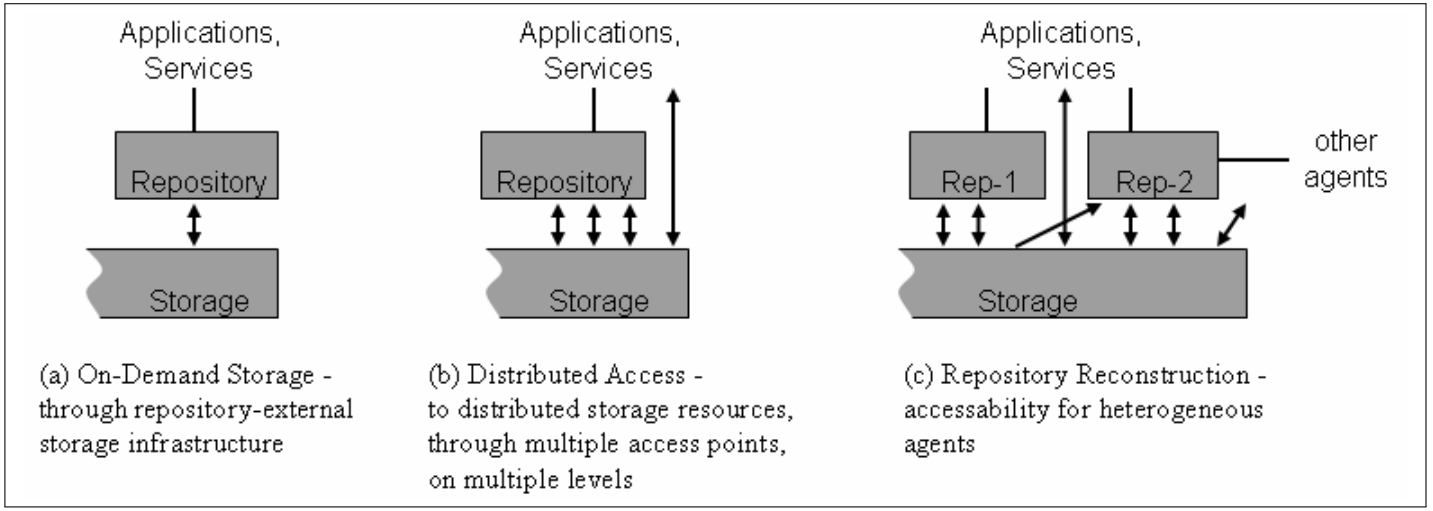


Figure 3.5: Schematic view of how the Open Storage Interface implements the three storage scenarios (cf. section 1.1.1).

4 Dariah: Federation for Decentralised Agents

This chapter analyses the *Federation* interface, which connects object and the application layers in the repository reference architecture (cf. section 2.2.3). It creates a more fine-grained view on interoperability in repository federations and analyses existing approaches with regard to the interoperability attributes *syntax* and *semantics* in digital objects, as well as *structure* and *patterns* in the overall information system (cf. section 4.1). This discussion finds that least explored among those attributes are patterns for interaction between independent agents in repository environments.

With the objective to analyse and extend federation patterns, we make a case for pattern-driven design of repository environments, and introduce – in addition to the *Distributed Query* and *Harvest* patterns that have been previously implemented in repository environments – event-based *Notification* as an enabler for immediate and directed federation (cf. section 4.2). The findings with regard to Notification were developed in the course of the Dariah project, which establishes a European e-Infrastructure for the humanities. [86] Section 4.3 therefore demonstrates Notification-based federation in the Dariah environment.

4.1 Attributes of Interoperability

Section 2.2.3 introduced the Federation interface and various existing federations. This

section looks deeper into how a federation is technically built, and it establishes a continuum of federation attributes. This continuum helps in the design of federation activities, and it also helps isolating gaps in federation technologies. To achieve this we illustrate how a federation works on different technical levels, and that object syntax and semantics, as well as system patterns and architecture are equally important to achieve interoperability.

Z39.50 for querying library catalogues has been around since 1988, and became a US standard in 1992. [200] Z39.50 was widely spread and still is. 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 it is not using HTTP as a transport layer but defines its own protocol, it is neither RESTful [151] nor loosely-coupled [260], 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) in 2002 [132]. 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, [198] 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 September 2009, OAIster, a self-elected ”union catalog“ for digital resources, [50] cross-referenced 1100 repositories by way of the OAI-PMH and their more than 23 million digital resources. However, OAI-PMH bears various caveats. [179] The close connection of the ‘pidgin’ metadata format Dublin Core [98] to OAI-PMH is often problematic, since its rather loose definition leaves various possible interpretations open to the user. Various initiatives (e.g. [289, 310]) 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”. [310] Embedding other metadata formats into OAI-PMH may hence be requiremed for many harvesting initiatives.

Overall, this leads to the slightly paradox situation that while the OAI-PMH itself is rather light-weight ([236, 323]) and is hence well embeddable into existing systems, 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 [149], Dare [290], Europeana [21]). 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. [291, 309, 310]).

All of the federation mechanisms mentioned – Z39.50, SRU and OAI-PMH – focus purely on searching metadata catalogues, and until recently there were in fact few initiatives addressing other object features (cf. section 2.2.1). The first federation standard going beyond this metadata focus is OAI-ORE, the Open Archives Initiative's Object Reuse and Exchange format [222]. Its focus are whole digital objects, both simple as well as complex ones consisting of multiple distributed files (e.g. text, images, data, video), metadata, and relations to other objects. OAI-ORE defines standards for the description of digital objects, e.g. RDF serializations in both XML or by embedding RDF statements into the Atom Syndication Format [251]. As such, it is not so much a federation protocol as rather a format.

When looking at the federation mechanisms up to here, we touched upon various dimensions: SRU superseded Z39.50 to replace it – amongst other – with a HTTP-based *protocol*; other than the query protocols SRU and Z39.50, OAI-PMH follows a Harvest *pattern*; OAI-ORE is an object *format* addressing data, metadata and relations. So what are the individual components a federation mechanism is composed of? – Guidance on this question is given by Thibodeau in [304], where he identifies three dimensions of an object: the physical object (encoding), the logical object (syntax), and the conceptual object (semantics). In analogy to these three dimensions of an object, we also identify three dimensions of an information system: a protocol, an interaction pattern, and the overall architecture. Just like the OSI Reference Model [327] is largely agnostic to the content it carries (network plus data blob), we argue that on a conceptually higher level it is the system architecture that carries and interacts with the object (architecture plus object).

- **encoding** (object) – defines the byte serialisation that associates an abstract character and a code point [118], and is essential basis for machine interaction. [141] (Please note, Dublin Core also defines "syntax encoding" and "vocabulary encoding" [322], 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. [288] For example, an XML-document – a prevalent syntax for describing digital objects – that complies with the lexical rules and XML markup grammer is called "well-formed". [14]

- **semantics** (object) – define the meaning of terms and statements in a certain context [203], 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. [218, 264] Containers such as METS [238] 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 present a well-proven generic approach for its solution, consisting of the constituent components, their responsibilities and relationships. [69, 106] Patterns can be building blocks of system architecture, or define the way in which distinct information systems exchange information (e.g. triggers, workflow, 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; whereas it may be based on a reference architecture that is relevant in a domain or recurrent application context. [314]

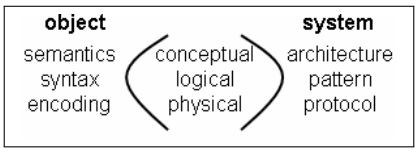


Figure 4.1: Attributes of Interoperability on the three layers of abstraction identified by [304], pertaining to both, the object and the system. An analysis of Federation particularly looks at the logical layer.

Of particular interest to us in this context are the dimensions *syntax (object)* and *pattern (system)* in the logical layer in the middle. 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. For example, for metadata there is a myriad of available standards [128] and also of techniques such as the Warwick Framework metadata component structure [218] or mixing and matching various standards in "application profiles" [184]. We therefore rather point to existing guidance on the physical and the conceptual layers.

However, it is the logical layer on which generic research on federation mechanisms can be conducted. In fact, various relevant standards have emerged from repository research with regard to the *syntax (object)* dimension. This includes foremost the construction plans and building blocks for digital objects: Various standards support the serialisation of digital objects, including XML-based container formats like METS [238] and MPEG21-DIDL [4], or the RDF-based aggregation format OAI-ORE [222]. One of the building blocks of these object serialisations are persistent identifiers, such as the URI or the DOI standards [185]. Apart from the digital objects themselves, there are also standards for e.g. query syntax for metadata catalogues (CQL, Contextual Query Language [10]), or for mark-up of search results from such queries (OpenSearch [224]).

While the logical layer of the object is fairly well researched, there is fewer work on *federation patterns* of the information system. Two different patterns can be clearly discerned from the protocols described above: Z39.50 and SRU call one or many servers with a specific query, and OAI-PMH harvests metadata from a number of servers. We are not aware of significant other patterns. Inspired by other works on patterns (e.g. [106] describes more than 100 patterns for distributed software environments) we therefore look closer at patterns used in repository environments in the following section, and in particular we argue that a *Notification* pattern deserves a key role in repository environments.

4.2 Federation Patterns

The previous section established a continuum of federation techniques and diagnosed a gap with regard to federation patterns. After the brief definition of *patterns* as generic approaches to recurring design problems, this section looks closer at federation patterns and fills this gap. In particular, this section analyses two federation approaches that are prevalent in current federation systems – *Distributed Query* and *Harvest* – and supplements them with event-driven *Notification*, which has not been considered as a federation approach up to now. Moreover, we argue that *Notification* is the missing link between *Query* and *Harvest*.

Christopher Alexander, a mathematician and architect, was the first to define the notion of "patterns" in the context of architecture: "Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solutions millions time over, without ever doing it the same way twice." [70] Patterns were later picked up and transferred to software architecture by numerous people. [169, 209]

However, already in these early works, there are slightly differing approaches to how granular and abstract patterns should be, and how they can be embedded into a whole language of related patterns. As Fielding [151] points out, Alexander's patterns have more in common with architectural styles than with programming-oriented constructs and are hence on a higher level of abstraction and granularity. Also with regards to

how patterns interact, existing approaches to patterns differ. In 1997 [209] find that many popular pattern texts at that time were collections of isolated patterns, rather than embedded into a comprehensive pattern language of hierarchically linked patterns, which constitutes a whole architectural style.

Patterns were mentioned as constituting for "architectural styles". Fielding [151] defines an architectural style as "a coordinated set of architectural constraints". Architectural styles may overlap and support each other. For example, [275] considers Service Oriented Architecture to be derived from four architectural styles: Client/Server, Layered System, Pipe and Filters, and composition and orchestration of Distributed Agents. [106, 138] Hohpe [190] adds Declarative Programming and Event-based Programming to that. We will be looking at aspects of some of these architectural styles in the following.

In this section we are looking at patterns for federating distributed digital repositories. We describe, compare and link three high-level patterns for repository federation – *Query*, *Notification*, and *Harvest* (cf. figure 4.2). Each pattern is largely along the lines of a different architectural style – *Query* by a client-server style, *Notification* by event-driven programming, and *Harvest* by the REST style –, indicating a broad scope of application contexts. At the same time however, they overlap in parts and hence show no obvious gaps between them, thereby indicating a good coverage of conceivable application scenarios.

Each of the patterns can be extended through *filters* or *transformations*. Filters allow to better define the set of objects to be selected for federation. For example, a filter on descriptive metadata of digital objects is the Contextual Query Language (CQL) [10]. On the other hand, a transformation can be applied on messages as they are passed from the source to the client, an approach that is inspired by the Pipes-and-Filters style. [107] All the federation patterns potentially benefit substantially from adequate filters and transformation mechanisms, both in terms of efficiency, robustness and their manageability.

The patterns described below are inspired by comprehensive pattern languages of numerous, hierarchically linked patterns. [106, 169, 191] However, we deviate from their

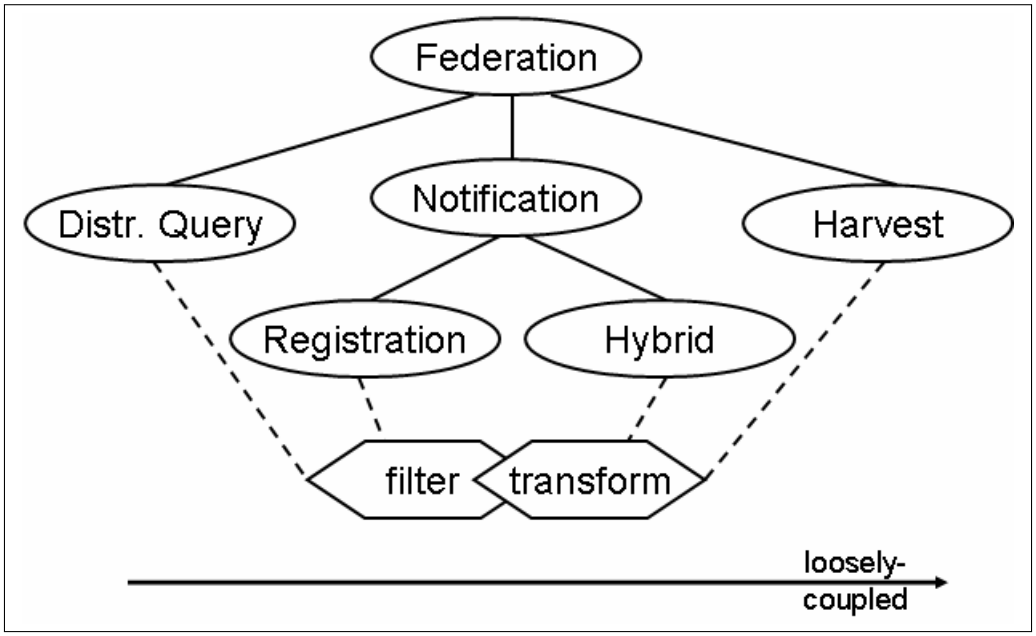


Figure 4.2: A pattern language for federation patterns. Mechanisms to filter and transform objects can be embedded into each pattern to raise the scalability and manageability of the federation.

structure to some extent, as the language of Federation patterns spelt out here is smaller and the scope of the patterns is more focused. Other than the template in [106], the patterns are more succinct and solution oriented, also giving an overview of current implementations where they exist. Structural elements are spelled out for each pattern, to support readability.

Before addressing each of the patterns individually, the following paragraphs summarize context and objective of Federation patterns as a whole:

Problem area: **Repository federation** encompasses viewing, re-using or processing both, individual objects as well as entire sets of objects, between independent software agents. The agents involved can be digital repositories or any other software agent in a repository environment (e.g. registries, added-value services).

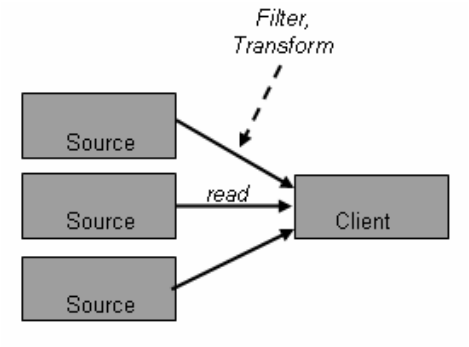
Some of the *challenges* that are addressed by federation patterns to a different degree include

- efficiency – Efficiency in a federated environment is particularly dependent on the multiple, independent agents. Each additional agent raises the risk that the low performance of that one agent impacts detrimentally on the overall performance of the whole federation.
- consistency, completeness – 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.
- scalability – The overall performance of a federation should not degrade with an increasing number of agents.
- openness – This thesis argues that openness is one of the key properties of federations. In particular, it characterises 'openness' to be constituted of the three attributes loosely-coupled, simple, and decentralised (cf. section 2.2.2).

- standard – Enabling openness and decentralisation indirectly calls for a minimum level of standardisation or also the flexibility to embed standards with regard to syntax, semantics, or structure – the other elements of the interoperability levels –, since standards support the implementation of federation mechanisms into decentralised agents that build on heterogeneous platforms and are governed independently.

4.2.1 Distributed Query

Short Description: 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. Result sets can be filtered through adaptation of the query; responses can be transformed on delivery either through re-representation services or workflows.



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 are often difficult to integrate along both, efficiency and content at the same time. 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" [109].

♦♦♦

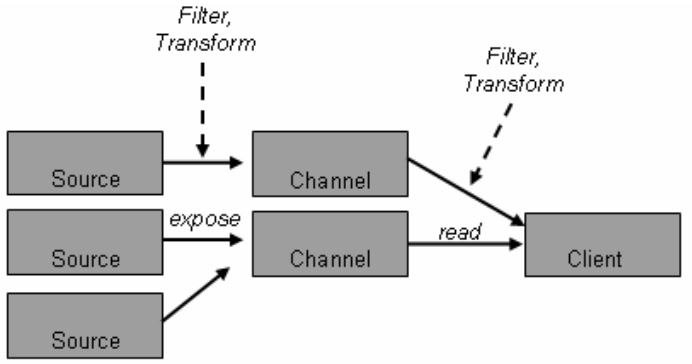
Exemplary Implementations: There are various implementations of the Distributed Query pattern in the repository community. Z39.50 for querying library catalogues has been around since 1988. Z39.50 was widely spread and still is, along with its successor SRU/W that is based on web services respectively REST. While Z39.50 and SRU/W merely exchange object metadata, other messages are conceivable including added-value services [49, 305].

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. [125] SDMX has been chosen, 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.

4.2.2 Notification

Short Description: 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. A common focus on CrUD events could facilitate a standard interface across heterogeneous agents, yet specialised notifications are conceivable.

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. Filters can be applied during the exposure into the channel respectively on read.



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. [189]

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. [107] The implementation of e.g. an Observer pattern on CrUD events allows the client to follow state changes in the repository as they occur. [106]

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. [190, 319] 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. In the worst case a delay of a whole poll cycle is needed until a notification is retrieved. However, this impact is generally not seen as critical as pointed out e.g. by the cloud infrastructure provider Bycast [280]. Bycast's Hybrid push/poll notification system is at the core of its cloud infrastructure, and as a mechanism for its broker, it employs the Atom syndication protocol.

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 [147]. 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. [54]

The probably most comprehensive implementation of messaging is in place in the iRODS rules system that is triggered through administrative actions. [267] The iRODS rule system provides a customizable framework for executing tasks – so-called

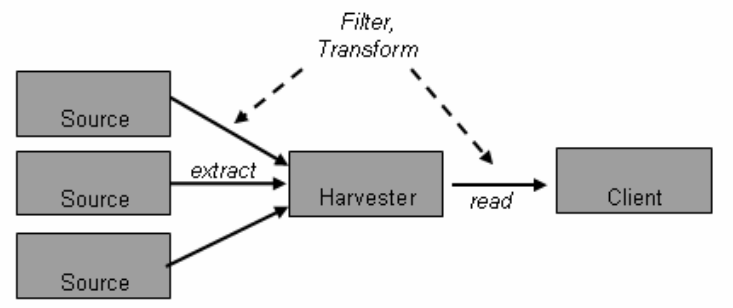
"microservices" – on occurrence of definable events. In a way, rules register microservices with specific events, and because of this basic similarity we can classify rules as event-based notifications. However, rules go beyond notifications since they are capable of defining microservice workflows. [101] All these messaging frameworks existing in repository installations, however, are system-internal. We are not aware of an open approach that is employed as a federation mechanism across heterogeneous agents in a repository environment. A first step towards such a Notification-based Federation could be a Hybrid Notification based on the Atom protocol. Since Atom is an XML-based standard, it enables communication across heterogeneous agents with different software bases. As mentioned above, 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. [231, 280] 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 4.3 we suggest a prototype for that, which was developed in the framework of the Dariah e-Humanities infrastructure.

4.2.3 Harvest

Short Description: 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 amongst other 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).

Filtering of the objects to be exchanged occurs in the communication between the source and the harvester, if the source provides relevant stubs. A transformation of objects can theoretically be conducted during the harvest, though we are not aware of any respective implementation in practice.



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, as is outlined in the next paragraph, 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, and the size and complexity of the digital objects involved. [95]

♦♦♦

Pattern Details: Harvesters such as those for web search engines are well researched, and there are relevant experiences from this community. [180] 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. [260]
- 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. [9]

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. In September 2009, OAIster, a "union catalog" for digital resources [50], cross-referenced more than 1100 repositories by way of the OAI-PMH protocol and their more than 23 million digital resources – 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. [113, 235, 276]) and

it has also been tweaked to harvest whole objects marked up in METS [291] or MPEG-DIDL [310]. 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" [19] 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.

Taking this one step further unveils a connection between the Harvest and the Hybrid Notification pattern (cf. section 4.2.2). The Sitemaps exposing the repository contents (exemplary described above as an alternative mechanism for the Harvest pattern) are very similar to exposing repository contents via an Atom feed (an exemplary implementation suggested for the Hybrid Notification mechanism). Similarly, recurrent iterative harvesting cycles are comparable to polling the message queue. The only difference between the two pattern is that for the Harvest pattern a complete list of objects (and object metadata including the last update) is exposed, whereas Atom-feeds provide a history of repository events and hence can only infer the complete list by reconstructing the current state.

In conclusion, the Sitemaps-based harvesting shows that the Harvest pattern is a universal pattern that is not tied to OAI-PMH or any specific technology. Furthermore, the touching point between iterative harvesting and hybrid notification can be interpreted as an indicator of the completeness of the pattern language at this point.

4.3 An Atom-based Repository Federation

After the previous sections introduced the context and concept of Federation patterns,

this section presents an actual federation environment with multiple repositories and other independent agents. This environment will establish Dariah, a closely-knit, yet open repository infrastructure for the humanities. [80, 86] The close interaction between the heterogeneous agents calls for a Notification-based federation approach. Therefore, this section illustrates the application of Atom-based Hybrid Notification (cf. section 4.2.2) in the context of Dariah.

Dariah is a project in the framework of the European Strategy Forum on Research Infrastructures (ESFRI) [256] and is currently in its initial phase. ESFRI projects are designed to offer research communities essential infrastructure for decades to come, e.g. a large telescope for astronomy and an icebreaker ship for the polar sciences. For the humanities, Dariah builds a digital infrastructure to share cultural artefacts, re-use existing tools, and collaborate across institutional, cultural, and disciplinary boundaries. Partners in Dariah include researchers and humanities centres, including DANS (Data Archiving and Networked Services) in the Netherlands, the Centre for e-Research (CeRCH) at King's College London, as well as the State and University Library Goettingen, Germany.

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 (hence *open*), while the contents in Dariah and functionalities provided through Dariah should be accessible to the researcher as if Dariah was a single platform (hence *closely-knit*). Foremost, as a virtual research environment that supports active research, resources in Dariah may change over time and in an early stage of creation they may indeed be private. These prerequisites – decentralised, heterogeneous agents that need to stay in sync with the state of other agents; with digital objects that may change frequently – call for a Notification-based approach.

The Dariah test environment for linking heterogeneous repositories spans three different systems: TextGrid, iRODS, and Fedora. In order to synchronise the states of the three repositories, notifications are sent to the respective other repositories on the creation or modification of a digital object. In the production environment this mechanism will be used to replicate data across multiple sites, and to update external applications such as

search and analysis about changes made in any of the Dariah sites.

Both, iRODS and Fedora offer internal event mechanisms – iRODS through its rules/microservices [267], and Fedora implements the Java Message Service JMS [147]. The exposure of repository events via Atom can be directly integrated into these event mechanisms. While TextGrid does not offer an internal event mechanism, the TG-crud interface handles all updates to objects in TextGrid and it can easily be adapted to expose object creation or modification. The Atom feeds from the three repositories are all handled via a single Apache Abdera [163] server. While the production environment will likely consist of multiple Atom servers, a central server is sufficient for the test environment.

The repositories poll the feeds of the respective other servers and thereby synchronise with their states. As an additional feature, a repository may offer multiple feeds via the Atom server – e.g. one feed exposes all the objects, whereas others may only expose specific format types such as only XML objects. This type of server-side filtering is more efficient for both client and server – for the client since it does not need to filter itself based on the metadata of the object, for the server since this will reduce overall polling.

The reason for why this will reduce overall polling at the server is related to the fact that Atom feeds are HTTP-based services, embedded in the web architecture, and hence also supported by the infrastructure of proxies and caching servers. Furthermore, polling an Atom feed that is unchanged only puts minimal load on the Atom server. Specifically, conditional HTTP GET's (i.e. the HTTP header 'If-Modified-Since') ensure on a HTTP level that the feed is only downloaded if it actually changed.

In conclusion, we have presented the Dariah research infrastructure for the humanities, the diversity of its collections and the vision of an open environment of decentralised agents. To ensure coherence among these 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 ideas. An experimental setup that links TextGrid, an iRODS and a Fedora test server have demonstrated the viability of this approach.

4.4 Conclusions

Decentralised information environments are emerging, in which a repository is but one agent among a multitude of others. To name just some of the conceivable scenarios of such environments, repositories may replicate relevant objects of another source (e.g. institutional vs. thematic repositories), parts of a single digital object may be spread over various repositories (e.g. e-Publications [38]), repositories may depend upon external re-representation and preservation services [186]. This chapter analysed repository federation, which is capable of interweaving multiple heterogeneous repositories as well as other dispersed agents into a single virtual repository.

The Dariah e-Humanities infrastructure is one such environment, in which multiple repositories aim to federate their content while remaining independent organisations. Other than existing federations, the content in Dariah may be in diverse formats, change frequently, and be private.

This chapter identified attributes on three levels – physical, logical, and conceptual – in both digital objects and information systems, that contribute to interoperability between diverse agents in an open repository environment. Thinking in these attributes of interoperability fosters a new perspective on the challenges involved in federation. This new perspective offers a structured way to develop new federation models with existing mechanisms, and also to identify gaps in existing mechanisms.

The most pressing gap in context of the Dariah e-Humanities infrastructure pertained to interaction patterns between agents. This chapter resolved this by introducing the Notification pattern, specifically Hybrid Push/Pull Notification based on the Atom syndication format. Notification is particularly suited for decentralised environments, in which changes to e.g. digital objects may occur frequently and the agents need to be closely synchronised.

Rather than convergence to a small set of concepts and technologies, we are expecting diversity and decentralisation to increase in repository federations. New application contexts of repositories (e.g. data-driven research, enterprise systems) and subsequently changing requirements to repository infrastructure, as well as the ongoing integration of

new technologies (e.g. Linked Data [179], clouds as in DuraCloud [117]) in the field seem to point that way. In the face of this growth and diversity, the approach presented in this chapter may contribute to a more structured discussion and avoid disintegration and redundancies within the repository community.

But how do the results of this chapter relate to the federation **scenarios** put forth in section 1.1.5 – specifically Scientific Analysis and Task Management? (cf. figure 4.3)

Search and Analysis is a recurrent requirement in the Dariah environment. However, a simple Google-type search is insufficient for a scientific environment. Specialised analysis services may process various types of data and their metadata, including images and sound. In other words, rather than providing a generic search portal, Dariah aims to facilitate the creation of external search and analysis services, such that any community or project can develop their own portal. Thereby, one-time analysis efforts that research a specific question on a specific set of digital objects are offered possibilities to harvest the objects into a dedicated analysis environment. Ongoing services that grow with the availability of new material are provided with notifications about object creation, update, or deletion. Dariah therefore aims to support various protocols and patterns, to allow for different approaches to system interoperability.

Just like the Dariah infrastructure aims to support various approaches to system interoperability, it equally aims to expose its content in different formats to facilitate different approaches to object interoperability. To underline this, the Task Management scenario is similar to the search and analysis portals described above, yet it operates purely on the object metadata and whether objects are available in the first place. Thereby, the exposure of object metadata through a Query or a Notification pattern, as well as adequate filters enable the implementation of a Task Management application.

Dariah aims to foster interoperability of these mechanisms across the diverse repositories and other agents in the Dariah infrastructure. This allows that external agents can embed their own application environments into the Dariah infrastructure, just like it enables the Scientific Analysis and the Task management scenarios.

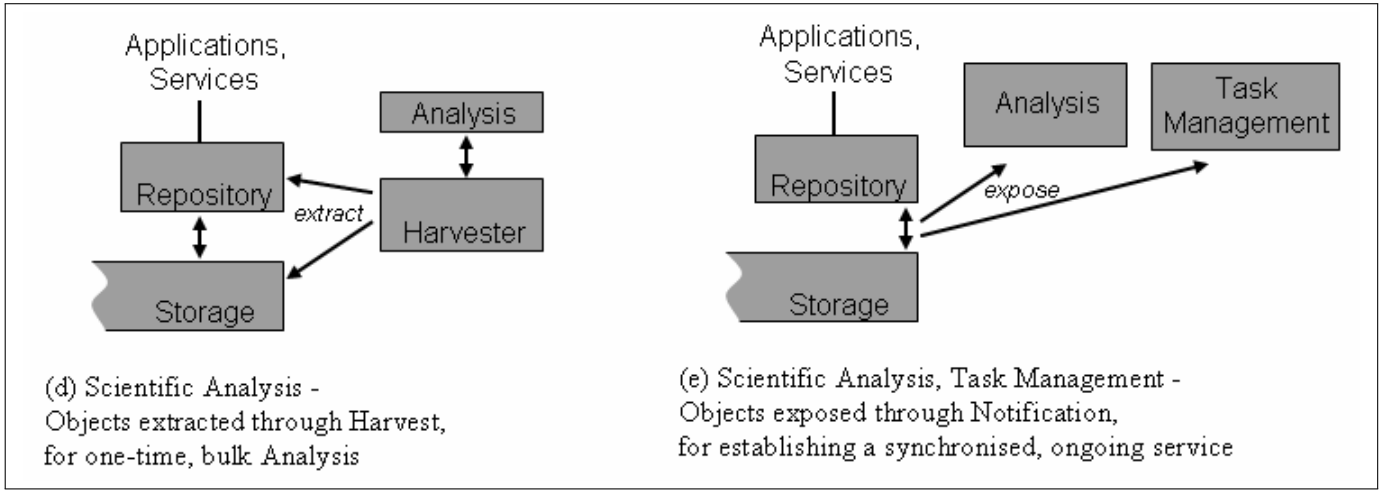


Figure 4.3: Schematic view of options for resolving the scenarios 2.a, Scientific Analysis, and 2.b, Task Management, through repository federation (cf. section 1.1.5).

5 TextGrid: A Repository Infrastructure for the e-Humanities

This chapter introduces TextGrid, an e-Infrastructure for the humanities and part of the German national grid infrastructure D-Grid [249]. While built on the basis of grid technologies rather than repository software, it is implemented based on the concepts introduced in the previous chapters. By introducing TextGrid, this chapter demonstrates the community-driven creation and release into production of an open repository environment: the translation of the task-oriented requirements of the TextGrid target community into a technical architecture that implements the repository reference architecture on a grid-based infrastructure.

The humanities have, until recently, not been seen as technology-shapers. Until recently. Recent years have seen a host of activities in the humanities. Large-scale programs to digitize historic documents or relics, the construction of portals and online virtual libraries, and systems for preserving digital objects perpetually are just some of the fields where the humanities have been particularly active in the last decade or two. [72] Arguably, the repository community builds extensively on the traditions and experiences of the humanities, in particular their ability to deal even with scattered or garbled information and centuries of experience in organizing knowledge. [72] Metadata, digital curation and preservation, scholarly communication and open access – all these issues resonate deeply with the humanities, and they are at the core of digital repositories.

TextGrid is one of the early humanities projects working with grid technologies, the first at least in Germany. [81, 93] Its initial target group is German language and literature, for which it establishes a virtual research environment for the collaborative editing, annotation, analysis and publication of specialist texts. In its endeavour for sustainability, other disciplines from the humanities including musicology and linguistics have been included into the TextGrid community. TextGrid stands united with various other international initiatives such as DARIAH [86] and Bamboo [29] in its key objectives including federation and preservation of humanities data; novel technologies to operate on huge, federated collections; and effectively new methodologies to deal with new research questions. [121]

To date, processing specialist texts in German language and literature is the most advanced use case in TextGrid. Core data in this context are images of digitised manuscripts or other specialist texts, as well as XML-annotated versions of the texts following the schema of the Text Encoding Initiative (TEI). TEI is a rich format, which scholars can use for most parts of their research – from initial transcription, to semantic annotations, to eventual publication. TextGrid offers scholars a place preserve this data (the TextGridRep / TextGrid repository), as well as a research environment to collaboratively work on this data (the TextGridLab / TextGrid laboratory).

While building a virtual research environment for the humanities, TextGrid is part of a larger programme assigned by the German Federal Ministry of Education and Research BMBF to build the German future e-Infrastructure "D-Grid" with communities such as high energy physics, astronomy, climate, and medicine. TextGrid represents the humanities in D-Grid and benefits substantially from the interaction with other communities and their experiences in building e-Infrastructure. At the same time it contributes its perspectives and expertise. In this perspective the virtualization of hardware resources – the original trigger of grid technologies – is but a useful boon. The key resources in the humanities are data, knowledge about data, and services. So rather than virtualising physical hardware, connecting these intellectual resources is the central challenge and opportunity when building e-Infrastructure in the humanities.

5.1 Levels of Participation

Before delving deeper into the technical architecture of TextGrid and the lessons that can be extracted from this, this section takes a deeper look at the specific situation in the humanities and identifies requirements for the TextGrid technical architecture. It focuses on attaining interoperability with regards to object semantics, which is a core objective of TextGrid. Also, semantic interoperability is clearly more than a purely technical issue, and it therefore illustrates how a technical design must interplay and – if possible – take advantage of the social and organisational context.

The humanities are large and diverse. Resources in the humanities and means of scholarly research are mostly low-cost (compared to, e.g., a hadron collider for nuclear research or a satellite network), and well accessible as they are not confined to a group of distinguished experts. This allows scholars, students, interested laymen and just anybody to participate in scholarly research as equal partners and contribute their perspectives. Further adding to the multi-faceted nature of the humanities are regional traditions within certain fields, with each school of thought employing different notions and research methodologies. Eventually, countries differ in their interpretations as to what constitutes "the humanities" in the first place, whereby e.g. sociology or the arts are considered part of the humanities or distinct fields altogether. [164]

This size and diversity in the humanities means that apart from a few sub-fields such as the psycholinguistics, there are few authoritative standards that a whole (sub-)community complies with. Specific research questions and specialised methodologies to address those questions often entail wholly incompatible data encodings. For example, two research projects about Shakespeare's textitMacbeth with varying research questions may use incompatible tools for processing the data, encode different annotations and derived data, and may even work on different texts in the first place (e.g. the original version from 1606, versus a later, revised version in different languages; or even the musical version by Richard Strauss from 1888). For these reasons, humanities scholars hold their freedom high to express and encode without constraints. Acknowledged standards like TEI are themselves masterpieces in flexibility – two TEI manifestations of the same text by two different research projects can be,

but are not necessarily compatible to a given degree.

Recognising these idiosyncrasies of the research context in the humanities, TextGrid put the following principles at the basis of its technical and organisational design. In addition to the requirements posed by the community context, these key principles are guided by the goal of creating a collaborative infrastructure that can grow and evolve over time.

- an **generic infrastructure** – There is no *a priori* expectation for data, tools, or methodologies. In particular, functionalities are only constrained where indispensable; the application context of a specific module or service is not pre-defined. Thereby, functionalities can be re-used in different contexts and re-mixed for efficient development of novel applications.
- fosters **specialised applications** and semantically deep processing – Specialized research questions, methodologies, and contexts require data formats, metadata, and interfaces to be freely adapted. While there is a high level of flexibility, the level of support and interoperability for specific formats, metadata schemata, or interfaces may vary.
- motivates **participation** – Community participation is crucial to obtain a growing base of scholarly texts, range of diverse tools and methodologies, and ultimately the sustainability of the infrastructure.

Particularly the first two principles – open and generic vs. specialised – may seem to be contradictory at first sight, and they reflect the contradictions in the original requirements for enabling idiosyncratic research questions and methodologies while fostering collaboration and interoperability. However, they are only contradictory at first sight, when aiming to build a comprehensive system that fulfils all the posed requirements. Yet, TextGrid is not so much a system as rather an open platform that enables scholars to adapt the environment to their needs.

TextGrid's layered approach aims to achieve interoperability in this diverse environment by engaging people. TextGrid partners are domain researchers from the humanities with their own ideas about how to encode domain semantics "the right

way". Even among TextGrid partners these ideas vary, and the definition of a shared standard for the project (let alone a standard for the whole community) would be a challenge. Therefore, rather than defining data standards and imposing rigid technical guidelines, TextGrid defined an organisational framework that allows any researcher to choose their level of interoperability or flexibility as to their discretion. At its core this organisational framework consists of layered incentives. The lowest level allows maximum flexibility and hence a low entry barrier, while the highest level enables interoperability and offers tools and support. In other words, TextGrid offers the carrot rather than the stick, by nurturing opportunities rather than enforcement.

The layered approach manifests in various aspects, including data, services and preservation stores. To illustrate the TextGrid collaboration layers on data:

- any data format can be uploaded, TextGrid ensures bit-preservation
- metadata facilitates data management and retrieval (metadata-based search)
- by uploadig XML-based texts, a series of services can be used on the data including streaming tools, an XML-editor, and other functionalities
- if the XML follows TEI encoding, TextGrid offers graphical editing, metadata extraction, and other functionalities
- defining a mapping to the TextGrid recommendation for a TEI core encoding allows interoperability on a semantic level

In its design of the incentive approach, TextGrid follows the experiences of collaborative environments. [210] The user can do whatever she wants, but by being interoperable and compliant to TextGrid recommendations she increases exposure and is provided with more functionality in the TextGrid virtual research environment. Moreover, the platform is designed to be open, and is expected to grow over time. There may even be competing incentive systems within TextGrid at some point in the future. It is the conviction of TextGrid partners that rigid guidelines hamper participation more than they enable interoperability. Eventually, an interoperable environment without any data fails to be useful. TextGrid thus puts enabling participation over interoperability, while fostering interoperability wherever it can.

5.2 Grid-Repository Architecture

Building on the requirements identified in the previous section, this section describes the TextGrid reference architecture. Even though TextGrid is built on grid technologies rather than existing repository software, this section finds TextGrid to be a repository-like system by mapping the reference architecture onto the repository reference architecture developed in section 2.2. This is the first step of a mapping to be continued in the following section, which looks into whether TextGrid implements the open storage and federation interfaces as well, and hence can be identified as not only a repository architecture but an open repository environment.

Technologically, the requirements and guiding principles formulated in the previous section translate into a service-oriented architecture [115] in four conceptual layers (cf. figure 5.1). In tandem, the layers enable openness, while ensuring robustness in the face of evolutionary development of the technology environment and the organizational context. For the user, they hide the complexity of the multiplicity of software components involved and offer entry points at various levels. [299]

- TextGrid **application environments** – The main application environment in the initial project phase is Eclipse-based and geared towards use in German language and literature. However, other user groups or workflows may call for other application environments, which can likewise be installed atop the other, more generic TextGrid layers.

- functional building blocks wrapped into **services** – Atomic functionalities such as tokenization, lemmatizing, or collation are implemented as individual REST- or SOAP-based services ([151, 175, 274]) to be re-used by other services or plugged into one or many application environments. The service environment is open and growing, only tied together with optional interoperability mechanisms that enable authentication and access to shared services across TextGrid. [300] Projects like DARIAH, Bamboo, and Interedition are contributing or will be contributing in the future to a shared humanities service network.

- the robust core of the TextGrid infrastructure consists of **archives** and

middleware for virtualisation – The TextGrid utilities comply with the interoperability framework for services, however they offer more generic functionality at increased stability and scalability.

In this architecture the layers are not merely a way of guiding our conceptual thinking. Each layer employs a separate set of technologies and is still coupled to all the other layers. In its initial setup, the bottom two layers employ grid services (WSRF), the service layer supports both SOAP- and REST-based services, and the application environment is an Eclipse-based rich client. In addition to the separation of the layer-internals, each layer offers varying degrees of flexibility and complexity, and services from each layer can be composed as to the needs of the research activity in question: sharing data in the archives layer, co-developing functionalities in the services layer, and collaborating on a methodological level in the application environment. The combination of all three layers establishes an open platform to address specialised research activities by reusing data and combining generic functionalities, and it fosters participation on all levels. In this way, the TextGrid reference architecture implements the key requirements as formulated in the previous section.

Taking a closer look at the TextGrid middleware (cf. figure 5.2), it consists of various components for handling files in the data grid, rights management in an RBAC-enabled database, metadata in an XML database, and relations in an RDF triple store. [299] These distinct components are combined into three core TextGrid utilities that offer one-stop-shops for data management and preservation, content analysis, and team management functionalities:

- **TG-crud** – the component for data management. It offers CRUD functionalities – create, read, update and delete – for TextGrid objects consisting of a container of both data and metadata. TextGrid metadata is inspired by Dublin Core with mandatory fields for object management and freely extensible optional fields. [301]
- **TG-search** – a component for search and analysis. It goes beyond average search mechanisms and offers functionalities dedicated to the needs of researchers in literature and language. As part of these capabilities, TG-search offers search based on object metadata, relations, full text, as well as XQuery capabilities for

XML-based files, or any combination thereof.

- **TG-auth*** – supports the collaborative processes in TextGrid by offering an interface for managing users and their roles, project teams and rights. Both TG-crud and TG-search interact transparently with TG-auth* to ensure adequate rights management.

This approach of combining distinct grid services and databases in task-oriented interfaces (TG-crud, TG-search, TG-auth*) is a prevalent software architecture pattern called a Facade, which promises to make client code more usable, raises overall consistency, and de-couples the layers by reducing the dependency of client code on individual infrastructure components. [106] (The interaction between the various components and the utilities to ensure consistency is further described in the following section.) Each utility interface offers both REST- and SOAP-based protocols, and it fully implements the interoperability recommendations of the TextGrid service layer which sits on top of it.

With decoupled storage infrastructure as one aspect, TextGrid displays many features of a repository environment, which becomes apparent when comparing TextGrid with the repository reference architecture as identified in chapter 2.2. First of all, TextGrid's collections are built up of digital objects as defined through the following features (cf. chapter 2.2.1): It consists of one or many files, is annotated with metadata, may have relations to other objects (e.g. versions, formats, components such as volumes in a series or letters in a correspondence), as well as characteristics and administrative metadata for rights management or dedicated mechanisms for analysis. All these elements of a digital object are handled in TextGrid within containers that can be exported to OAI-ORE or METS packages.

Handling of objects as defined is the key requirement for digital repositories, and TextGrid bears more features of a repository environment as a comparison with the repository reference architecture from chapter 2.2.3 shows. The reference architecture is built of three layers dealing with files, objects and applications successively with growth of abstraction towards the upper layers. These three layers can be mapped to the TextGrid architecture as follows.

The lowest layer of the repository reference architecture is covered by the file handling capabilities of TG-crud. It expects containers of a file and metadata, and stores them as files in the D-Grid data grid. On ingest TG-crud creates a unique identifier and returns that ID along with other administrative metadata to the sender. These simple CRUD capabilities for the object containers are sufficient to realise the repository file layer. The implementation of TG-crud [302] employs Globus [42] in the D-Grid software stack [177] through a grid interoperability abstraction offered by the Grid Application Toolkit GAT [71]. However, the current implementation of TG-crud is of no particular importance for this thesis and for the argumentation here. There are discussions within TextGrid of moving away from GAT, however this switch of technologies does not impact on the basic repository architecture nor on the interfaces which link the file layer with the object and the application layers on top of it.

The TextGrid object layer is implemented through a number of services. First of all, TG-search, which builds on both TG-crud and TG-auth*, is the facilitator for search and analysis of digital objects. A metadata manager [297] allows for defining new metadata fields and relations, however the core set of TextGrid metadata, both descriptive and administrative metadata and characteristics, is set. In that TextGrid offers metadata management that is not as sophisticated as e.g. the object modelling offered by Fedora or Tupelo, yet is more flexible as e.g. EPrints and DSpace. The TextGrid metadata manager is implemented as a service in the TextGrid service layer and complies equally with the interoperability requirements as any other service. Analogously, other object management functionalities can be added to TextGrid as the project evolves and in parallel to other object management facilities, as long as they all comply with the basic TextGrid container and metadata specification.

Since the TextGrid file and object layers are designed to be open and decentralised, TextGrid conceptually supports the creation of multiple application environments that can each be tailored to specific user requirements or be built on dedicated technologies. The TextGrid application layer allows for the creation of graphical and interactive user environments, whereas the TextGrid service layer enables users to add their own services tailored to a specific research activity into the TextGrid environment. Therefore, in a sense both the TextGrid application layer as well as the user-oriented

parts of the service layer can be mapped to the application layer of the repository reference architecture.

The last few paragraphs illustrated how TextGrid implements the repository reference architecture. Lastly with regards to the federation interface, TextGrid fails to make its objects available through Atom feeds, OAI-PMH, or other standard federation mechanisms, though this may be added in the future. Despite the lack of standard federation protocols, the following section expands on how concepts and federation patterns were applied in TextGrid – the last building block for an open repository environment.

5.3 Open TextGrid Environment

The previous chapter introduced the TextGrid architecture and demonstrated that it implements the repository reference architecture (cf. chapter 2.2). This section describes TextGrid's resemblance to an open repository environment because it implements an open storage interface (cf. chapter 3) as well as federation patterns (cf. chapter 4). At the same time, this chapter illustrates how TextGrid benefits from opening up its infrastructure and by interacting with various agents, which may be external to the core system and may be added over time.

TG-crud was introduced as the interface that enables file management for TextGrid's digital objects in the storage resources of the D-Grid data grid. For its simple CRUD capabilities, it offers both REST- and SOAP-based interaction, only imposes minimal interoperability requirements for authentication, and can hence be embedded in various technical environments. Being based on grid technologies, it is inherently distributed. Moreover, it can be employed by various clients in parallel without impacting on stability or consistency. In other words, TG-crud is de-coupled, distributed and decentralised – the main features of an open storage interface as identified in chapter 3.

A closer look at interaction patterns with TG-crud better illustrates this. First and foremost, TG-crud is designed to interact with just any service in the service layer, maybe with multiple, unrelated services in parallel. At the same time, the

Eclipse-based application environment TextGridLab also interacts with TG-crud directly. In fact, TG-crud was deeply embedded into the Eclipse File System [37] and replaces the local storage handler there. This abstraction through the Eclipse File System renders the interaction with the remote TG-crud transparent. In other words, the Eclipse framework interacts directly with TG-crud and Eclipse plugins do not even recognise that (except for some minimal latency). [216] All this – access to TG-crud from the numerous running clients on user's desktops as well as from heterogeneous services – is happening in parallel and in a decentralised manner, enabled through the TG-crud open storage interface.

Moving one layer up from the storage layer and before analysing federation patterns in TextGrid, we take a look at re-representation services (cf. chapter 1.1).
Re-representation services offer an effective way of adapting the repository on an object level, through services that may be external to the repository core and distributed. While TextGrid does not provide re-representation services as such, it offers comparable mechanisms through the TextGrid service environment. Objects, even those still subject to change, have unique identifiers, can be accessed through TG-crud, and can be modified through services or even service workflows on delivery. Some of the TextGrid tools do, in fact, use the TextGrid service environment in exactly that way. For example the web publishing tool uses a streaming service to convert TEI/XML through XSLT scripts into adequate representations for web viewing. Likewise, a sample for a one-day TextGrid tutorial [230] extracts names from a text and syndicates them with the respective biographies of the persons on Wikipedia. Both examples use service workflows as delivery mechanisms and hence bear resemblance to re-representation services, but other than re-representation services these mechanisms need to be triggered from the application and cannot be attached to objects or even object classes on a lower layer.

Up to this point in the analysis of TextGrid's interaction capabilities with decentralised agents, there is a clear "yes" for the open storage interface, and a partial "yes" for re-representation services. The following analysis of federation patterns will not yield a single answer, but it aims to trace various patterns in the environment as a whole.

A pivotal pattern for the TextGrid infrastructure is the notification mechanism that

ensures consistency between the TextGrid utilities TG-crud, TG-search, and TG-auth*. As part of this pattern, TG-auth* conducts the rights management in both TG-crud and TG-search. This means that whenever a new object is included into the TextGrid collections, it is not only stored through TG-crud but there is also a respective entry into TG-auth*. To ensure adequate performance while maintaining consistency, TG-crud and TG-search read and filter authorization information directly from the RBAC databases, yet changes to the authorization settings are only done via the TG-auth* interface. Independent from the synchronisation between TG-crud and TG-auth*, metadata and relations are redundantly stored in file storage (TG-crud) as well as in the XML- and RDF-databases for search and analysis (TG-search). This redundancy ensures the capacity and performance of the TextGrid search mechanism.

In both these cases, both TG-auth* and TG-search are notified by TG-crud whenever a create, update, or delete message is received (note: not on a read). The component taking care of this is called the "Adaptor Manager", and it is capable of handling any number of notifications for any service. In fact, as illustrated in figure 5.3 there is one more service currently subscribed for notification, which forwards incoming XML documents to another XML database that enables XQuery-based analysis [105]. In the prevalent XML/TEI-annotated texts in TextGrid, this allows complex queries like "return all plays by Shakespeare in which a speaker says the words 'sound' and 'fury' (or a morphological variant) in a single speech". The notification also triggers a normalisation of the text to ensure interoperability of heterogeneous texts coming from a variety of projects, and it is hence a specialised service rather than a generic utility.

This mechanism for XQuery-based analysis of XML/TEI texts demonstrates two things: The Adaptor Manager is capable of filtering notifications (in this case: for XML/TEI formats) and of piping them through re-representation services on delivery (for the normalisation); and it is suitable for handling both permanent subscriptions that are essential for consistency within the TextGrid infrastructure (the notification to TG-auth), as well as those for specialised components (the XQuery-based analysis) that are peripheral to the overall system, may exist in multiple tailored versions on the services or the application layers, and may be live only temporarily.

Taking this one step further, TextGrid considered externalising the XQuery-based

analysis into a component that provides its functionality to various repositories. The Oxford University Research Archive (ORA) [258] implemented almost the same pattern for the Oxford Text Archive (OTA)[244]. They employ the XML-database eXist [240] to enable XQuery-based access to the TEI-holdings of the OTA. Like in TextGrid, all TEI-data are forwarded to the database on ingest. Other than TextGrid, which is built upon grid services, ORA is built on a Fedora repository. Despite unlike technical infrastructure, notification from both sources to a joint XQuery-based analysis component is conceivable with low effort, and it would give rise to a unified access portal to English and German literature research.

Up to now, TextGrid does not employ a federation mechanism in addition to client/server interactions in the services layer and the notification mechanism attached to TG-crud. With regard to harvesting, there were considerations of offering TextGrid content via METS or OAI-PMH, respectively of using OAI-PMH for creating a bibliography tool, however at the time of writing (August 2009) these considerations have not materialised yet. One scenario, where harvesting patterns could be useful, is for scientific analysis of specific collections. For example, Burrow's Delta [284] offers an algorithm for authorship attribution. The algorithm first parses and indexes a collection of documents with known authors, and it then bases authorship attribution based on document similarity of a document with unknown author against this indexed collection. The separate phase for building the index, which is then frozen, suggests harvesting the collection of documents as selected by the user to build the index. The whole service could then run on an external server. TextGrid experimented with such a configuration for analysis services in an implementation of Burrow's Delta based on the hadoop framework [26, 27]. The harvesting mechanism has also been implemented in context of an experimental co-occurrence text analysis module for TextGrid. [181, 311]

Overall, this section has shown the high level of interaction between TextGrid repository services and external agents, both through the TextGrid open storage interface as well as federation patterns, and it has thereby identified TextGrid as an open repository environment. In addition to client-server interactions between internal and external agents in the TextGrid environment, notifications are a recurring pattern to ensure consistency in a loosely-coupled, scalable manner – with or without filtering,

with internal and external agents, and as-is or normalised on delivery. Apart from client-server and notification, harvesting has been tested but not permanently deployed. However, being an open repository environment, harvesting based services may be added in the future just like all TextGrid is designed to grow and diversify.

5.4 Conclusions

This chapter illustrated the context as well as the internals of an open repository environment. TextGrid is a national e-Infrastructure for the humanities in Germany. While it is not built on conventional repository software such as Fedora or DSpace, it exhibits all the features of a repository-like open environment. In this conclusion we reflect on the effects with regard to organisational context and stability that TextGrid displays because of these open repository features, and we also pick up the requirements formulated in section 5.1.

Section 5.2 has mapped TextGrid onto the repository reference architecture composed of file, object and application layers, and section 5.3 further illustrated how TextGrid implements both an open storage interface as well as various federation patterns to interact with external agents. Looking back at the various examples cited in these sections shows the robust flexibility an open repository environment offers. At its core TextGrid consists merely of the open storage interface TG-crud as well as TG-auth* for team and rights management. Together they compose the minimum building blocks for a *generic infrastructure*. All other services and applications are built around them to offer their *specialised functionalities*, and they can grow and evolve over time in any conceivable direction. TG-search with its functionalities tailored to TextGrid's initial target group essentially builds on notification by TG-crud. Another TG-search tailored at the needs of another target group or any number of alternative TG-search's can be deployed the very same way, without impacting on the first TG-search. Likewise all functionalities in the TextGrid services and application layers are enabled through open storage and federation mechanisms, and external initiatives are encouraged to *participate* and add their components into the environment.

The three highlighted items in the last paragraph – generic infrastructure, specialised

functionalities, and participation – are the key principles that should build requirements as identified in section 5.1. This shows how well the open environment fits the TextGrid requirements. Section 5.1 also made aware of the fact that the first two items – generic infrastructure and specialised functionalities – appear to work against each other at first sight. Yet, in an open repository environment they actually support each other. Moreover, an open repository environment calls for lateral thinking: information not only flows up and down the layers in a predictable way, as in many closed, tightly-coupled systems, but it may be directed anywhere – within layers, bouncing back and forth between layers, and to components the original system architects did not conceive of and may be unaware of. Indeed, external initiatives are called to add their own components and new utilisation of existing components.

The largest opportunity, however, is non-technical. It is the robust flexibility that allows an adequate translation of organisational requirements and that is capable of evolving as requirements change over time. The DuraSpace project calls this a 'chinese menu' of added-value services [117], assuming that the hungry user will find a plate of her choosing in a large menu. Or add her own.

The design of TextGrid is grounded in the organisational and social context of research in the humanities as described in section 5.1. Above we successfully mapped the principles – generic infrastructure, specialised functionalities, and participation – onto the current TextGrid architecture or open repository environments in general. As another factor for encouraging participation and lowering entry barriers, section 5.1 suggested layered approaches also for organisational aspects. Layered conventions are e.g. conceivable for both data and service interoperability, reaching from low interoperability with a low entry barrier to high interoperability and hence a high value for re-use and collaboration. TextGrid supports the creation of such layered community conventions with the same mechanisms with which it supports the fusion of generic infrastructure and specialised functionalities in a single environment.

While this robust flexibility of TextGrid facilitates all these principles, they are essentially social and organisational notions rather than technical ones. In other words, while the technology is ready to support what is needed, this openness requires a higher level of organisation within the user community. Other open repository environments

may therefore opt to constrict the openness to suit the community.

But how do the results of this chapter relate to the federation **scenarios** put forth in section 1.1.5 – specifically Scientific Analysis and Out-Sourcing Preservation Actions? (cf. figure 5.4)

One of the central TextGrid tools is its Scientific Analysis of TEI documents, which allows XQuery-based access to all of TextGrid's XML/TEI holdings. TG-search implements a Notification pattern that is triggered by TG-crud and filters all XML/TEI objects. Notable about TG-search is its separation of concerns between TG-crud and TG-search: as outlined above, it is conceivable to add other Analysis portals in parallel to TG-search, and it is equally conceivable that TG-search combines the material from TextGrid, the Oxford Text Archive and any other source in a single analysis portal.

Another notable feature of TG-search are its capabilities for enabling interoperability across heterogeneous data. XML/TEI holdings are automatically converted into a normalised form, the TextGrid baseline encoding, and ingested into TG-search alongside the original document. To achieve this automated workflow, the conversion is triggered by the very same Notification pattern mentioned in the previous paragraph, and delegated to a normalisation service. A similar setup, respectively a Harvest-based approach is conceivable for Out-Sourcing Preservation Actions, as described in scenario 2.c (section 1.1.8).

Please note though, that all services involved in this notification-triggered workflow – TG-crud, the normaliser, and TG-search – interact with TG-auth* to ensure adequate rights management. Horizontal services like authentication and rights management must be addressed for the overall system architecture.

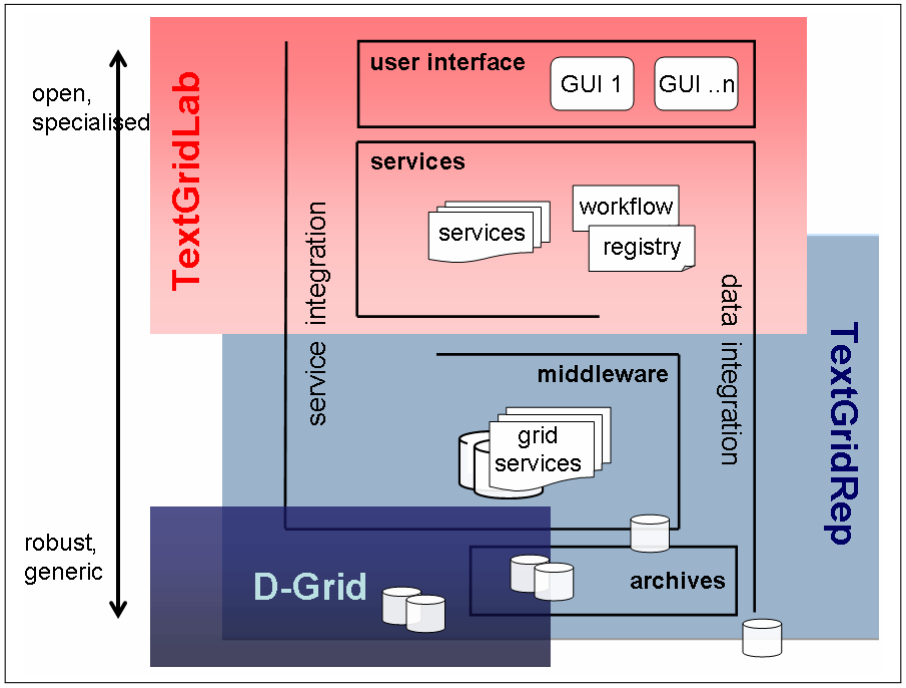


Figure 5.1: TextGrid – Reference Architecture [299]

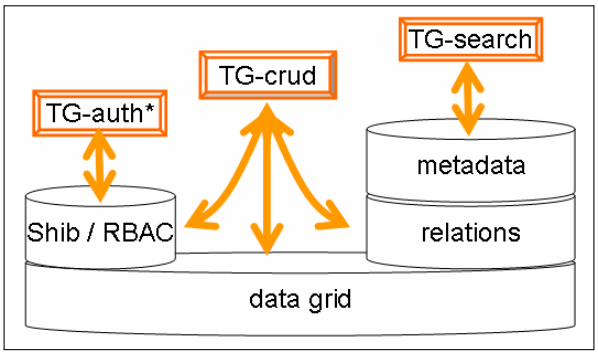


Figure 5.2: TextGrid middleware – infrastructure utilities

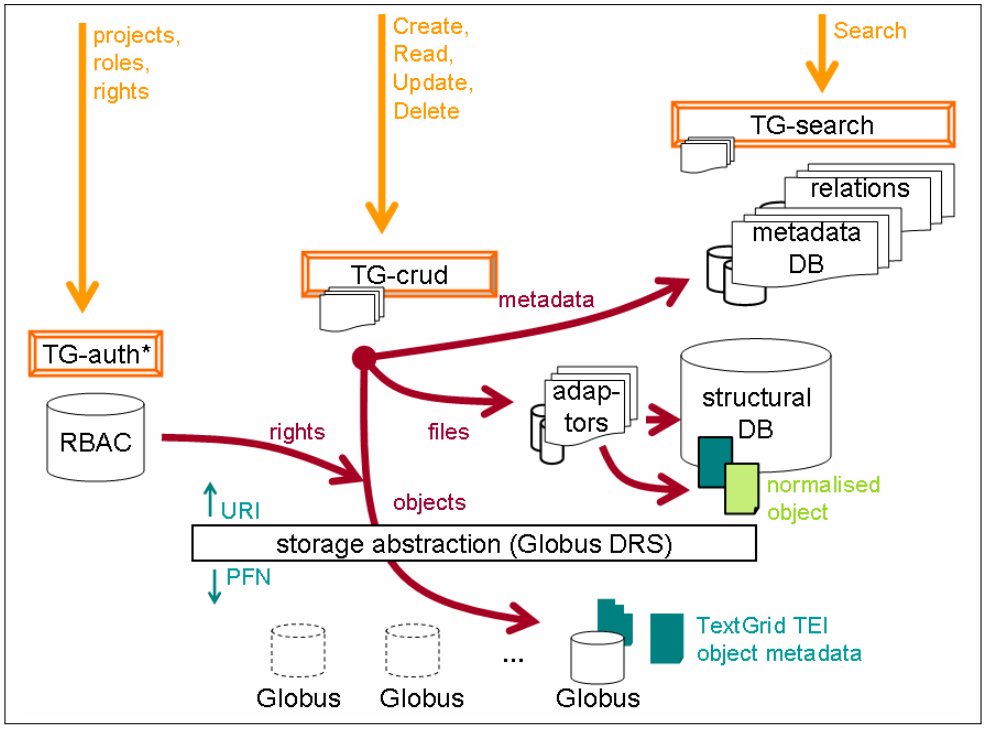


Figure 5.3: TextGrid – Ingest infrastructure processes

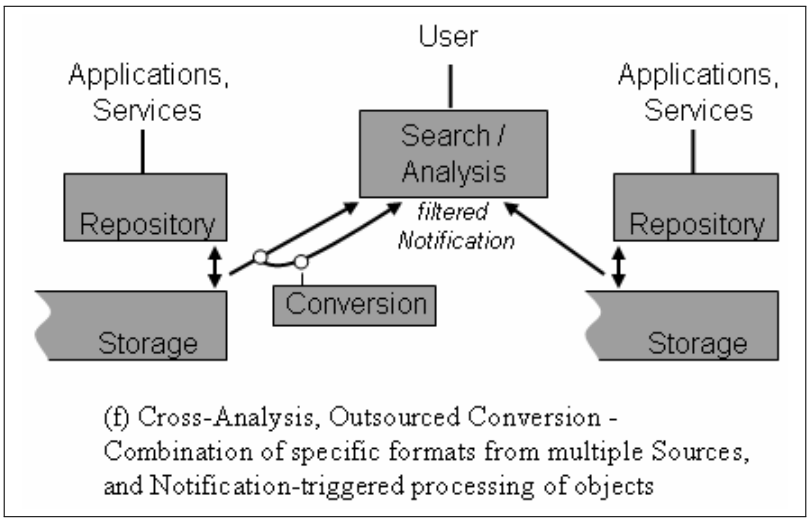


Figure 5.4: Schematic view of how repository federation creates shared applications and external workflows to resolve the scenarios 2.a, Scientific Analysis, as well as 2.b, Outsourcing Preservation Actions (cf. section 1.1.5).

6 Conclusions

This chapter reflects the contributions of this thesis, first by illustrating how the scenarios – presented as a "benchmark" for this thesis in section 1.1 – can be solved through the concepts developed in this thesis, and by re-iterating through the individual scientific contributions. Lastly, it also attempts to take a step back and reflect on the role of this thesis in the long-term evolution of digital infrastructure.

Every systems architect knows the moment of catharsis when a key design idea seems to resolve primary requirements and makes the entire system seem so obvious. This moment has recurred for the author with each of the systems described in this thesis – from the cloud-like storage infrastructure, to the TextGrid virtual research environment, to the Dariah e-Infrastructure.

Software engineering as a scientific field captures such design ideas in architectural styles, software patterns and other forms of documentation. Ideally, such documentation is generic, re-usable, and reflects on various options and approaches when building a specific type of system or when designing a system that embeds a specific technology. There is a host of such guidance on database modelling and theory, data warehouse design, and for other systems, however, we know of no comparable documentation for digital repositories. We can name numerous tutorials for specific repository systems [52, 206], design documents for specific repository interfaces [58, 59, 199], or architectural sketches of repository-based infrastructure for specific requirements and backgrounds [136, 215, 306], yet we cannot name any that documents architectural models for repository-based infrastructure in a system- and

context-independent manner.

Starting from this background, this thesis derives architectural design concepts from current experiences in digital repositories and related fields. It then carries on to develop the approach of "open repository environments", that bridge various infrastructures (e.g. repository, preservation, e-Infrastructure) to enable interoperation between currently existing repository islands and other agents (e.g. registries, added-value services, applications). The scenarios from section 1.1 give a tangible idea of how open repository environments are different from the repositories of today.

In conclusion, the following sections describe the novel concepts developed in this thesis, how these concepts solve the scenarios from section 1.1, and how this fits into the long-term evolution of repository-based infrastructure.

6.1 Contribution of this Thesis

This section starts by giving a brief description of the background in which this thesis was developed. After all, the numerous discussions with experts as well as the diverse contexts, in which the concepts were already applied, contribute significantly to the validation of this thesis. Subsequently, the key scientific contributions of this thesis are re-iterated.

The activities that led up to this thesis have been conducted in the time from 2001 to 2009 in various contexts and on all levels of a repository environment: e-Infrastructure, notably the German D-Grid initiative [249] and particularly the storage, metadata and data curation activities there [79, 122]; preservation and "trusted digital repositories" [78], notably the European projects ERPANET [77] and reUSE [92]; added value services, notably the infrastructure for authority files developed in LEAF [208]; as well as virtual research environments, notably the TextGrid and Dariah [86] projects. Each of these projects would have deserved a Ph.D. thesis, however this thesis emerged as a cross section through them. Particularly during the development of the TextGrid and the Dariah architectures, the author was fortunate enough to participate in and moderate discussions with experts from a variety of fields

and all over the world [83]. Lastly, this thesis not only emerged from personal experiences and discussions with experts, it is also validated through the implementations conducted during the TextGrid and the Dariah projects.

As mentioned above, there were few generic, reusable concepts for digital repositories when this thesis was started. Most repositories were tightly-integrated systems, and approaches for repository architecture were hardly transferable between systems. While this does not necessarily diminish the quality of the individual systems, it lowers interoperability between systems and between system components. Essentially it is the key hurdle to enable open repository environments, that enable interoperability across systems to exchange data, share system components, and reuse of independent agents (e.g. registries, added-value services and applications).

Tackling this lack of overarching architectural concepts, chapter 2.2 lays the groundwork for this thesis by defining a generic **repository reference architecture**. The development of this layered-architecture drew from the wealth of experiences out there in project descriptions, architectures of specific systems (e.g. DSpace, Fedora, EPrints, iRODS), and discussions in the community.

Striving for evolution rather than revolution, this architecture is as simple as possible without constraining functionalities, and it employs common repository terminology and standards where available. Its three layers reflect three layers of abstraction on digital objects, the contents of repositories: file, object, and application. To achieve interoperability across repositories and foster open repository environments, the following chapters address the interfaces between those layers respectively: the open storage and the federation interface.

Chapter 3 analyses the **open storage interface**, which connects the file and object layers. The objective for open storage is on-demand storage that is hosted externally to the repository system, which is of value to small organisations that otherwise cannot benefit from the economies of scale of large data centres, as well as large institutions that host multiple repositories and seek a single integrated, trustworthy storage infrastructure [174]. To ensure the reliability and efficiency of such an overall system, the interface needs to be independent from a specific repository system or other agent,

stable as agents evolve, and capable of serving multiple agents.

Such a storage infrastructure for repositories does not exist as of now. Existing generic storage paradigms including SAN or NAS [308], data grids, or storage clouds are insufficient for repository requirements, since file-based storage does not adequately represent the structure and components of digital objects: file-based data, metadata, and relations e.g. to other objects.

This thesis analysed the essential properties of an Open Storage Interface, without defining the nature of the underlying storage infrastructure. The findings were derived from a series of experiments with diverse distributed storage systems offering dissimilar interfaces: a virtual file system (Cleversafe), a dedicated storage handler (iRODS), as well as a RESTful interface (S3-like). While S3-like does not yet fully implement all the properties required for an open storage interface, it served to evaluate key properties: de-coupling of the underlying infrastructure technology, multi-agent interaction, as well as a simple interface that fosters interoperability and decentralisation. S3-like was implemented using Python WSGI and tested through deployment of a DSpace repository and other utilities.

Chapter 4 analyses the **Federation Interface**. As of today, federation approaches have been largely created *ad hoc* in order to satisfy a specific need, and they are overall limited mainly to metadata and search applications. Only gradually other aspects of digital objects and other types of applications are being served, but this growth again seems to proceed *ad hoc* rather than according to a scientific roadmap.

A mapping of existing federation approaches along attributes pertaining to both, the digital object and the overall information system – syntax and semantics (object), structure and patterns (system) – reveals that particularly the attribute "pattern" is insufficiently covered. The two dominant patterns – client-pull and harvesting – allow either direct access while being rather fragile and offering bad performance (client-pull), or they allow increased robustness and performance while the indirection via a harvester raises the overall implementation effort and may create inconsistencies through slow propagation. We introduce a "Notification" pattern as more direct than a harvesting pattern and more robust than a client-pull pattern. Particularly an

Atom-based hybrid push/pull notification pattern is robust, light-weight, and flexible, enabling federated applications that are more interactive than previously. Components for an Atom-based federation were experimentally developed and are likely to become a key building block of the Dariah European e-Infrastructure for the humanities.

Other than Dariah, the TextGrid e-Infrastructure is already live with users. The author of this thesis moderated the development of TextGrid's technical architecture and embedded in it many of the concepts developed in this thesis. Chapter 5 thus gives an abstract of the **open e-Infrastructure TextGrid** and describes the open storage interface as well as some of the federation mechanisms. These open repository concepts were implemented into the D-Grid environment and were pivotal in creating an environment that is open on three levels: as an infrastructure (primarily focused on virtualised storage), as a platform (where new functionalities can be added and re-used), as well as a software (targeting scholarly research in the humanities). In addition to illustrating the concepts developed in this thesis in a production environment, chapter 5 examines some of the social and organisational aspects that come with creating an open repository environment.

To sum up, the findings of this thesis effect users, researchers, and developers of repository-based environments. First of all with regard to usage, this thesis fosters interoperability in repository-based environments, which allows immediate and robust exchange of digital objects, sharing of system components, and a decentralised environment of independent agents (e.g. registries, added-value services, applications). How this places new sorts of applications into federated environments is discussed in the following section 6.2. This thesis also creates scientific frameworks where formerly *ad hoc* approaches prevailed and thereby introduces a new perspective into repository research. Lastly, through its systematic analysis this thesis hopes to shift the development of repository-based environments away from the creative and serendipitous, towards a craft that can be analysed, documented, and taught. We are well aware that this thesis only contributes but a step in this ongoing evolution, building extensively on existing experiences in the repository community and related fields. Yet, we hope it leads the right way and others follow suite.

6.2 Implementing the Scenarios

In its introduction this thesis identified a number of prototypical scenarios of open repository environments. The challenges posed by these scenarios is not in the design of individual systems, but rather in the interaction of multiple repositories and other agents (e.g. registries, added-value services). Today's state of the art in repository research circles mainly around the repository instead on the interactions between various repositories and other agents. As a consequence, islands of systems proliferate and are incapable of solving scenarios like the ones listed in section 1.1.

This section shows how these scenarios can be resolved through the concepts developed in this thesis. The common framework of this thesis' open repository environments guide the information architect faced with such scenarios and help avoid islands of idiosyncratic implementations.

1 Open Storage scenarios

1.a On-Demand Storage

1.b Distributed Access

1.c Repository Reconstruction

Decoupling repository storage from other repository functionalities is geared to ensure stability, raise scalability, and benefit from economies of scale as described in the first three scenarios 1.a-c. This is enabled through the Open Storage Interface (section 2.2.3), which allows multiple agents to access digital objects retained in a shared storage infrastructure. This independent storage infrastructure may outsource storage management tasks to a service provider that provides *On-Demand Storage* (e.g. institutional or community data center, commercial provider).

Chapter 3 discussed various storage models (i.e. virtual file system, a bespoke data grid, and a REST-ful intermediary) and whether they support *Distributed Access*, allowing multiple agents to access repository contents independent from each other. In addition to the technical interoperability needed that allows multiple agents to access a

storage infrastructure, Distributed Access requires a level of semantic interoperability that facilitates decentralisation.

In order to enable decentralised access to storage, the Open Storage Interface goes beyond generic file-based storage as e.g. provided through the WebDAV protocol or commercial clouds like Amazon S3, and defines standards for serialising digital objects. The OAI-ORE format is such a standard. This additional level of semantic interoperability is required, to allow heterogeneous agents to access digital objects independent of the software environments and application contexts of the agents involved. For example, this allows heterogeneous repositories to share a common object collection and enables *Repository Reconstruction*, as well as it enables added-value services to interact seamlessly with DSpace, iRODS, or other software packages.

The implementation of the scenarios is discussed in more detail in section 3.4. Chapter 3 also discusses the experiences from prototyping an Open Storage Interface using various .

2 Fedearation scenarios

2.a Scientific Analysis

2.b Task Tracking

2.c Out-Sourcing Preservation Actions

Federation in open repository environments enables exchange of all features of a digital object, including data, metadata, and relations. This enables use cases across disparate repository systems, like a *Scientific Analysis* mechanism that operates on the contents of digital objects, as well as a *Task Tracking* service that monitors the metadata of objects as they occur. Only by involving all features of digital objects in this way, disparate repositories can be united into a integrated virtual repository with use cases beyond search, which current Federation mechanisms focus on.

In addition to the extension of federation mechanisms to cover all features of a digital object, this thesis also identified gaps in the federation attributes of the information system, particularly with regard to patterns. The Atom-based Notification pattern

developed in section 4.2.2 and prototypically implemented in section 4.3 enables applications across disparate repositories that are more immediate and robust than with previous federation patterns (i.e. Query and Harvest).

Attributes of object features and the information system combined in an overarching federation framework (cf. section 4.1) allows to take more choice and more informed decisions in the design of a repository environment. For example, the *Scientific Analysis* scenario can be implemented using the Harvest pattern for one-off, resource-intense use cases, whereas a Notification pattern is better suited when a permanent service is established that needs to keep track of frequent changes in one or many sources.

The implementation of the *Scientific Analysis* and the *Task Management* scenarios are discussed in more detail in section 4.4, and another angle on the *Scientific Analysis* scenario as well as *Out-Sourcing Preservation Actions* is discussed in section 5.4.

6.3 Stability of Open Repository Environments

This section collates some trends that became clear while the thesis was written, thereby projecting the evolution beyond what was presented in this thesis to better understand the potential role of open repository environments. We argue that digital infrastructure is increasingly influenced by repository concepts and evolves towards interoperability between existing infrastructures and repository islands, which is exactly what this thesis aims to support through its concepts of open repository environments. Only the future can tell whether open repository environments will grow to be pervasive digital infrastructure, yet some indicators for that can be identified: the growing interest of funders into repositories ([144, 205, 253]), the increasing number of repository-based infrastructures striving to serve perpetually (rather than for a fixed-term project) (e.g. [86, 173, 229, 253]), and – as discussed in this section – an analysis of repository evolution up to now [85], as well as comparison with other infrastructures [140].

This thesis analysed a variety of interoperability protocols, both those established [198, 200] and those emerging [59, 279]. Due to their proliferation and their

fragmentation at the same time (cf. chapter 4.1), we could not cover them all in detail. One of the protocols we did not cover in depth is a precursor to OAI-PMH called Dienst [217]. Dienst was a visionary protocol at the time and found a major implementation with over 100 nodes in the NCSTRL (Networked Computer Science Technical Reference Library) [127], which later migrated to OAI-PMH before it was discontinued. The problem in Dienst was its goal to constrain the openness and universality of existing web technologies in order to raise quality and efficiency. While the technical approach was sound, this non-compliance with the the web being the pervasive digital environment at the time led to a lack of adoption and eventually to the disappearance of Dienst.

Repository environments will see more standards emerge and vanish, even ubiquitous ones such as OAI-PMH – this technical progression is the fundamental challenge underlying digital preservation. [317] How long- or short-lived concepts are, is often difficult to predict. For example, mixing of existing concepts has brought forth an extension of WebDAV by Tupelo [245] that adds metadata management to WebDAV's file transfer capabilities, as well as the Fuse interface for mounting repositories as a file system [214]. While a model of file hierarchies drastically constrains the capabilities of repository systems, these activities may live on as convenience interfaces or they may transform into yet another approach. Salient concepts are often carried on through technical change, just like some ideas of Dienst were incorporated into OAI-PMH and may continue to be reflected in the next technical manifestation.

There is no complete chain of evidence to scientifically predict the future of repositories, we can only provide some scientifically-backed indicators. In 1996 [166] diagnose file systems as dated and instead envision environments that are associative (rather than hierarchical) and based on metadata (rather than simple file names). According to their vision, data will be "stored and maintained on the Net" and compatibility will be automatic. Although unrelated, much of their vision has since been implemented in repository systems, yet there remains work to be done before repositories become a robust and pervasive infrastructure of that kind. Current activities such as the repository cloud DuraCloud [212] or the repository-based web framework Apache Sling [28] push further towards that goal of a pervasive repository

infrastructure. Already now there are numerous web portals like citeseer [32] or Flickr [40] that are essentially repository-like. Many of these things were not on the horizon when this thesis was incepted, and the evolution continues.

Figure 6.1 positions repositories on a Gartner Hype Curve [227] and analyses that repositories are only just entering the trough of disillusionment, and future growth will depend on increasing interoperability and interconnection. This mapping of repository evolution has been conducted by plotting the salient topics in repository research since the 1990, and validated through discussion with the repository community [85] (cf. "Positioning a Technology on the Hype Cycle", [227]). In the late 1990s there was a surge of open source repository packages and repository installation projects – every institution wanted to have one. [103, 257] At the peek of this hype, institutions recognised that while the technology is available, the organisational and social context often is not ready. Many repositories struggled for deposits, and many still do. [155, 162] Today, we can already see the slope of enlightenment in this evolution, as repository-based infrastructure is increasingly built in other context and for other purposes than academic publication management systems, where they originate from, including virtual research environments [136, 215] and organisational content management [271].

Current repository islands fail to attract a critical mass of users and block technological advance. This thesis has shown how to open up repositories and move from such islands to open repository environments. With these and similar projects creating the gateways and networks for robust and pervasive infrastructure to emerge, what is really needed is adoption and agreement across the communities; in other words: shared standards. However, standards are *per se* often unattractive for technological innovators as they see the sometimes time-consuming standards processes and immediately associate "standards" with "stasis". Yet, on the contrary, standards facilitate innovation [246]. Overall it seems, moving up the "slope of enlightenment" towards an open repository environment and eventually towards pervasive repository-based infrastructure still requires work, mostly of a social and organisational nature.

In the course of this thesis, the notions "repositories" and "infrastructure" have started to converge. e-Infrastructure and grid technologies in particular focus on virtualising

computational and storage resources. However, according to the vision by an expert group to the European Commission [145] also semantic services will be primary resources in the future. This thesis underlined that next to compute, storage, and services also (semantically annotated) data and digital objects deserve a place amongst the primary resources that build our digital research infrastructures. This position can recently also be traced in current e-Infrastructure activities, such as the focus on data curation in the D-Grid project WissGrid [122], the focus on preservation and management of digital objects in the planned European Grid Institution, as well as other infrastructure initiatives (e.g. [173, 253]) and the general discussion about data-centric science [73, 228, 247]. Ideally, this burgeoning approach of virtualising services and information will help closing the perceivable gap between digital infrastructure and application environments. [129]

If e-Infrastructure and repository concepts are indeed converging, where are we in this evolution? Comparison with other infrastructures like electrical power and railways [140] explains that all infrastructure technologies undergo several phases. After early phases of "system building" and "technology transfer and growth", almost the last stage is "consolidation and network formation". This stage sees gateways emerging to bridge disparate technological islands, such as different sized rail-road tracks or the large-scale conversion of AC power to DC.

The concepts developed in this thesis, as well as some of the related initiatives mentioned throughout this thesis (e.g. [20, 39, 250, 321]) is all about building such gateways and enabling interoperability. So we may be at the last stage of an emerging new infrastructure, "consolidation and network formation". But again, scientific proof of whether or not repository-based infrastructure will be pervasive infrastructure in the future can only be collected in hindsight.

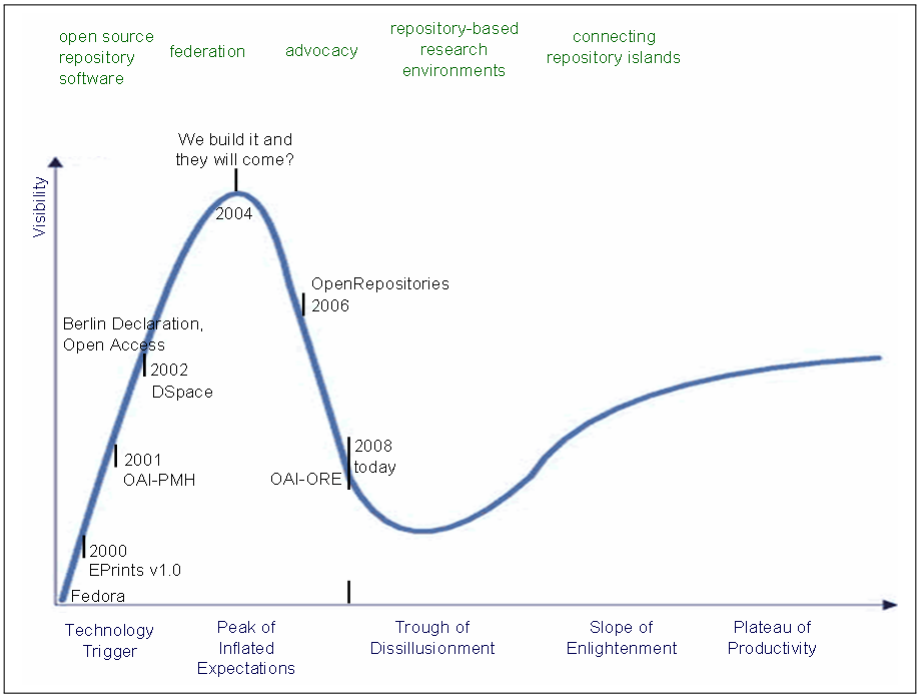


Figure 6.1: Repository Evolution based on Gartner Hype Curve [85]

Acknowledgements

Many people contributed to this thesis, sometimes without their knowledge. I apologize for that – at the time I did not know either. This thesis was not planned. It evolved from work on a variety of projects over almost the last 10 years, in which "repository infrastructure" turned out to be a recurrent issue.

Colleagues in the Technical University Vienna and the Austrian National Library (web archive); as well as from the EC projects ERPANET and reUSE! (trusted digital repositories) started the series. This thesis was written while employed at the State and University Library Goettingen (SUB) on a variety of projects (mostly on e-Infrastructure and research repositories). Work at the SUB is like an ongoing ride with a constant level of surprise. There is always a colleague next door, who happens to work on exactly the topic you are seeking help in. It is often at a conference somewhere on this planet where you find out, rather than in the office.

The project TextGrid had the biggest part in triggering this thesis. At the time TextGrid was designed, repository technologies proved insufficient for the distributed, open environment, yet the translation of repository concepts into e-Infrastructure proved a viable approach to us. The intense work with the 20+ members of the TextGrid project was a truly enriching experience. Thanks to all the TextGrid colleagues, and particularly to the core techies with whom even innocent breakfast tables were transformed into architectural planning boards.

While TextGrid provided a trigger, it is the contrast with other projects and discussion with colleagues from related projects that enabled the distance and abstract thinking

needed. As such, the colleagues from D-Grid, Dariah and all the other collaborators deserve great thanks as well. Particularly colleagues in the UK proved a source for seminal discussions.

Last but not least, the whole community must be mentioned at this point. Or rather in plural, the communities – discussions at OpenRepositories, the Open Grid Forum, IEEE conferences, and other venues were very inspiring. It is amazing how a group of people, internationally dispersed, with different backgrounds and goals, can develop amazingly similar patterns of thought, turning this dispersed group into a more or less coherent movement.

List of Figures

Bibliography

[1] Amazon Simple Storage Service – Developer Guide. API Documentation, Version 2006-03-01. http://docs.amazonwebservices.com/AmazonS3/2006-03-01/.

[2] DSpace/SRB Integration Project – Integration of Digital Library Lifecycle Management Processes with Data Grids. Project Website, Last updated April 2005. https://libnet.ucsd.edu/nara/.

[3] Berlin Declaration on Open Access to Knowledge in the Sciences and Humanities, October 2003. http://oa.mpg.de/openaccess-berlin/berlindeclaration.html.

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

[5] Introduction to Service Oriented Architecture. Toolbox for IT, Blog Post, 2005. http://it.toolbox.com/blogs/it-alignment/introduction-to-service-oriented-architecture-6213.

[6] To Stand the Test of Time – Long-term Stewardship of Digital Data Sets in Science and Engineering. A Report to the National Science Foundation from the ARL Workshop on New Collaborative Relationships: The Role of Academic Libraries in the Digital Data Universe, Arlington, VA, September 2006.

[7] Towards a European e-Infrastructure for e-Science Digital Repositories. Presentation at the e-IRG Workshop, Lisbon, October 2007. http://oldsite.e-irg.eu/meetings/2007-PT/4-e_IRG_Pres_Oct07_v3.pdf.

[8] ANU Data Management Manual: Managing Digital Research Data at the Australian National University. Manual, Information Literacy Program, August 2008. http://ilp.anu.edu.au/dm/.

[9] Automatisiertes Abliefern über Harvesting-Verfahren – Wege zur effizienten Ablieferung von Netzpublikationen. Deutsche Nationalbibliothek, August 2008. http://www.d-nb.de/netzpub/ablief/pdf/automatisierte_ablieferung.pdf.

[10] Contextual Query Language, CQL. SRU Version 1.2 Specifications, August 2008. http://www.loc.gov/standards/sru/specs/cql.html.

[11] DELOS Digital Library Reference Model. Version 0.98, February 2008. http://www.delos.info/ReferenceModel.

[12] DSpace Foundation and Fedora Commons Receive Grant from the Mellon Foundation for DuraSpace, November 2008. http://expertvoices.nsdl.org/hatcheck/2008/11/11/dspace-foundation-and-fedora-commons-receive-grant-from-the-mellon-foundation-for-duraspace/.

[13] EGI Blueprint. European Grid Initiative Design Study, December 2008. http://www.eu-egi.eu/blueprint.pdf.

[14] Extensible Markup Language (XML), 1.0 (Fifth Edition). W3C Recommendation, November 26 2008. http://www.w3.org/TR/REC-xml/.

[15] Fedora Commons Technology Roadmap V0.9, February 2008. http://www.fedora-commons.org/resources/roadmap.php.

[16] OASIS Reference Architecture for Service Oriented Architecture 1.0. Public Review Draft 1, April 23 2008. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm.

[17] PREMIS Data Dictionary for Preservation Metadata, version 2.0. Technical report, March 2008. http://www.loc.gov/standards/premis/.

[18] *Proceedings of the 2nd DORSDL Workshop on Digital Repositories*, Aarhus, Denmark, 2008. Collocated with the European Conference on Digital Libraries, ECDL. http://dorsdl2.cvt.dk/.

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

[20] JCR Tools For Sharing Content Between – Open Source Applications and Dissimilar Repositories. Project Fact Sheet, March 2009. http://rit.mellon.org/projects/PloneJCR.pdf.

[21] Specification for the Europeana Semantic Elements, version 3.1. Report, February 2009. http://dev.europeana.eu/.

[22] SRM-SRB Interface: A new milestone for grid interoperation. BELIEF-II, News Item, May 2009. http://www.beliefproject.org/news/srm-srb-interface-a-new-milestone-for-grid-interoperation.

[23] Update On Scientific Linux. In *Proceedings of the HEPiX Spring Meeting*, CERN, Geneva, Switzerland, May Dawson, Troy.

[24] Phase 1 of the Open-Systems Fedora Repository Development Project. Andrew W. Mellon Foundation Grant, funded December, 2001. http://www.lib.virginia.edu/digital/resndev/fedora_grant.html.

[25] Amazon Web Services. Homepage, Viewed August 2009. http://aws.amazon.com/.

[26] Apache Hadoop. Project Homepage, Viewed August 2009. http://hadoop.apache.org/.

[27] Apache Hadoop. Project Homepage, Viewed August 2009. http://code.google.com/p/partext/.

[28] Apache Sling. Project Homepage, Viewed August 2009. http://sling.apache.org/.

[29] Bamboo Program Document. Implementation Proposal, Version 1.0, Viewed August 2009. https://wiki.projectbamboo.org/display/BPUB/Bamboo+Program+Document.

[30] CERN openlab for DataGrid applications. Website, Viewed August 2009. http://www.cern.ch/openlab.

[31] CherryPy. Tool Website, Viewed August 2009. http://www.cherrypy.org/.

[32] Citeseer, Computer and Information Science Papers. Homepage, Viewed August 2009. http://citeseer.ist.psu.edu/.

[33] Cleversafe Dispersed Storage. Community Website, Viewed August 2009. http://www.cleversafe.org/.

[34] CloudClimate – watching the clouds. Project Website, Viewed August 2009. http://www.cloudclimate.com/s3/.

[35] Comparison of project management software. Wikipedia entry, Viewed August 2009. http://en.wikipedia.org/wiki/Comparison_of_project_management_software.

[36] DSpace 2.0/Pluggable Storage. DSpace Wiki, Viewed August 2009. http://wiki.dspace.org/index.php/DSpace_2.0/Pluggable_Storage.

[37] Eclipse File System, EFS. Eclipse Wiki, Viewed August 2009. http://wiki.eclipse.org/index.php/EFS.

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

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

[40] Flickr. Photo Community, Viewed August 2009. http://www.flickr.com/.

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

[42] Globus Grid Toolkit. Project Website, Viewed August 2009. www.globus.org.

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

[44] Grid File Access Library – GFAL. Online Manual, Viewed August 2009. http://www-numi.fnal.gov/offline_software/srt_public_context/GridTools/docs/data_gfal.html.

[45] Jargon Java API for Rules, Grids, and Online Networks. Online project page, Viewed August 2009. http://www.sdsc.edu/srb/jargon/.

[46] JetS3t – Java toolkit for Amazon S3. Project Homepage, Viewed August 2009. https://jets3t.dev.java.net/.

[47] JISC Common Repository Interfaces Group (CRIG). website, Viewed August 2009. http://www.ukoln.ac.uk/repositories/digirep/index/CRIG.

[48] JISC Repositories - communicating the idea. Ideascale Online Poll, Viewed August 2009. http://jiscrepository.ideascale.com/akira/panel.do?id=784.

[49] Names: Pilot national name and factual authority service. JISC Project Outline, Viewed August 2009. http://www.jisc.ac.uk/whatwedo/programmes/reppres/sharedservices/names.aspx.

[50] OAIster. Website, Viewed August 2009. www.oaister.org.

[51] Open Cloud Computing Interface WG (OCCI-WG). Online Gridforge by the Open Grid Forum OGF, Viewed August 2009. http://forge.gridforum.org/sf/projects/occi-wg.

[52] Open Repositories. Conference Series, Website, (Viewed August 2009). http://openrepositories.org/.

[53] OpenDOAR – Directory of Open Access Repositories. Website, Viewed August 2009. http://www.opendoar.org/.

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

[55] Researching e-Science Analysis of Census Holdings. Online Project Description, Viewed August 2009. http://www.ucl.ac.uk/infostudies/research/reach/.

[56] Sakai - collaboration and learning. Website, Viewed August 2009. http://sakaiproject.org/.

[57] SEE-GRID – South Eastern European GRid-enabled eInfrastructure Development. Project Website, Viewed August 2009. http://www.see-grid.org/.

[58] Standards at the Library of Congress. website, Viewed August 2009. http://www.loc.gov/standards/.

[59] SWORD, Simple Web-service Offering Repository Deposit. website, Viewed August 2009. http://www.swordapp.org/.

[60] UK National Grid Service. Homepage, Viewed August 2009. http://www.grid-support.ac.uk/.

[61] Dropbox. A Cloud-based File Management Tool, Viewed October 2009. https://www.getdropbox.com/.

[62] MDS - Monitoring and Discovery System. Globus Component, Website, Viewed October 2009. http://www.globus.org/mds/.

[63] Nagios - The Industry Standard in IT Infrastructure Monitoring. Homepage, Viewed October 2009. http://www.nagios.org/.

[64] Simple Cloud API. Project Website, Viewed October 2009. http://simplecloud.org/.

[65] Virtual Research Environment Landscape Study. JISC study, Viewed September 2009. http://www.survey.bris.ac.uk/kcl/vrelandscape.

[66] Stephen Abrams, Patricia Cruse, and John Kunze. Preservation Is Not a Place. *International Journal of Digital Curation*, 4(1), 2009.

[67] AIIM. Enterprise Content Management. Online Glossary, Viewed July 2009. http://www.aiim.org/what-is-ecm-enterprise-content-management.aspx.

[68] AIIM. Web Content Management. Online Glossary, Viewed July 2009. http://www.aiim.org/What-is-Web-CMS-WCM-System-Content-Management.aspx.

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

[70] Christopher Alexander, Sara Ishikawa, and Murray Silverstein. *A Pattern Language : Towns, Buildings, Construction.* Oxford University Press, New York. USA, 1977.

[71] Gabrielle Allen, Kelly Davis, Thomas Dramlitsch, Tom Goodale, Ian Kelley, Gerd Lanfermann, Jason Novotny, Thomas Radke, Kashif Rasul, Michael Russell, Ed Seidel, and Oliver Wehrens. The GridLab Grid Application Toolkit. *High-Performance Distributed Computing, International Symposium on*, 0:411, 2002.

[72] American Council of Learned Societies Commission on Cyberinfrastructure for the Humanities and Social Sciences. Our Cultural Commonwealth, December 2006. http://www.acls.org/cyberinfrastructure/.

[73] Chris Anderson et al. The Petabyte Age: Because More Isn't Just More – More Is Different. *Wired Magazine*, 16(07), 2008. http://www.wired.com/science/discoveries/magazine/16-07/pb_intro.

[74] Andreas Rauber Andreas Aschenbrenner. *Web Archiving*, chapter Mining Web Collections, pages 153 – 176. Springer, 2006.

[75] UK Data Archive. Managing and Sharing Data - a best practice guide for reserachers, 2009. http://www.data-archive.ac.uk/news/publications/managingsharing.pdf.

[76] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy H. Katz, Andrew Konwinski, Gunho Lee, David A. Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. Above the Clouds: A Berkeley View of Cloud Computing. Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley, Feb 2009. http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html.

[77] Andreas Aschenbrenner. ERPANET. In *Proceedings of the Chinese-European Workshop on Digital Preservation (iPRES)*, Beijing, China, July 2004. http://rdd.sub.uni-goettingen.de/conferences/ipres04/.

[78] Andreas Aschenbrenner. Ziel: vertrauenswürdiges Archiv - ein methodischer Vergleich der reUSE Demonstratoren. Presentation at the nestor Workshop "Vertrauenswürdige digitale Langzeitarchive: Kriterien und deren Bewertung", Munich, Juni 21 2005.

[79] Andreas Aschenbrenner. Data Management in TextGrid. In *Proceedings of the 2nd D-Grid Data Management Workshop*, Zuse-Institut Berlin, May 2006.

[80] Andreas Aschenbrenner. e-Humanities in Europe – DARIAH. In *Proceedings of the Open Grid Forum 20*, Manchester, UK, May 7-11 2007.

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

[82] Andreas Aschenbrenner. Repositories at the Goettingen State and University Library. Presented at the DARIAH Meeting, Max Planck Digital Library (MPDL), Munich, March 19-20 2009.

[83] Andreas Aschenbrenner, Tobias Blanke, Neil P Chue Hong, Nicholas Ferguson, and Mark Hedges. A Workshop Series for Grid/Repository Integration. *D-Lib Magazine*, 15(1/2), January/February 2009.

[84] 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 für Kunst und Kultur. *Bibliothek, Forschung und Praxis*, 31(1), 2007.
`http://www.bibliothek-saur.de/2007_1/011-021.pdf.`

[85] 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(11/12), November/December 2008.

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

[87] Andreas Aschenbrenner, Tobias Blanke, and Mark Hedges. Synergies between Grid and Repository Technologies – a Methodical Mapping. *IEEE International Conference on e-Science*, 0:778–781, 2008.

[88] Andreas Aschenbrenner, Flavia Donno, and Senka Drobac. Infrastructure for Interactivity – Decoupled Systems on the Loose. In *Proceedings of the IEEE Digital Ecosystems and Technologies Conference (DEST)*, Istanbul, Turkey, June 2009.

[89] Andreas Aschenbrenner and Malte Dreyer. (digital library architecture)^2 - service patterns for large-scale digital libraries. In Paolo Manghi, Pasquale Pagano, and Pavel Zezula, editors, *Proceedings of the First DELOS Workshop on Very Large Digital Libraries (VLDL), held in conjunction with ECDL 2008*, volume 37 of *ACM SIGMOD Record*, December 2008.

[90] Andreas Aschenbrenner and Nicholas Ferguson. Open Grid Forum, Digital Repositories Research Group. OGF GridForge, Viewed August 2009. http://www.ogf.org/gf/group_info/view.php?group=dr-rg.

[91] Andreas Aschenbrenner, Mark Hedges, Tobias Blanke, and Frank Schwichtenberg. (repository +/= e-Infrastructure). In *Proceedings of the Third International Conference on OpenRepositories*, Southampton, United Kingdom, 1-4 April 2008.

[92] Andreas Aschenbrenner and Max Kaiser. White Paper on Digital Repositories. reUSE! Deliverable, March 2005. http://www2.uibk.ac.at/reuse/docs/reuse-d11_whitepaper_10.pdf.

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

[94] Andreas Aschenbrenner and Katja Meffert. Wissenschaftliche Infrastruktur in den Geisteswissenschaften? – Eine Wegbeschreibung. *Jahrbuch für Computerphilologie*, 9, 2007. Paderborn: mentis Verlag 2009.

[95] Andreas Aschenbrenner and Andreas Rauber. Die Bewahrung unserer Online-Kultur. Vorschläge zu Strategien der Webarchivierung. *Sichtungen*, 2003.

[96] Andreas Aschenbrenner and Bing Zhu. iRODS-Fedora Integration. iRODS Wiki, March 2009. https://www.irods.org/index.php/Fedora.

[97] Daniel E. Atkins, Kelvin K. Droegemeier, Stuart I. Feldman, Hector Garcia-Molina, Michael L. Klein, David G. Messerschmitt, Paul Messina, Jeremiah P. Ostriker, and Margaret H. Wright. Revolutionizing Science and Engineering Through Cyberinfrastructure: Report of the National Science Foundation Blue-Ribbon Advisory Panel on Cyberinfrastructure, January 2003. http://www.nsf.gov/publications/pub_summ.jsp?ods_key=cise051203.

[98] Thomas Baker. A Grammar of Dublin Core. *D-Lib Magazine*, 6(10), October 2000.

[99] Jeroen Bekaert, Xiaoming Liu, and Herbert Van de Sompel. aDORe: a Modular and Standards-based Digital Object Repository at the Los Alamos National Laboratory. In *JCDL '05: Proceedings of the 5th ACM/IEEE-CS joint Conference on Digital Libraries*, pages 367–367, New York, NY, USA, 2005. ACM.

[100] Francine Berman. Got data?: a guide to data preservation in the information age. *Commun. ACM*, 51(12):50–56, 2008.

[101] Tobias Blanke and Mark Hedges. Providing linked-up access to Cultural Heritage Data. In *Proceedings of the ECDL 2008 Workshop Information Access to Cultural Heritage (IACH)*, Aarhus, Denmark, September 18 2008.
`http://ilps.science.uva.nl/IACH2008/proceedings/proceedings.html`.

[102] Alan Blunt. From Database to Learning Service – The Sustainable Business. In *Proceedings of the ERPANET Workshop on Business Models related to Digital Preservation*, Amsterdam, The Netherlands, September 2004.

[103] Uwe M. Borghoff et al. Vergleich bestehender Archivierungssysteme. nestor Report, 2005.

[104] Martha L. Brogan. Contexts and Contributions: Building the Distributed Library. Technical report, University of Pennsylvania, 2006.
`http://repository.upenn.edu/librarypapers/31`.

[105] Lou Burnard, Katherine O'Brien O'Keeffe, and John Unsworth, editors. *Electronic Textual Editing*, chapter Storage, Retrieval, and Rendering [Sebastian Rahtz]. Modern Language Association of America, September 2006.

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

[107] Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, and Michael Stal. *Pattern-Oriented Software Architecture, Volume 1: A System of Patterns*. John Wiley & Sons, August 1996.

[108] Priscilla Caplan. Repository to Repository Transfer of Enriched Archival Information Packages. *D-Lib Magazine*, 14(11/12), November/December 2008.

[109] Leona Carpenter. OAI for Beginners – the Open Archives Forum online tutorial, 2003. http://www.oaforum.org/tutorial/english/intro.htm.

[110] Tony Cass. The CASTOR project – CERN Advanced STORage manager. In *Proceedings of the first HEPNT-HEPiX Meeting*, Stanford Linear Accelerator Center, October 1999. http://castor.web.cern.ch/castor/.

[111] CCSDS - Consultative Committee for Space Data Systems. *Reference Model for an Open Archival Information System (OAIS)*, 2003.

[112] Arcot Rajasekar Michael Wan Chaitanya Baru, Reagan Moore. The SDSC Storage Resource Broker. In *Proceedings of the CASCON'98 Conference*, Toronto, Canada, Nov.30-Dec.3 1998.

[113] 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. In Julio Gonzalo, Costantino Thanos, M. Felisa Verdejo, and Rafael C. Carrasco, editors, *Proceedings of the ECDL 2006*, volume 4172 of *Lecture Notes in Computer Science*, pages 524–527. Springer, 2006.

[114] Ryan Chute and Herbert Van de Sompel. Introducing djatoka – A Reuse Friendly, Open Source JPEG 2000 Image Server. *D-Lib*, 14(9/10), September/October 2008.

[115] OASIS Committee. *Reference Model for Service Oriented Architecture 1.0*. OASIS Standard, October 12 2006. http://www.oasis-open.org/specs/index.php#soa-rmv1.0.

[116] Fedora Commons. Command-Line Utilities. Software Documentation, Viewed August 2009.
http://www.fedora-commons.org/confluence/display/FCR30/Command-Line+Utilities.

[117] Fedora Commons and DSpace Federation. DuraSpace – Trust and reliability in the Cloud. Midterm Report to the Mellon Foundation, February 2009.
http://rit.mellon.org/projects/DuraSpace.pdf.

[118] The U. Consortium, editor. Addison-Wesley Professional, 2006.

[119] Oscar Corcho, Pinar Alper, Ioannis Kotsiopoulos, Paolo Missier, Sean Bechhofer, and Carole Goble. An overview of S-OGSA: A Reference Semantic Grid Architecture. *Web Semantics: Science, Services and Agents on the World Wide Web*, 4(2):102–115, June 2006.

[120] Greig Cowen. Tier-2 optimisation of dCache and DPM. Presentation at the HEPiX Spring Workshop, April 2006.

[121] Gregory Crane. What Do You Do with a Million Books? *D-Lib Magazine*, 12(3), March 2006.

[122] D-Grid. WissGrid – Grid for Science. Website, Viewed August 2009. http://www.wissgrid.de.

[123] DART. *Fedora-SRB Database integration module*. University of Queensland, 2007. http://www.itee.uq.edu.au/~eresearch/projects/dart/outcomes/FedoraDB.php.

[124] Statistical Data and Metadata Exchange Initiative. SDMX Guidelines for the Use of Web Services, November 2005. http://www.sdmx.org/docs/2_0/SDMX_2_0$%$20SECTION_07_WebServicesGuidelines.pdf.

[125] Statistical Data and Metadata Exchange Initiative. SDMX User Guide, January 2007. http://sdmx.org/docs/2007/Conf07/doc%2031%20Capacity%20Building%20Room%20Document%20-%20UserGuide%20-%20Working%20Draft.doc.

[126] Rob Davies. Europeana: An Infrastructure for Adding Local Content. *Ariadne*, 57, October 2008.

[127] James R. Davis and Carl Lagoze. NCSTRL: design and deployment of a globally distributed digital library. *J. Am. Soc. Inf. Sci.*, 51(3):273–280, 2000.

[128] Michael Day. Metadata in a nutshell. *Information Europe*, 6(2), 2001.

[129] David De Roure. The Grid and the Cloud: Reflections and Projections. In *Proceedings of the international Conference on Grid computing, high-performAnce and Distributed Applications (GADA)*, Monterrey, Mexico, November 2008.

[130] David De Roure and Carole Goble. Research Objects for Data Intensive Research. In *Submitted to IEEE e-Science 2009, Oxford UK*, December 2009. http://wiki.myexperiment.org/index.php/Research_Objects_for_Data_Intensive_Research.

[131] David De Roure, Carole Goble, and Robert Stevens. Designing the myExperiment Virtual Research Environment for the Social Sharing of Workflows. In *Proceedings of the Third IEEE International Conference on e-Science and Grid Computing*, pages 603–610, Bangalore, India, December 10-13 2007.

[132] Ray Denenberg. Search Web Services – The OASIS SWS Technical Committee Work. *D-Lib Magazine*, 15(1/2), January/February 2009.

[133] Elly Dijk, Chris Baars, Arjan Hogenaar, and Marga van Meel. NARCIS: The Gateway to Dutch Scientific Information. In Bob Martens and Milena Dobreva, editors, *Digital Spectrum: Integrating Technology and Culture - Proceedings of the 10th International Conference on Electronic Publishing*, pages 49–58, Bansko, Bulgaria, June 14-16 2006. ELPUB2006.

[134] Tharam S. Dillon, Chen Wu, and Elizabeth Chang. Reference Architectural Styles for Service-Oriented Computing . In *Network and Parallel Computing. Proceedings of the IFIP International Conference NPC 2007*, volume 4672 of *Lecture Notes in Computer Science*, pages 543–555, Dalian, China, September 2007. Springer.

[135] Cory Doctorow. Google's new data-center cools with weather prediction, not electricity. *Boingboing*, July 2009. http://boingboing.net/2009/07/15/googles-new-data-cen.html.

[136] Malte Dreyer, Natasa Bulatovic, Ulla Tschida, and Matthias Razum. eSciDoc – a Scholarly Information and Communication Platform for the Max Planck Society. In *Proceedings of the German e-Science Conference*, 2007.

[137] DSpace. Storage Layer. System Documentation, Viewed August 2009. http://www.dspace.org/index.php/Architecture/technology/system-docs/storage.html.

[138] Jürgen Dunkel, Andreas Eberhart, Stefan Fischer, Carsten Kleiner, and Arne Koschel. *Systemarchitekturen für verteilte Anwendungen. Client-Server,*

Multi-Tier, SOA, Event Driven Architecture, P2P, Grid, Web 2.0. Hanser Fachbuch, 2008.

[139] Phillip J. Eby. Python Web Server Gateway Interface, WSGI. Python Enhancement Proposal, PEP 333, v1.0, December 2003.
http://www.python.org/dev/peps/pep-0333/.

[140] Paul N. Edwards, Steven J. Jackson, Geoffrey C. Bowker, and Cory P. Knobel. Understanding Infrastructure: Dynamics, Tensions, and Design. Report of a Workshop on the History and Theory of Infrastructure: Lessons for New Scientific Cyberinfrastructures, January 2007.
http://deepblue.lib.umich.edu/bitstream/2027.42/49353/3/UnderstandingInfrastructure2007.pdf.

[141] Victor Eijkhout. Character encoding. Course Handout, August 2004.
http://www.cs.utk.edu/~eijkhout/594-LaTeX/handouts/fonts/encoding-tutorial.pdf.

[142] Google Engineering. The Data Liberation Front. Website, Viewed August 2009.
http://www.dataliberation.org/.

[143] Michael Ernst, Patrick Fuhrmann, Martin Gasthuber, Tigran Mkrtchyan, and Charles Waldman. dCache, a distributed storage data caching system. In *Proceedings of the International Conference on Computing in High Energy and Nuclear Physics, CHEP*, Beijing, China, September 2001.

[144] eInfrastructure EU IST. European Infrastructure for Repositories of Scientific Information. Liaison Workshop, June 2006.
http://cordis.europa.eu/ist/rn/ri-cnd/wshop-080606.htm.

[145] European Commission. *From Grids to Service-Oriented Knowledge Utilities*. Office for Official Publications of the European Communities, Luxembourg, 2006.

[146] Adam Farquhar and Helen Hockx-Yu. Planets: Integrated Services for Digital Preservation. *International Journal of Digital Curation*, 2(3), 2007.

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

[148] The Fedora Content Model Architecture (CMA). Fedora Commons Report, Viewed August 2009.
http://www.fedora-commons.org/documentation/3.0b1/userdocs/digitalobjects/cmda.html.

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

[150] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. Technical report, The Internet Society, June 1999.

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

[152] Dale Flecker and Neil McLean. Digital Library Content and Course Management Systems: Issues of Interoperation: Report of a study group. *Digital Library Content and Course Management Systems: Issues of Interoperation*, 2004.
http://www.diglib.org/pubs/dlf100/.

[153] European Commission / German Commission for UNESCO. Open Access. Opportunities and Challenges - a Handbook, 2008.
http://www.unesco.de/openaccess.html.

[154] Data Management Forum. Information Lifecycle Management Roadmap. Technical report, SNIA, Storage Networking Industry Association, October 12 2004. http://www.snia.org/forums/dmf/programs/ilmi/SNIA-DMF_ILM_Roadmap_041012.pdf.

[155] Andrea Foster. Papers Wanted: Online archives run by universities struggle to attract material. *Chronicle of Higher Education*, June 2004.

[156] Ian Foster. What is the Grid? – a three point checklist. *GRIDtoday*, 1(6), July 2002. http://www.gridtoday.com/02/0722/100136.html.

[157] Ian Foster. A critique of "Using Clouds to Provide Grids...". Ian Foster's blog, September 2008. http://ianfoster.typepad.com/blog/2008/09/a-critique-of-u.html.

[158] Ian Foster and Carl Kesselman. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan-Kaufmann, July 1998.

[159] Ian Foster, Carl Kesselman, Gene Tsudik, and Steven Tuecke. A security architecture for computational grids. In *CCS '98: Proceedings of the 5th ACM conference on Computer and communications security*, pages 83–92, New York, NY, USA, 1998. ACM.

[160] Ian Foster, Carl Kesselman, and Steven Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International Journal of Supercomputer Applications*, 15, 2001.

[161] Ian Foster, H. Kishimoto, A. Savva, Dave Berry, A. Djaoui, Andrew Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell, and J. Von Reich. The Open Grid Services Architecture, Version 1.5. Technical report, Open Grid Forum, July 2006. `http://forge.gridforum.org/projects/ogsa-wg`.

[162] Nancy Fried Foster and Susan Gibbons. Understanding Faculty to Improve Content Recruitment for Institutional Repositories. *D-Lib Magazine*, January 2005.

[163] Apache Software Foundation. Apache Abdera. Version 0.4, Viewed August 2009. `http://abdera.apache.org/`.

[164] European Science Foundation. European Reference Index for the Humanities (ERIH). Online Topic Gateway, Viewed August 2009. `http://www.esf.org/research-areas/humanities/research-infrastructures-including-erih.html`.

[165] Gesellschaft für Informatik e.V. Data Warehouse. Informatiklexikon. `http://tinyurl.com/mj5fkw`.

[166] Eric Freeman and David Gelernter. Lifestreams: A Storage Model for Personal Data. *SIGMOD Record*, 25(1), March 1996.

[167] Marc Fresko. MoReq2: the New Model for Developing, Procuring Electronic Records Management Systems. *Information Management Journal*, July 2008.

[168] Fabrizio Gagliardi, Bob Jones, Francois Grey, Marc-Elian Begin, and Matti Heikkurinen. Building an infrastructure for scientific Grid computing: status and goals of the EGEE project. *Philosophical Transactions of the Royal Society*, 363(1833):1729–1742, August 2005.

[169] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design patterns: elements of reusable object-oriented software*. Addison-Wesley Professional, 1995.

[170] Simson L. Garfinkel. An Evaluation of Amazon's Grid Computing Services: EC2, S3 and SQS. Technical report, Center for Research on Computation and Society, Harvard University, Cambridge, MA, 2007.
http://simson.net/clips/academic/2007.Harvard.S3.pdf.

[171] Y.Y. Goland, E.J. Whitehead, A. Faizi, S.R. Carter, and D. Jensen. *Extensions for Distributed Authoring and Versioning on the World Wide Web – WEBDAV*. Internet Engineering Task Force (IETF), September 29 1997.
http://tools.ietf.org/html/draft-ietf-webdav-protocol-03.

[172] Ann Green, Stuart Macdonald, and Robin Rice. Policy-making for Research Data in Repositories: A Guide, May 2009. http://www.disc-uk.org/docs/guide.pdf.

[173] ANDS Technical Working Group. Towards the Australian Data Commons – A proposal for an Australian National Data Service. Australian Government, Department of Education, Science and Training, October 2007.
http://www.pfc.org.au//pub/Main/Data/TowardstheAustralianDataCommons.pdf.

[174] Research Libraries Group and OCLC. Trusted Digital Repositories: Attributes and Responsibilities, 2002.

[175] XML Protocol Working Group. SOAP Version 1.2 Part 0: Primer (Second Edition). W3C Recommendation, April 2007. http://www.w3.org/TR/soap/.

[176] Junmin Gu, Alex Sim, and Arie Shoshani. The Storage Resource Manager: Interface Specification. Version 2.2, May 2008.
http://sdm.lbl.gov/srm-wg/doc/SRM.v2.2.html.

[177] Wolfgang Gürich (ed.). Betriebskonzept für die D-Grid Infrastruktur. In *Fachgebiet 2.2 - Aufbau des Kern-D-Grid und Integration von Ressourcen und Diensten*. October 2007.

[178] Richard Harada. Are you prepared for long-term data preservation? - first in/first out. *Computer Technology Review*, October 2003.
http://findarticles.com/p/articles/mi_m0BRZ/is_10_23/ai_111062977/.

[179] Bernhard Haslhofer and Bernhard Schandl. The OAI2LOD Server: Exposing OAI-PMH Metadata as Linked Data. In *Proceedings of the WWW 2008 Workshop: Linked Data on the Web*, Beijing, China, April 22 2008.

[180] Erik Hatcher and Otis Gospodnetić. *Lucene in Action*. Manning Publications, December 2004.

[181] Roman Hausner. Semantic Text Mining - linguistische Tools im Preprocessing von Text Mining Methoden. Master's thesis, Georg-August-Universität Göttingen, April 2009.
http://www.informatik.uni-goettingen.de/studies/courses/theses.htm?&show=year&year=2009.

[182] Margaret Hedstrom and Christopher A Lee. Significant Properties of Digital Objects: Definitions, Applications, Implications. In *Proceedings of the DLM-Forum*, May 2002.

[183] Rachel Heery and Sheila Anderson. Digital Repositories Review. UKOLN, AHDS, 2005. http://www.jisc.ac.uk/uploaded_documents/digital-repositories-review-2005.pdf.

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

[185] Hans-Werner Hilse and Jochen Kothe. Implementing Persistent Identifiers: Overview of Concepts, Guidelines and Recommendations, November 2006.
http://xml.coverpages.org/ECPA-PersistentIdentifiers.pdf.

[186] 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*, 13(5/6), May/June 2007.

[187] Steve Hitchcock, Dave Tarrant, Adrian Brown, Ben O'Steen, Neil Jefferies, and Leslie Carr. Towards smart storage for repository preservation services. In *Proceedings of the Fifth International Conference on Preservation of Digital Objects (iPRES)*, London, UK, September 29-30 2008.

[188] Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, and York Sure. *Semantic Web: Grundlagen (eXamen.press)*. Springer, October 2007.

[189] Gregor Hohpe. Programming Without a Call Stack – Event-driven Architectures. Whitepaper, 2006. http://www.enterpriseintegrationpatterns.com/docs/EDA.pdf.

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

[191] Gregor Hohpe and Bobby Wolf. *Enterprise Integration Patterns – Designing, Building, and Deploying Messaging Solutions*. Addison-Wesley, December 2008.

[192] Annette Holtkamp. Inspire: Innovatives Informationsmanagement in der Hochenergiephysik. In *Proceedings of the 98. Deutscher Bibliothekartag*, Erfurt, 2009.

[193] HostedFTP.com. Amazon S3 and EC2 Performance Report, February 2009. http://www.slideshare.net/richm711/amazon-performance-report2.

[194] Jane Hunter and Sharmin Choudhury. A Semi-Automated Digital Preservation System based on Semantic Web Services. In *Proceedings of the Joint Conference on Digital Libraries*, Tucson, Arizona, USA, June 2004.

[195] IMS. IMS Digital Repositories v1.0. Final specification, 2003. http://www.imsglobal.org/digitalrepositories/.

[196] Content Management Software Info. E-learning products for Plone. Web Portal, Viewed August 2009. http://www.contentmanagementsoftware.info/plone/e-learning.

[197] Open Archives Initiative. The Open Archives Initiative Protocol for Metadata Harvesting. OAI Protocol, Version 1.0, 2001. http://www.openarchives.org/OAI/1.0/openarchivesprotocol.htm.

[198] Open Archives Initiative. The Open Archives Initiative Protocol for Metadata Harvesting. OAI Protocol, Version 2.0, 2002.
http://www.openarchives.org/OAI/openarchivesprotocol.html.

[199] Open Archives Initiatives. Standards for Web Content Interoperability. website, Viewed August 2009. http://www.openarchives.org/.

[200] American National Standards Institute, editor. *Application Service Definition and Protocol Specification for Open Systems Interconnection*. NISO Press, Bethesda, Maryland, U.S.A., 1992. http://www.cni.org/pub/NISO/docs/Z39.50-1992/.

[201] iRODS. Glossary. System Documentation, Viewed August 2009.
https://www.irods.org/index.php/iRODS_Glossary.

[202] Pierre-Yves Jallud. The pilot-project for long-term preservation. Presented at the DARIAH Meeting, Max Planck Digital Library (MPDL), Munich, March 19-20 2009.

[203] Mattias Jansson. Information Systems Integration and Interoperability – what it is and how to achieve it between two information systems. Master thesis, KTH Electrical Engineering, Stockholm, Sweden, 2007.
http://www.ee.kth.se/php/modules/publications/reports/2007/XR-EE-ICS_2007_009.pdf.

[204] Shantenu Jha, Andre Merzky, and Geoffrey Fox. Using Clouds to Provide Grids Higher-Levels of Abstraction and Explicit Support for Usage Modes. OGF Report, January.

[205] JISC. Repositories and Preservation Programme. Funding Programme 2006-2009.
http://www.jisc.ac.uk/programme_rep_pres.aspx.

[206] Gareth J. Johnson. The Repositories Support Project. In *Presentation at the JISC e-Science All Hands Meeting*. University of Nottingham, September 12 2007.

[207] Robert Kahn and Robert Wilensky. A Framework for Distributed Digital Object Services. Technical report, May 1995. Republished by Springer in 2006.

[208] Max Kaiser, Hans-Jörg Lieder, Kurt Majcen, and Heribert Vallant. New Ways of Sharing and Using Authority Information: The LEAF Project. *D-Lib Magazine*, 9(11), 2003.

[209] Norman L. Kerth and Ward Cunningham. Using Patterns to Improve Our Architectural Vision. *IEEE Softw.*, 14(1):53–59, 1997.

[210] Amy Jo Kim. *Community Building on the Web – Secret Strategies for Successful Online Communities* . Peachpit Press, 2000.

[211] Ralph Kimball. The Soul of the Data Warehouse. *intelligent enterprise*, March/April 2003. `http://www.ralphkimball.com/html/articles.html`.

[212] Michele Kimpton. DuraCloud: Managing Durable Data in the Cloud. Presentation at the 2009 NDIIPP Partners Meeting, Washington, DC, June 2009.

[213] Jackie Knowles and Steve Bailey. Institutional repositories and records management: overlaps, obstacles and opportunities. In *Presentation at the HE and FE Records Management and Information Compliance Group Meeting*. University of Manchester, December 4 2007.

[214] Rebecca Koeser. FedoraFS. dev8D RepoChallenge Winner, May 2009. `http://dev8d.jiscinvolve.org/2009/05/20/repochallenge-winners/`.

[215] Dean B. Krafft, Aaron Birkland, and Ellen J. Cramer. Ncore: architecture and implementation of a flexible, collaborative digital library. In *JCDL '08: Proceedings of the 8th ACM/IEEE-CS joint conference on digital libraries*, pages 313–322, New York, NY, USA, 2008. ACM.

[216] Marc Wilhelm Küster, Thorsten Vitt, and Andreas Aschenbrenner, editors. *TextGrid Tutorial*, Istanbul, Turkey, June 2009. IEEE Digital Ecosystems and Technologies Conference (DEST).

[217] C. Lagoze, E. Shaw, J.R. Davis, and D.B. Krafft. Dienst Implementation Reference Manual. Technical report, Cornell Computer Science, May 1995.

[218] Carl Lagoze. The Warwick Framework – A Container Architecture for Diverse Sets of Metadata. *D-Lib Magazine*, July/August 1996.

[219] Carl Lagoze, Dean B. Krafft, Sandy Payette, and Susan Jesuroga. What Is a Digital Library Anymore, Anyway? – Beyond Search and Access in the NSDL. *D-Lib Magazine*, 11(11), November 2005.

[220] Carl Lagoze and Sandra Payette. An Infrastructure for Open Architecture Digital Libraries. Technical report, Cornell Computer Science, June 1998. http://ecommons.library.cornell.edu/bitstream/1813/7344/1/98-1690.pdf.

[221] Carl Lagoze, Sandy Payette, Edwin Shin, and Chris Wilper. Fedora: an architecture for complex objects and their relationships. *Int. J. Digit. Libr.*, 6(2):124–138, 2006.

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

[223] Erwin Laure et al. Programming the Grid with gLite. In *Computational Methods in Science and Technology*, volume 12, 2006.

[224] Ralph LeVan. OpenSearch and SRU: Continuum of Searching. *Information Technology and Libraries (ITAL)*, 25(3):151–153, September 2006. http://www.oclc.org/research/publications/archive/2006/levan-ital.pdf.

[225] Eliezer Levy and Abraham Silberschatz. Distributed file systems: concepts and examples. *ACM Comput. Surv.*, 22(4):321–374, 1990.

[226] Leo Lewis. Scandal over lost pensions may be the final straw for ruling party. *The Times*, July 3 2007. http://www.timesonline.co.uk/tol/news/world/asia/article2017410.ece.

[227] A. Linden and J. Fenn. Understanding Gartner's Hype Cycles. Strategic Analysis Report, May 2003. http://carbon.cudenver.edu/~jgerlach/emergingtechnologyOL/FirstReadings/HypeCycleIntro.pdf.

[228] Philip Lord and Alison MacDonald. Data curation for e-Science in the UK: an audit to establish requirements for future curation and provision. JISC e-Science Curation Report, November 2003. http://www.jisc.ac.uk/publications/documents/esciencefinalreport.aspx.

[229] Norbert Lossau and Dale Peters. DRIVER: Building a Sustainable Infrastructure of European Scientific Repositories. *Liber Quarterly*, 18(3/4):437–448, 2008.

[230] Christoph Ludwig and Thorsten Vitt. TextGrid Tutorial. Report from the TextGrid Summit, January 2009. http://www.textgrid.de/berichte.html.

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

[232] Trevor Lui. Towards a knowledge-based economy. *InsideKnowledge Magazine*, 5(6), March 2002. http://tinyurl.com/lkdwrx.

[233] Clifford Lynch. Institutional Repositories: Essential Infrastructure for Scholarship in the Digital Age. *ARL Bimonthly Report*, 226, February 2003.

[234] Clifford A. Lynch. The Transformation of Scholarly Communication and the Role of the Library in the Age of Networked Information. *The Serials Librarian*, 23(3):5–20, 1993.

[235] 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. In *Proceedings of the UK e-Science All Hands Conference*, pages 711–719. Engineering and Physical Sciences Research Council, 2004.

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

[237] Nancy L. Maron and K. Kirby Smith. Current Models of Digital Scholarly Communication. Results of an Investigation Conducted by Ithaka for the Association of Research Libraries, November 2008.
http://www.arl.org/bm~doc/current-models-report.pdf.

[238] Jerome Mcdonough. METS: standardized encoding for digital library objects. *International Journal on Digital Libraries*, (2):148–158, April.

[239] Neil McLean and Clifford Lynch. Interoperability between library information services and learning environments – bridging the gaps. A joint white paper on behalf of the IMS Global learning consortium and the Coalition for Networked Information, May 2004. http://www.imsglobal.org/digitalrepositories/CNIandIMS_2004.pdf.

[240] Wolfgang Meier. eXist: An Open Source Native XML Database. In *Web, Web-Services, and Database Systems*, volume 2593 of *LNCS*, pages 169–183. Springer-Verlag, 2003.

[241] Kalman Z. Meth and Julian Satran. Design of the iSCSI protocol. pages 116–122, 2003.

[242] Microsoft. Support Lifecycle. Website, Viewed September 2009. http://support.microsoft.com/?LN=en-us&x=9&y=2&scid=gp%3B[ln]%3Blifecycle.

[243] Reagan W. Moore. Building Preservation Environments with Data Grid Technology. *American Archivist*, 69(1):139–158, July 2006.

[244] Alan Morrison. Delivering Electronic Texts Over the Web: The Current and Planned Practices of the Oxford Text Archive. *Computers and the Humanities*, 33(1-2), April 1999.

[245] James D. Myers, Joe Futrelle, Jeff Gaynor, Joel Plutchak, Peter Bajcsy, Jason Kastner, Kailash Kotwani, Jong Sung Lee, Luigi Marini, Rob Kooper, Robert E. McGrath, Terry McLaren, Alejandro Rodríguez, and Yong Liu. Embedding Data within Knowledge Spaces. *CoRR*, 2009.

[246] Matt Nelson. In Search of a Digital Standard. *M@R*, 2006. http://www.marketingatretailnews.com/article/8884.aspx.

[247] Michael L. Nelson. Data-Driven Science: A New Paradigm? *EDUCAUSE Review*, 44(4), July/August 2009.

[248] nestor Arbeitsgruppe Grid/e-Science und Langzeitarchivierung. Digitale Forschungsdaten bewahren und nutzen – für die Wissenschaft und die Zukunft. nestor Bericht (in German), 2009.

[249] Heike Neuroth, Martina Kerzel, and Wolfgang Gentzsch, editors. *German Grid Initiative D-Grid*. Universitätsverlag Göttingen, September 2007.

[250] Andrew Newman, Stephen Jefferies, Ron Chernich, and Jane Hunter. *A Semantic Search Engine for SRB*. dart, Storage and Infrastructure Work Package 3, May 2007. http://www.dart.edu.au/workpackages/si/si3-finalreport.pdf.

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

[252] National Science Foundation (NSF). High Performance Computing System Acquisition: Towards a Petascale Computing Environment for Science and Engineering. Funding Call November 2008.
http://www.nsf.gov/pubs/2008/nsf08573/nsf08573.htm.

[253] National Science Foundation (NSF). Sustainable Digital Data Preservation and Access Network Partners (DataNet). Funding Call August 2007.
http://www.nsf.gov/funding/pgm_summ.jsp?pims_id=503141.

[254] OECD. Principles and Guidelines for Access to Research Data from Public Funding, April 2007.
http://www.oecd.org/document/55/0,3343,en_2649_34293_38500791_1_1_1_1,00.html.

[255] Task Force on Archiving of Digital Information. *Preserving digital information*. commissioned by the Commission on Preservation and Access and the Research Libraries Group, May 1996.

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

[257] Budapest Open Access Initiative OSI. OSI Guide to Institutional Repository Software, August 2004.

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

[259] Mayur R. Palankar, Adriana Iamnitchi, Matei Ripeanu, and Simson Garfinkel. Amazon S3 for Science Grids: a Viable Solution? In *DADC '08: Proceedings of the 2008 international workshop on Data-aware distributed computing*, pages 55–64, New York, NY, USA, 2008. ACM.

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

[261] Kathrin Peter. Einstiegsmaterial iRODS - integrated Rule-Oriented Data System. Report, D-Grid Integrationsprojekt 2 (DGI-2), December 2008. `http://dgi.d-grid.de/fileadmin/user_upload/documents/DGI2-FG4/FG4-4-irods/m12_iRODSEinstieg.pdf`.

[262] Macario Polo, Mario Piattini, and Francisco Ruiz. Reflective Persistence – Reflective CRUD: Reflective Create, Read, Update and Delete. `http://hillside.net/europlop/HillsideEurope/Papers/ReflectivePersistence.pdf`, 2001.

[263] Andy Powell. JISC Information Environment Architecture, 2005. `http://www.ukoln.ac.uk/distributed-systems/jisc-ie/arch/`.

[264] Andy Powell, Mikael Nilsson, Ambjörn Naeve, Pete Johnston, and Thomas Baker. DCMI Abstract Model. Technical report, June 2007. `http://dublincore.org/documents/abstract-model/`.

[265] The Apache Cocoon Project. *How to Publish XML Documents in HTML and PDF*.

[266] Arcot Rajasekar, Mike Wan, Reagan Moore, and Wayne Schroeder. Data Grid Federation. In *PDPTA 2004 - Special Session on New Trends in Distributed Data Access*, June 2004.

[267] Arcot Rajasekar, Mike Wan, Reagan Moore, and Wayne Schroeder. A Prototype Rule-based Distributed Data Management System. In *Proceedings of the HPDC workshop on "Next Generation Distributed Data Management"*, Paris, France, May 2006.

[268] Jose Carlos Ramalho, Miguel Ferreira, Luis Faria, Rui Castro, Francisco Barbedo, and Luis Corujo. RODA and CRiB a service-oriented digital repository. In *Proceedings of the International Conference on Preservation of Digital Objects (iPRES)*, London, 2008.

[269] Carl Rauch, Harald Krottmaier, and Klaus Tochtermann. File-Formats for Preservation: Evaluating the Long-Term Stability of File-Formats. In *Proceedings of the ELPUB2007 Conference on Electronic Publishing*, Vienna, Austria, June 2007.

[270] Eric S. Raymond. *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary.* O'Reilly & Associates, Inc., Sebastopol, CA, USA, 2001. Foreword By-Young, Bob.

[271] Matthias Razum. FIZ Karlsruhe. Personal Communication, 2009.

[272] William Reilly. DSpace and E-Learning. In *Proceedings of the 5th Sakai Conference*, Vancouver, BC, Canada, May 2006.

[273] Peter Reimer and Jochen Schirrwagen. Eine 'Publikationsmaschine'. Die Technik von Digital Peer Publishing NRW. *ProLibris*, 10(1):19–21, 2005.

[274] Leonard Richardson and Sam Ruby. *RESTful web services.* O'Reilly, Farnham, 2007.

[275] Arnon Rotem-Gal-Oz. Defining SOA. techweb, Dr.Dobb's Blog, November 2007. `http://www.ddj.com/blog/architectblog/archives/2007/11/defining_soa_pa.html`.

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

[277] Najla Semple. Digital Repositories. DCC Report, April 2006. `http://www.dcc.ac.uk/resource/briefing-papers/digital-repositories/`.

[278] Peter M. Senge. *The Fifth Discipline: The Art + Practice of the Learning Organization.* Doubleday Business, October 1994.

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

[280] David Slik. Bycast's Cloud Storage HTTP API. Presented at the SNIA Cloud Storage Group Meeting, May 2009.
http://groups.google.com/group/snia-cloud/web/cloud-storage-twg-chicago-2009.

[281] MacKenzie Smith, Mary Barton, Mick Bass, Margret Branschofsky, Greg McClellan, Dave Stuve, Robert Tansley, and Julie Harford Walker. DSpace – An Open Source Dynamic Digital Repository. *D-Lib Magazine*, 9(1), 2003.

[282] MacKenzie Smith and Reagan W. Moore. Digital Archive Policies and Trusted Digital Repositories. *International Journal of Digital Curation*, 2(1), June 2007.

[283] Java Community Specification. *JSR 283 - Content Repository for JavaTM technology API*, September 2007. http://jcp.org/en/jsr/detail?id=283.

[284] S. Stein and S. Argamon. A mathematical explanation of Burrows's Delta. In *In Proceedings of the Digital Humanities Conference*, pages 207–209, Paris, France, 2006.

[285] Marcia Stepanek. Data storage: From digits to dust. *Business Week*, April 20 1998. http://www.businessweek.com/archives/1998/b3574124.arc.htm.

[286] SNIA Storage Networking Industry Association. Cloud Storage Initiative (CSI). Standardization Initiative, Viewed August 2009. http://www.snia.org/forums/csi.

[287] Mark W. Storer, Kevin M. Greenan, Ethan L. Miller, and Kaladhar Voruganti. POTSHARDS: Secure Long-Term Storage Without Encryption. In *Proceedings of the 2007 USENIX Annual Technical Conference*, pages 143–156, 2007.

[288] Robert Strandh. Programming methods and techniques – Syntax of programming languages. Online Course Material, Viewed August 2009.
http://www.labri.fr/perso/strandh/Teaching/MTP/Common/Strandh-Tutorial/syntax.html.

[289] Friedrich Summann and Norbert Lossau. Search Engine Technology and Digital Libraries – Moving from Theory to Practice. *D-Lib Magazine*, 10(9), 2004.

[290] Stichting SURF. DARE use of Dublin Core, version 2.0. Report, December 2004. http://www.surffoundation.nl/wiki/download/attachments/2490505/DARE+use+of+DC+v.+2.0.pdf?version=1.

[291] Robert Tansley. Building a Distributed, Standards-based Repository Federation – The China Digital Museum Project. *D-Lib Magazine*, 12(7/8), July/August 2006.

[292] Robert Tansley and Christopher Gutteridge. Eprints Archive Software. System Documentation, Viewed August 2009. http://www.eprints.org/files/eprints1/docs/eprints-system.html.

[293] D. Tarrant and S. Hitchcock. The KeepIt Project - Kultur, eCrystals, EdShare (and NECTAR) - Preserve It! In *The 4th annual international Open Repositories Conference (or09) - EPrints User Group*, Atlanta, Georgia, May 18th - 21st 2009.

[294] David Tarrant, Tim Brody, and Leslie Carr. From the Desktop to the Cloud: Leveraging Hybrid Storage Architectures in your Repository. In *Proceedings of the 4th annual international Open Repositories Conference (or09)*, Atlanta, Georgia, May 2009.

[295] UKOLN Repositories Team. Repositories Research Team Wiki (DigiRep), Viewed August 2009. http://www.ukoln.ac.uk/repositories/digirep/index/Repositories_Research.

[296] TextGrid. Scenarios. TextGrid Report, December 2006. http://www.textgrid.de/berichte.html.

[297] TextGrid. TextGrid Komponenten. TextGrid Report, December 2006. http://www.textgrid.de/berichte.html.

[298] TextGrid. Text Retrieval. TextGrid Report 1.3, May 2007. http://www.textgrid.de/berichte.html.

[299] TextGrid. TextGrid Architecture. TextGrid Report 3.2, January 2007. http://www.textgrid.de/berichte.html.

[300] TextGrid. TextGrid Manual: Tool Development. TextGrid Report 3.5, February 2008. http://www.textgrid.de/berichte.html.

[301] TextGrid. Baseline Encoding, Metadata Scheme, and Metadata Management. TextGrid Report, January 2009. http://www.textgrid.de/berichte.html.

[302] TextGrid. Installation eines Datengrid-Knotens. TextGrid Report 3.6, March 2009. http://www.textgrid.de/berichte.html.

[303] Nannette Thacker. 3-Tier Web Application Development. Blog Post, 2008. http://weblogs.asp.net/nannettethacker/archive/2008/03/05/3-tier-web-application-development.aspx.

[304] 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.

[305] Barbara B. Tillett. A Virtual International Authority File. In *Workshop on Authority Control among Chinese, Korean and Japanese Languages (CJK Authority 3)*, Karuizawa, Tokyo, Kyoto, March 14-18 2002.

[306] Andrew Treloar. ARROW Targets: Institutional Repositories, Open-Source, and Web Services. In *Proceedings of AusWeb05, the Eleventh Australian World Wide Web Conference*. Southern Cross University, Southern Cross University Press, July 2005.

[307] Andrew Treloar and Cathrine Harboe-Ree. Data management and the curation continuum: how the Monash experience is informing repository relationships. In *Proceedings of the VALA Conference 2008*, Melbourne, Australia, February 2008.

[308] Ulf Troppens. Speicher im Netz – Professionelle Speicherverwaltung mit SAN und NAS. *c't*, August 2003.

[309] Herbert Van de Sompel, Jeroen Bekaert, Xiaoming Liu, Lyudmila Balakireva, and Thorsten Schwander. aDORe: a modular, standards-based Digital Object Repository. *The Computer Journal*, 48(5), 2005.

[310] Herbert Van de Sompel, Michael L. Nelson, Carl Lagoze, and Simeon Warner. Resource Harvesting within the OAI-PMH Framework. *D-Lib Magazine*, 10(12), December 2004.

[311] Ubbo Veentjer. Bestimmung relevanter Worte eines Textes und Darstellung unter der Oberfläche TextGrid-Workbench. Master's thesis, Georg-August-Universität Göttingen, April 2009.
http://www.informatik.uni-goettingen.de/studies/courses/theses.htm?&show=year&year=2009.

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

[313] Srikumar Venugopal, Rajkumar Buyya, and Kotagiri Ramamohanarao. A taxonomy of Data Grids for distributed data sharing, management, and processing. *ACM Comput. Surv.*, 38(1):3, 2006.

[314] Oliver Vogel, Ingo Arnold, Arif Chughtai, and Markus Völter. *Software-Architektur. Grundlagen - Konzepte - Praxis.* Spektrum Akademischer Verlag, January 2005.

[315] Patricia Ward and George A Dafoulas. *Database Management Systems.* Int. Thomson Business Press, 2006.

[316] Paul Watry. Digital Preservation Theory and Application: Transcontinental Persistent Archives Testbed Activity. *The International Journal of Digital Curation*, 2(2), November 2007.

[317] Colin Webb. Guidelines for the Preservation of Digital Heritage. UNESCO Report, United Nations Educational, Scientific and Cultural Organization, Paris, March 2003. http://unesdoc.unesco.org/images/0013/001300/130071e.pdf.

[318] Andrea Weise. Handling Small Files in iRODS. In *Proceedings of the iRODS Week*, Edinburgh, UK, May 27 2008.

[319] John Wetherill. Messaging Systems and the Java Message Service (JMS). SUN Developer Network, Viewed August 2009.
http://java.sun.com/developer/technicalArticles/Networking/messaging/.

[320] Brad Wheeler. E-Research is a Fad: Scholarship 2.0, Cyberinfrastructure, and IT Governance. In Richard N. Katz, editor, *The Tower and The Cloud – Higher Education in the Age of Cloud Computing*, pages 108–117. Educause, 2008.

[321] Michael Witt and Jigar Kadakia. *OAISRB*. Purdue University, 2006. http://www.lib.purdue.edu/research/oaisrb.

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

[323] 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(10), October 2005.

[324] Ronald Yanosky. From Users to Choosers: The Cloud and the Changing Shape of Enterprise Authority. In Richard N. Katz, editor, *The Tower and The Cloud – Higher Education in the Age of Cloud Computing*, pages 108–117. Educause, 2008.

[325] Joseph W. Yoder, Ralph E. Johnson, and Quince D. Wilson. Connecting Business Objects to Relational Databases, 1998.

[326] Bing Zhu, Richard Marciano, and Reagan Moore. Enabling Inter-repository Access Management between iRODS and Fedora. In *Proceedings of the 4th International Conference on OpenRepositories*, Atlanta, Georgia, USA, May 2009.

[327] Hubert Zimmermann. OSI Reference Model – The ISO Model of Architecture for Open Systems Interconnection. *Communications, IEEE Transactions on*, 28(4):425–432, April 1980.

Curriculum Vitae of
Andreas Aschenbrenner

M.Sc., B.SocEcSc.

nationality: Austrian
address: Goettingen, Germany

language skills

German mother-tongue
English fluent
Dutch, French, Latin, and **Swedish** basic

employment

2006 - **State and University Library** Goettingen, DE
TextGrid (www.textgrid.info), Interedition, DARIAH, D-Grid:
technical architect - e-Humanities, digital repositories, grid technologies

2005 - 2006 **Austrian Research Centers** Vienna, AT
project developer and research -
online collaboration, context mining, music information retrieval

2004 - 2005 **reUSE!** Vienna, AT
reUSE (reuse.uibk.ac.at): eContent Programme, No. 11173
(contracted at the Austrian National Library)
analyst - trusted digital repositories

2002 - 2004 **ERPANET** The Hague, NL
ERPANET (www.erpanet.org): IST-2001-32706
analyst, content editor, workshop organisation - digital preservation

2003 **LEAF**
LEAF (www.leaf-eu.org): IST-2000-26323
(contracted at the Austrian National Library)
technical consultant - library systems, metadata, standardisation

education

2004 - 2006 **technical university** Vienna, AT
studies in **computer science management**, with **honours**

1997 - 2002 **technical university** Vienna, AT
studies in **computer science**, graduated with **honours**
exchange studies in Stockholm, Sweden

1989 - 1997 **school** (humanistic) Vienna, AT

Thesis Publications

scientific activities in the context of this thesis

- e-IRG DMTF (2008-) - e-Infrastructure Reflection Group, Data Mngmnt Task Force
- OGF Digital Repositories (2008-) - Chair of the Digital Repositories Research Group at the Open Grid Forum (OGF)
- DReSNet (2008-) - member of the Digital Repositories e-Science Network
- DARIAH (2008-2010) - a European grid-based repository infrastructure for the humanities
- WissGrid (2009-2011) - establishing a repository network for research data in Germany (D-Grid)
- D-Grid, FG4 Data (2008-2010) - metadata and repositories in grid environments
- TextGrid (2006-2012) - a grid-based repository for the humanities (Germany)
- reuse! (2004-2005) - evaluation of trusted digital repositories
- ERPANET (2002-2004) - digital preservation
- AOLA (2000-2001) - construction of national web repository - Austrian On-Line Archive

workshops organised in context of thesis

- 2 December 2009, REPRISE Workshop on Repository/Preservation Environments, Digital Curation Conference, London
- 27 May 2009, Repositories RG, Open Grid Forum (OGF) 26, Chapel Hill
- 12 December 2008, IEEE e-Science 2008, Indianapolis
- 1 December 2008, Digital Curation Conference (DCC) 2008, Edinburgh
- 5 June 2008, Open Grid Forum (OGF) 23, Barcelona

publications: peer-reviewed papers and journal articles

- Andreas Aschenbrenner, Tobias Blanke, Eric Haswell, and Mark Hedges. **The DARIAH e-Infrastructure**. Zero-In Magazin, 3, October 2009.

- Andreas Aschenbrenner, Flavia Donno, Senka Drobac: **Infrastructure for Interactivity -- Decoupled Systems on the Loose**. In: Proceedings of the IEEE Digital Ecosystems and Technologies (DEST) 2009, Istanbul, Turkey. 1-3 June 2009.

- Andreas Aschenbrenner, Marc Wilhelm Küster, Christoph Ludwig, Thorsten Vitt: **Open eHumanities Digital Ecosystems and the Role of Resource Registries**. In: Proceedings of the IEEE Digital Ecosystems and Technologies Conference (DEST)

2009, Istanbul, Turkey. 1-3 June 2009.

- Andreas Aschenbrenner, Tobias Blanke, Neil P. Chue Hong, Nicholas Ferguson, Mark Hedges: **A Workshop Series for Grid/Repository Integration.** In: D-Lib Magazine, January/February 2009.

- Marc Wilhelm Küster, Christoph Ludwig, Andreas Aschenbrenner: **TextGrid: eScholarship und vernetzte Angebote**. In: Claudine Moulin, Thomas Burch und Andrea Rapp (eds.): *it - Information Technology*. Themenheft "Informatik in den Philologien", 2009. Jahrgang 51 (2009) Heft 4, S. 183-190.

- Tobias Blanke, Andreas Aschenbrenner, Marc Küster, Christoph Ludwig: **No Claims for Universal Solutions - Possible Lessons from Current e-Humanities Practices in Germany and the UK**. In: e-Humanities - An Emerging Discipline. Workshop at the 4th IEEE International Conference on e-Science. December 2008.

- Andreas Aschenbrenner, Tobias Blanke, David Flanders, Mark Hedges, Ben O'Steen: **The Future of Repositories? - Patterns for (Cross-)Repository Architectures.** In: D-Lib Magazine, November/December 2008.

- Andreas Aschenbrenner, Tobias Blanke, Mark Hedges: **Synergies between Grid and Repository Technologies - A Methodical Mapping.** In: IEEE International Conference on e-Science, pp. 778-781, 2008 Fourth IEEE International Conference on eScience, 2008.

- Andreas Aschenbrenner, Malte Dreyer: **(digital library architecture)^2 - service patterns for large-scale digital libraries** In: Paolo Manghi, Pasquale Pagano, and Pavel Zezula: Proceedings of the First DELOS Workshop on Very Large Digital Libraries (VLDL), held in conjunction with ECDL 2008. Aarhus, Denmark.

- Andreas Aschenbrenner, Katja Meffert: **Wissenschaftliche Infrastruktur in den Geisteswissenschaften? - Eine Wegbeschreibung**. In: Jahrbuch der Computerphilologie, August 2008.

- Andreas Aschenbrenner, Tobias Blanke, Mark Hedges, Frank Schwichtenberg: **(repository +/= e-Infrastructure) ?!** In: Third International Conference on OpenRepositories 2008, 1-4 April 2008, Southampton, United Kingdom.

- Andreas Aschenbrenner: **Editing, analyzing, annotating, publishing: TextGrid takes the a, b, c to D-Grid**. In: iSGTW 30 January 2008, Jg. 54.

- Andreas Aschenbrenner, Thomas Wollschläger: **File Format Registries**, and **Institutional Repositories**. In: Heike Neuroth et al. (eds.): nestor Handbuch: Eine kleine Enzyklopädie der digitalen Langzeitarchivierung. 2007.

- Andreas Aschenbrenner, Tobias Blanke, Stuart Dunn, Martina Kerzel, Andrea Rapp, Andrea Zielinski: **Von e-Science zu e-Humanities - Digital vernetzte Wissenschaft als neuer Arbeits- und Kreativbereich für Kunst und Kultur**. In: Bibliothek. Forschung und Praxis, Heft 1, 2007. S.11-21.

- Marc Wilhelm Küster, Christoph Ludwig und Andreas Aschenbrenner: **TextGrid as a Digital Ecosystem**. Proceedings of IEEE Digital Ecosystems and Technologies Conference (DEST) 2007, Cairns, Australien.

- Heike Neuroth, Andreas Aschenbrenner, Felix Lohmeier: **e-Humanities - eine virtuelle Forschungsumgebung für die Geistes -, Kultur- und Sozialwissenschaften**. In: Bibliothek. Forschung und Praxis (K.G. Saur Verlag). 31.2007, Nr. 3, S. 272-279.

- Andreas Aschenbrenner, Peter Gietz, Marc Wilhelm Küster, Christoph Ludwig and Heike Neuroth: **TextGrid - a modular platform for collaborative textual editing**. In: Digital Library Goes e-Science (DLSci06), European Conference on Digital Libraries 2006, September 21, 2006, Alicante, Spain. S. 27-36.

- Peter Gietz, Andreas Aschenbrenner, et al: **TextGrid and eHumanities**. Proceedings of the Second IEEE International Conference on e-Science, December 2006, Amsterdam.

- Andreas Aschenbrenner, Andreas Rauber: **Mining Web Collections**. In: Julien Masanes (eds.): "Web Archiving", Springer, 2006, ISBN: 978-3-540-23338-1, S. 153-176.

- Andreas Aschenbrenner, Olaf Brandt, Stephan Strodl: **Report on the 5th International Web Archiving Workshop (IWAW)**. D-Lib Magazine, v.11, n.11 (November 2005).

- Andreas Aschenbrenner, Maria Baumgartner, Christine Böhm, Erich Gstrein, Oliver Holle, Karl Maier, Silvia Miksch, Bruno Schernhammer: **Fostering social relations in a distributed environment - WAFF's Intralife.** In Proceedings of the 9th European Conference on Computer-Supported Cooperative Work (ECSCW), 18-22 September 2005, Paris, France.

- Andreas Aschenbrenner, Max Kaiser: **White Paper on Digital Repositories**. Deliverable of the European project reUSE. March 2005.

- Andreas Aschenbrenner: **a methodology for metadata modelling - depth for a flat world**. International Conference on Dublin Core and Metadata Applications, DC-2004. Shanghai, China, 11-14 October 2004.

- Andreas Aschenbrenner: **The Bits and Bites of Data Formats - Stainless Design for Digital Endurance**. In: RLG DigiNews (ISSN 1093-5371), v8, n1; February 2004.

- Andreas Aschenbrenner, Andreas Rauber: **Die Bewahrung unserer Online-Kultur. Vorschläge zu Strategien der Webarchivierung** (26.02.2003). In: Sichtungen.

- Andreas Rauber, Andreas Aschenbrenner, Robert M. Bruckner, Oliver Witvoet, Max Kaiser. **Uncovering Information Hidden in Web Archives: A Glimpse at Web Analysis Building on Data Warehouses**. D-Lib Magazine, v.9, n.12 (December 2002).

- Andreas Rauber, Andreas Aschenbrenner, Oliver Witvoet. **Austrian On-Line Archive Processing: Analyzing Archives of the World Wide Web**. In: Proceedings of the 6th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2002); September 16-18, 2002; Rome, Italy.

- Andreas Aschenbrenner. **Long-Term Preservation of Digital Material - Building an Archive to Preserve Digital Cultural Heritage from the Internet**. Master Thesis, Technical University Vienna, December 2001.

- Andreas Rauber, Andreas Aschenbrenner. **Part of Our Culture is Born Digital - On Efforts to Preserve it for Future Generations**; appeared in TRANS. Internet-Zeitschrift für Kulturwissenschaften. No. 10/2001,
  available as PDF and in HTML (English version)

invited presentations (selection)

- Andreas Aschenbrenner: **e-Infrastructure for Research Repositories**. Invited Talk at the First OGF Repository Workshop, Open Grid Forum, June 5, 2008, Barcelona.

- Andreas Aschenbrenner: **DARIAH - e-Humanities in Europe**. At: Workshop on eArts and eHumanities, Open Grid Forum 20, May 2007, Manchester.

- Andreas Aschenbrenner: **TextGrid - modular platform for collaborative text editing - a community grid for the humanities**. GridKa Summer School, 11.-15.9. 2006, Karlsruhe.

- Andreas Aschenbrenner, Stefan Farrenkopf: **Shibboleth, der Generalschlüssel im Internet**. Präsentation im Weiterbildungsangebot der SUB Göttingen, 6. Juli 2006.

- Andreas Aschenbrenner: **Ziel : vertrauenswürdiges Archiv - ein methodischer Vergleich der reUSE Demonstratoren**. Invited Presentation at the nestor workshop Vertrauenswürdige digitale Langzeitarchive: Kriterien und deren Bewertung, 21. Juni 2005 in Munich (Germany).

- Andreas Aschenbrenner: **File Format Features and Significant Properties** (PPT), and: **File Format Registries** (PPT). Invited Presentation at the Chinese-European Workshop on Digital Preservation. BeiJing; July 14-16, 2004.

- Andreas Aschenbrenner: **Mining Web archives**. Presentation at the Aristote Seminar "Internet : la mémoire courte ?" at the Bibliothèque Nationale de France. Paris; April 22, 2004.

service to the scientific community

- (reviewer) IEEE e-Science. Oxford, December 2009.
- (reviewer) UK All-Hands meeting 2009, Data management Track. Oxford, Dec 2009.
- (organising committee) Repositories RG, Open Grid Forum (OGF) 26, Chapel Hill, 27 May 2009.

- (organising committee) IEEE e-Science 2008, Indianapolis. 12 December 2008.
- (organising committee) Digital Curation Conference (DCC) 2008, Edinburgh. 1 December 2008.
- (organising committee) Open Grid Forum (OGF) 23, Barcelona. 5 June 2008
- (organising committee) Fedora European User Group. Oxford, December 2009.
- (organising committee) REPRISE Workshop on Repository/Preservation Environments, Digital Curation Conference, London, December 2009
- (program committee) IEEE DEST, e-Humanities and IEEE e-Science, e-Humanities (2009)
- (organiser) Workshop Series for Grid/Repository Integration. (2008-2009)
- (program committee) DORSDL2 - Digital Object Repository Systems in Digital Libraries. Workshop at ECDL 2008. Aarhus, Denmark.
- (organising committee) Special Session on Digital Repositories. IEEE e-Science 2008. December 7, 2008. Indianapolis.
- (reviewer) New Review on Hypermedia and Multimedia (JNRHM), Special Issue on Web Archiving, 2007.
- (organisation) Göttinger Grid Seminar 2006-2007, Paulinerkirche Göttingen.
- (reviewer) International Web Archiving Workshop (IWAW), European Conference on Digital Libraries (ECDL). 2003-2008.
- (organisational assistance) iPRES - International Conference on Digital Preservation. 2004-2006.
- (organisation) ERPANET preservation workshops, 2002-2004.

teaching

- (Co-Betreuer) WS 2008/09: **Grid Technologie in der Wissenschaft - Konzepte, Methoden und Anwendungen**(Med, SWE)
- (Co-Betreuer) SoSe 2008: **Interdisziplinäres Praktikum zur Anwendung von Grid-Technologie** (UniVZ, SWE)
- (Co-Betreuer) WS 2007/08: **Grid Technologie in der Wissenschaft - Konzepte, Methoden und Anwendungen** (Med, SWE)
- (Tutor) Tutor für verschiedene Lehrveranstaltungen an der Technischen Universität Wien (1999-2001), darunter: **Einführung in das Programmieren, Logische Programmierung, Software Engineering, Distributed Systems**