The Planets Interoperability Framework
An Infrastructure for Digital Preservation Actions

Ross King[1], Rainer Schmidt[1],
Andrew N. Jackson[2], Carl Wilson[2], and Fabian Steeg[3]

[1] Austrian Research Centers GmbH - ARC
Donau-City-Strasse 1
1220 Vienna, Austria
firstname.lastname@arcs.ac.at
[2] The British Library
Boston Spa, Wetherby, West Yorkshire
LS23 7BQ, United Kingdom
firstname.lastname@bl.uk
[3] Universität zu Köln
Albertus-Magnus-Platz
50923 Cologne, Germany
fabian.steeg@uni-koeln.de

Abstract. We report on the implementation of a software infrastructure for preservation actions, carried out in the context of the European Integrated Project Planets – the Planets *Interoperability Framework* (IF). The design of the framework was driven by the requirements of logical preservation in the domain of libraries and archives, which include durable and scalable infrastructures for the characterisation and migration of digital documents, and emulation of digital environments. The IF is a Java-based software suite built on a number of open source components and Java standards such as Java Platform, Enterprise Edition (Java EE 5) and JSR-170 (Java Content Repository API). Specific features of interest include a web service architecture including specified preservation service interfaces for the integration of new and existing preservation tools, a workflow engine for the execution of flexible preservation plans, and a Job Submission Service (JSS) for managing computationally intensive preservation actions on millions of digital objects.

1 Introduction

This paper describes a software infrastructure developed in the context of the EU Integrated Project Planets known as the Planets Interoperability Framework (IF). In general, the Planets project concentrates on the questions of logical preservation (as opposed to bit-stream preservation), providing tools for the characterization and migration of digital objects as well as emulation tools for accessing original digital objects. A more detailed overview of the Planets project can be found elsewhere [1].

One of the visions of the Planets project is to provide a majority of project results in a single downloadable package, which should be simple to install, configure, and administer - the *Planets Software Suite*. When this package is deployed, establishing a *Planets Instance*, an administrator should be able to create user accounts, deploy and browse services, and administer data registries. A preservation expert should be able to define preservation workflows, define and evaluate preservation plans (Preservation Planning Application), as well as define and run preservation experiments. A librarian or archivist should be able to define and test preservation plans and execute preservation workflows on a repository.

The Planets IF supports this vision by providing the technical backbone for registries, services, and applications developed within the project, as well as standard software components such as authentication and authorization, web service provisioning, and data persistence.

This paper is organized as follows: we first present a short overview of other software infrastructures in the digital preservation domain. We then give a detailed description of the Interoperability Framework's design, components, and implementation details, followed by a specific discussion of our approach to achieving performance and scalability. We conclude the paper with an outlook on the testing and evaluation work to be carried out with the Interoperability Framework during the final year of the Planets project.

2 State of the Art

Digital preservation, and in particular the aspect of logical preservation, became a significant topic within the European Union's ICT 6th Framework Programme. There the DELOS Network of Excellence Preservation Cluster continued work that was pioneered within ERPANET. Results from this project provided for example methodologies for evaluating and testing preservation plans [2], laying the groundwork for preservation projects that followed like Planets and CASPAR[4]. These were followed by a new generation of projects funded under the ICT 7th Framework Programme, including PrestoPRIME[5], KEEP, PROTAGE[6], and SHAMAN[7].

In contrast to the content-agnostic approach of Planets, which is driven by the needs of national libraries and archives, PrestoPRIME will address long-term preservation of and access to digital audio-visual content by integrating media archives with European on-line digital libraries. The Keeping Emulation Environments Portable (KEEP) project extends work on emulation in Planets and will create portable emulators enabling access to and use of software that requires outdated computer hardware and the digital content that relies on it.

[4] http://www.casparpreserves.eu/
[5] http://wiki.prestospace.org/pmwiki.php?n=Main.PrestoPRIME
[6] http://www.protage.eu/
[7] http://shaman-ip.eu/

The PROTAGE (PReservation Organizations using Tools in AGent Environments) project plans to build and validate a system based on software agents that can cooperate with and be integrated in existing and new preservation systems. SHAMAN - Sustaining Heritage Access through Multivalent ArchiviNg - aims at providing a comprehensive long-term preservation system, also considering Grid and virtualisation techniques for scalability as in Planets (refer to the subsection on *Performance and Scalability* below).

One cannot address preservation architectures without mentioning the Open Archival Information System standard [3]. Significant effort in the preservation community has been dedicated towards producing OAIS-compliant archiving infrastructures, although OAIS itself is primarily a conceptual model and does not provide guidelines for implementation. The Planets project has concentrated on the specific OAIS component "Preservation Planning". The CASPAR project [4] places an emphasis on representation information.

One component of the overall CASPAR architecture that is comparable to the Planets Interoperability Framework is the Preservation Data Store (PDS) [5]. The PDS approach delegates preservation-related functionality to the storage component, under the hypothesis that this will increase the robustness of the digital preservation system and its ability to protect against corruption or loss. Similar to Planets, the PDS architecture supports integration with existing archives. However, PDS lacks the flexibility of the Planets IF plug-in approach to preservation tools and workflows.

One of the earliest OAIS-based preservation systems is the *e*-Depot [6] of the National Library of the Netherlands, based on IBM's Digital Information Archiving System (DIAS) [7] infrastructure. The DIAS system provides for automatic ingest (creation of AIPs from SIPs in the OAIS model) of digital objects from multiple sources. DIAS also makes use of the concept of the *View Path* as a method for both archiving representation information for objects and for identifying digital obsolescense. However, DIAS does not provide for the execution of preservation actions within the archive; the Planets IF would be a sensible extension of DIAS-based systems by providing this functionality.

Another project based on the OAIS reference model is the National Geospatial Digital Archive [8], part of the U.S. Library of Congress' National Digital Information Infrastructure and Preservation Program (NDIIPP). The system provides several web-based tools, including a proprietary collaborative format registry and workflow tool for managing the ingest process. Because the system is concerned with data rather than documents, preservation actions like characterisation, migration, and emulation are ignored in favor of *normalisation* – the conversion of the source object from its original data format into an open, preservation-friendly archival format.

The National Archives of Australia have defined a preservation approach [9] which also strongly builds on normalisation. Their work has resulted in the open source tool XENA[8], which can convert a large number of digital input formats to an XML-based archival format. In other words, XENA is a specific instance

[8] `http://xena.sourceforge.net/`

of a preservation migration tool, and as such, it has already been "wrapped" as a Planets Service (refer to the subsection on *Planets Service Interfaces* below) and integrated with the IF infrastructure.

The closest parallels to the Planets approach are the Australian project PANIC (Preservation services Architecture for New media and Interactive Collections) [10] and The United Kingdom's National Archives Seamless Flow framework [11]. Both are based on service-oriented architectures (SOA) in which preservation actions are invoked through Web Services. These actions are semi-automatically invoked based on comparing the preservation metadata of digital objects against technical registries; in the case of Seamless Flow, this is based on the PRONOM[9] registry, whereas PANIC has an open interface allowing integration with various registries, such as Harvard University Library's Global Digital Format Registry[10]. PANIC also distinguishes itself in making use of Semantic Web Services, using an OWL-based ontology for matching services with preservation actions. The Planets Interoperability Framework is flexible enough to support applications of this nature, but is presently focused on supporting the OAIS approach to preservation planning, by executing institutional *Preservation Plans*, which are the output of tools such as PLATO [12].

3 The Planets Interoperability Framework

3.1 Motivation

A preservation system has the following three objectives [6]

- Identifying objects that are in danger of digital obsolescence.
- Planning and carrying out technical preservation actions.
- Defining the necessary technical metadata in order to provide an environment needed for digital object delivery and access.

The Interoperability Framework fulfills the second objective by providing an infrastructure to carry out digital preservation actions in the form of flexible, service-based workflows. Planets is driven by the requirements of memory institutions, primarily national libraries and archives. These institutions generally already have archiving systems in place, which are often custom solutions or based on commercial tools. Replacing such systems is neither feasible nor desirable. Therefore the IF was designed to run in parallel with existing archive systems; it is in no way meant to replace these or even to provide archiving functionality.

The Interoperability Framework architecture has also evolved to support two conflicting requirements. On one hand, the system should accommodate preservation (potentially commercial) services that are hosted by third-parties – hence a web service architecture is ideal. On the other hand, the system should also

[9] `http://www.nationalarchives.gov.uk/pronom/`
[10] `http://www.gdfr.info/`

support large-scale processes that involve millions of digital objects and therefore must be local due to network bandwidth restraints.

In addition, there are a number of functions that multi-tier applications commonly need. These include data persistence, user management, authentication and authorization, monitoring, notification, and logging. There are also are some non-functional requirements on the infrastructure, which should be robust, scalable, and distributed. The Planets Interoperability Framework software infrastructure was designed to provide these commonly required functions and meet these non-functional requirements. For preservation applications, such as the PLATO preservation planning tool [12] or the Planets Testbed Application [13], the IF provides for common requirements on the Service Registry, Workflow Execution Engine, and Planets Service Interfaces, which are described in detail below.

There are two primary advantages to the IF approach: efficiency and interoperability. Because the above mentioned components are only developed once, the other Planets components and applications have been able to concentrate on their specific process logic. Also, when packaging the Planets software, the number of components will be optimized; for example, because the IF provides a single relational database management system for all components, only one such system need be installed. Finally, by providing common components, the IF can also help to assure that various applications remain interoperable. By enforcing Web Service standards, the IF can support access to remote and distributed third-party preservation action services.

3.2 IF Components

In the following sub-sections we describe a selection of IF components and their functionality. Refer to the *Business Layer* of figure 1.

Data Registry The Data Registry provides storage and persistence services to IF users, components and services via a consistent API. In particular the API provides methods for storing Digital Objects and Preservation Events. Storage of and access to files and metadata by services and applications is managed through Data Registries.

An IF instance installs a pre-configured Data Registry built upon the Apache Jackrabbit implementation of the Java Content Repository API. Additionally an IF administrator can create additional file system based Data Registries through an administration interface. These file based instances provide convenient, consistent access to local or network accessible disk storage.

Binary data can be stored by value or by reference. Storage by reference is often preferable to avoid the overhead of submitting and accessing large binary objects through Jackrabbit.

The Data Registry controls the IFs shared storage area, i.e. common network storage accessible by all IF components. Each registered user has their own storage area within Jackrabbit in which they can:

– Add content file references and associated metadata.
– Save references to files created by workflow tasks.
– Persist Digital Objects and associated metadata.
– Save metadata generated by workflow tasks.
– Search content and define filesets.

In addition there is a common area allowing users to share their content and the results of preservation actions with other registered users. The IF also provides a web-based Data Registry GUI which allows users to:

– Browse content and metadata in both the common area and their private area.
– Create, delete and modify digital objects in the data registry, depending upon their access rights.
– Import and export sections of their private area to and from the shared area or to other, file system based Data Registries.
– Search content and metadata by XPath/XQuery.

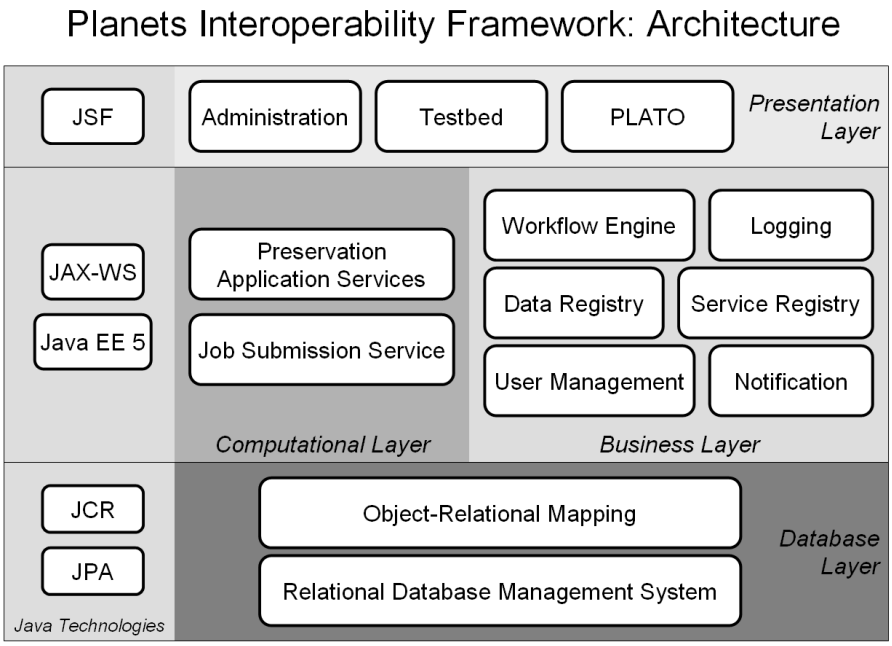

Fig. 1. Planets Interoperability Framework Architecture

Service Registry The Service Registry enables users and service providers to look up and publish information about preservation services, and enables Planets system administrators to manage information about Planets Services. This

information can be used to dynamically select and invoke simple services, as well as to reuse them as part of choreographed complex workflows (refer to the section *Workflow Execution Engine*). In order to support service discovery, the Service Registry contains information about services and the publishers that provide them. Moreover, it provides extensible, schema- or ontology-based service categorization mechanisms that allow the Service Registry to be queried using these categories. Finally, in order to invoke or reuse services, the Planets Service Registry contains information about the service interfaces, operations and parameters.

The Service Registry has two main components: the Service Registry backend which processes requests from clients, and the Service Registry user interface, which is implemented as a Web application using Java Server Faces (JSF) and is integrated within the IF administration user interface. The user interface allows administrators to browse, add, update, and delete items stored in the Service Registry. Planets Services can also be registered automatically in the registry based on the Planets Service Description they provide, either via the user interface or the API.

On the technical level, the IF contains two optional Service Registry backend implementations: a UDDI version 2.0 compliant implementation based on Apache jUDDI that is accessed via the Java API for XML Registries (JAXR, using the Apache Scout implementation), and a lightweight persistence mechanism based on Planets Service Descriptions which provides a simple and flexible query-by-example functionality (e.g. to find services that migrate from PDF to JPG, or services based on the JHOVE[11] tool, etc.). Both Service Registry backend implementations provide a Java API as well as language-independent access via SOAP-based Web Services.

Workflow Execution Engine A preservation workflow consists of a sequence of activities, carried out in a specific order, in which the output parameters of one action are validly mapped to the input parameters of the following action. An example of a Planets workflow would be: for a given file, first identify a file format, then validate the format, then characterize it, then migrate it to a new format, then characterize the new file, then compare with the original. Each of these steps could be carried out by a different component or service in the Planets architecture; hence, the orchestration of these services is required.

We surveyed service orchestration approaches and experimented with WS-BPEL (Web Service Business Process Execution Language). WS-BPEL is an XML-based workflow description language for SOAP-based Web services. The language is an official OASIS standard[12] and has strong industry support. Within the IF, experimental preservation workflows have been implemented using WS-BPEL v2.0 definitions and JBPM BPEL (JBoss Business Process Management module)[13] as a Workflow Execution Engine. This approach provided a large

[11] `http://hul.harvard.edu/jhove/`
[12] `http://www.oasis-open.org/committees/wsbpel/`
[13] `http://docs.jboss.com/jbpm/bpel/v1.1/`

degree of flexibility and allowed the specification of preservation processes at a low (i.e. messaging) level using Web service standard languages like BPEL, XML, XPath, and WSDL. The Eclipse *BPEL Visual Designer*, an IDE plugin adding comprehensive support for the definition and authoring of BPEL processes, served as a graphical interface for visualizing the process flow (e.g. invocations, branching, iterations). However, work in this direction was hindered by two difficulties; first, that the BPEL language is very powerful but also low-level and hence very complex; and second, at the time we conducted the experiments, BPEL related-tools proved to be not yet mature. Both points turned out to be a major hindrance for implementing preservation workflows by non-BPEL experts.

Consequently, we chose to implement a much simplified, custom workflow description language and corresponding execution engine. The main aim of this approach is to shield the user from the complexity of the underlying architecture and implementation issues allowing non-experts (i.e. librarians and archivists) to create and execute preservation workflows. The WEE-Registry allows one to choose from various abstract workflow scenarios (templates). Workflow templates can be easily assembled from existing Java components that may act upon a preservation service or provide utility functions such as metadata manipulation. Using the WEE, selected workflow templates can be dynamically configured and executed based on simple XML descriptions.

The Planets IF workflow engine implements a component-oriented enactor that governs the orchestration of the various preservation components, including functionalities like session-management, communication, and preservation metadata handling. Distributed preservation workflows are conducted from high-level components that abstract the underlying protocol layers. From the user's point of view, the IF workflow execution engine provides high-level interfaces for discovering, configuring, and instantiating preservation workflows. The WEE performs workflow execution asynchronously and may deliver status information to the user based on inquiry and Email-notification capabilities.

Planets Service Interfaces The Planets Interoperability Framework defines a tiered approach to the problem of creating digital preservation services and workflows.

When implementing digital preservation services, developers initially wish to concentrate on low-level concepts and actions. The Planets IF level-one service interfaces define basic digital preservation verbs, for example Identify (format identification), Validate (format validation), and Migrate (format migration). These interfaces perform actions upon single byte sequences without concerning the developer whether the bytes represent an image from a web page or a page from a book. They return results and status as simple structured types. It is possible to develop and deploy level-one services in a variety of environments using a variety of programming languages, and tools.

The interfaces are intended to be lightweight and simple to implement and share a set of common features:

– Operations are atomic.

– Operations have no requirement to store state.
– Operations have no requirement to handle transactions.
– Planets service data types are used for parameters and returns.
– Binary data is handled using a Planets Digital Object instance.

As the workflows a user wishes to implement become more sophisticated, there is a requirement to consider the data management aspect within a repository. Institutions view and model their digital collections in different ways and mapping even simple concepts to an institution's model can be time consuming.

To accommodate these institutional models, the IF supports level two services which implement digital preservation workflows. These higher level services operate upon and decompose institutional data model instances and map these concepts to the simple level-one interfaces. They provide the high-level activities and the necessary control structures required for data model manipulations, metadata mapping, and handling the serialization back to an institution's digital repository.

3.3 Implementation Details

The Interoperability Framework provides a Java-based infrastructure that leverages a number of standards and open source tools. Referring to the *Java Technologies* indicated in figure 1, the core of the IF implementation is the Java Platform, Enterprise Edition (Java EE 5) standard, which among other things provides a framework for the efficient implementation of Web Services and Web applications. It particular we make use of Sun Microsystems' Web Services Interoperability Technology (WSIT) suite and the underlying JAX-WS (Java API for XML Web Services) standard. The IF provides a pre-configured JBoss[14] application server as its default deployment environment. We chose JBoss as the most stable, best-supported open source implementation of the Java EE 5 standard. We also make use of JOSSO[15] (Java Open Single Sign-On), which provides single-sign-in authentication and authorization for all web applications deployed on the IF.

Another significant IF component is the Apache Jackrabbit[16] implementation of the Java Specification Request 170 (JSR-170) specification of the Java Content Repository (JCR) Application Programming Interface (API). The IF Data Registry component is built on top of the JCR API, adding specific functionality for persisting digital objects, as described in the previous section.

The underlying data persistence is provided through the Apache Derby[17] relational database management system (RDBMS). Derby was chosen because its small footprint and pure Java implementation allow it to be easily packaged and installed with minimum user expertise. However, as we use only the standard Java Database Connectivity (JDBC) API and no RDBMS-specific features, it is

[14] `http://www.jboss.org/`
[15] `http://www.josso.org/`
[16] `http://jackrabbit.apache.org/`
[17] `http://db.apache.org/derby/`

possible to configure the IF for operation with other open-source or commercial databases in a production environment. Access to the database layer is provided in many ways: directly, through a JDBC connector, through EJB 3.0, through the Java Persistence API (JPA - as implemented by the the Hibernate object-relational mapping tool), through the JCR API, or through the Data Registry API.

The IF application server supports a variety of web presentation layer technologies, but we have specifically recommended that application developers make use of Java Server Faces (JSF), first of all because it is a standard (presently in version 1.2, based on JSR-252) with wide community support, because it offers a clean separation between behavior and presentation, and because it can be easily integrated with the Java Portlet specification (JSR-168). As a result, JSF has been adopted by all of the major applications in the project, including the Interoperability Framework's own administration user interfaces.

In order to test, build, package, and distribute Planets results within a single software package, we have established an infrastructure for distributed software development, code management, and bug tracking. Unsurprisingly, we have chosen a framework used primarily for the development of open-source projects, GForge[18] (open-source software created for SourceForge), together with the revision control system Subversion[19] (SVN). In order to support a wide user base, the entire IF infrastructure can be downloaded as a single file from our GForge site and is configured entirely through this installation package. Additional applications and services can be optionally included from the installer, or added at a later time.

3.4 Performance and Scalability

A crucial aspect of the preservation system is the establishment of a distributed, reliable, and scalable computational tier. A typical preservation workflow may consist of a set of components for data characterization, migration, and verification and could be applied to millions of digital objects. In principle, these workflows could be easily parallelized and run in a massively parallel environment. However, the fact that preservation tools often rely on closed source, third party libraries and applications that often require platform-dependent and non-trivial installation procedures prevents the utilization of existing large-scale computing infrastructures, such as Grids. In order to efficiently execute a preservation plan, a varying set of preservation tools would need to be available on a scalable number of computational nodes.

Advances in virtualization allow the deployment of entire computational environments, including operating systems and applications, to distributed computational nodes. This allows one to instantiate sets of transient system images on demand, which can be federated as a virtualized cluster. We have implemented a prototype Job Submission Service (JSS) that can manage such infrastructures

[18] `http://gforge.org/gf/`
[19] `http://subversion.tigris.org/`

and execute Planets preservation workflows on a virtual computing cluster or *Cloud*. Initial experiments with the Amazon Elastic Compute Cloud (EC2) reported elsewhere [14] have demonstrated the feasibility of this approach for the Planets service architecture. Significant performance increases were achieved using only five cluster nodes, proving the potential of employing even small clusters for digital preservation actions on large volumes of data. Furthermore, these results indicated that the system achieves good scalability when significantly increasing the number of utilized cluster nodes.

The Storage Research Broker (SRB) of the San Diego Supercomputer center, along with the recent extension iRODS [15] provides an infrastructure for implementing preservation-specific policies on a data grid. However, its primary purpose is to provide scalable archival storage and metadata management, whereas the Planets IF approach is to provide scalable computational resources for carrying out preservation plans involving mass migration or characterisation of millions of digital objects.

4 Conclusions and Future Work

After the third project year, the Planets Interoperability Framework now provides a stable preservation infrastructure in its fourth release. This infrastructure is available for download[20] in a platform-independent Java-based installation package, with options for deploying additional Planets applications and services.

In the fourth and final year of the project, the primary focus will not be on functionality – component development will be limited primarily to bug-fixing – but on architecture improvements. Specifically, the database and application server layers will be separated in order to support clustering of Planets Instances, allowing the infrastructure to be deployed as a robust, high-availability service. In addition, work will continue along the lines of Schmidt et al. [14], in order to provide highly scalable computational services for preservation actions based on virtual computational clusters.

Finally, field tests of the Planets Software Suite will be carried out at partner institutions, demonstrating how the Interoperability Framework and the associated Planets applications and services can act as an added-value preservation action system for existing digital repositories at national libraries and archives.

Acknowledgments

Work presented in this paper is partially supported by the European Community under the Information Society Technologies (IST) Programme of the 6th FP for RTD - Project IST-033789.

References

1. Farquhar, A., Hockx-Yu, H.: Planets: Integrated Services for Digital Preservation. International Journal of Digital Curation, Vol. 2, No. 2 (2007)

[20] `http://gforge.planets-project.eu/gf/project/if_sp/`

2. Strodl, S., Rauber, A., Rauch, C., Hofman, H., Debole, F., Amato, G.: The DELOS Testbed for Choosing a Digital Preservation Strategy. In: Digital Libraries: Achievements, Challenges and Opportunities. LNCS, vol. 4312, pp. 323–332. Springer, Heidelberg (2006)
3. ISO Standard 14721:2003: Space Data and Information Transfer SystemsA Reference Model for an Open Archival Information System (OAIS). International Organization for Standardization (2003)
4. Giaretta, D.: The CASPAR Approach to Digital Preservation. The International Journal of Digital Curation. Issue 1, vol. 2. (2007)
http://www.ijdc.net/ijdc/article/view/29/32
5. Factor, M., Naor, D., Rabinovici-Cohen, S., Ramati ,L., Reshef, P., Ronen, S., Satran, J., Giaretta, D.: Preservation DataStores: New storage paradigm for preservation environments. IBM Journal of Research and Development on Storage Technologies and Systems, Volume 52, Number 4/5 (2008)
6. Oltmans, E., van Diessen, R.J., van Wijngaarden, H.: Preservation Functionality in a Digital Archive. JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries, 279–286 (2004)
7. van Diessen, R.J., van Rijnsoever, B.J.: IBM/KB Long-term Preservation Study: Managing Media Migration in a Deposit System (2002)
http://www-05.ibm.com/nl/dias/resource/migration.pdf
8. Janée, G., Mathena, J.; Frew, J.: A data model and architecture for long-term preservation. JCDL '08: Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries, 134–144 (2008)
9. Heslop, H., Davis, S., Wilson, A.: An Approach to the Preservation of Digital Records. National Archives of Australia Green Paper. (2002)
http://naa.gov.au/Images/An-approach-Green-Paper_tcm2-888.pdf
10. Hunter, J., Choudhury, S.: A semi-automated digital preservation system based on semantic web services. JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries, 269–278 (2004)
11. Brown, A.: Developing Practical Approaches to Active Preservation. The International Journal of Digital Curation. Issue 1, vol. 2. (2007)
http://www.ijdc.net/ijdc/article/view/37/42
12. Becker, C., Kulovits, H., Rauber, A., Hofman, H.: Plato: a service-oriented decision support system for preservation planning. JCDL '08: Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries, (2008)
13. Aitken,, B., Helwig, P., Jackson, A.N., Lindley, A., Nicchiarelli, E., Ross, S.: The Planets Testbed: Science for Digital Preservation. code{4}lib Journal, Issue 3, 2008-06-23 (2008)
http://journal.code4lib.org/articles/83
14. Schmidt, R., Sadilek, C., King, R.: A Service for Data-Intensive Computations on Virtual Clusters. INTENSIVE 2009: Proceedings of The First International Conference on Intensive Applications and Services, (to appear)
15. Rajasekar, A., Wan, M., Moore, R., Schroeder, W.: A Prototype Rule-based Distributed Data Management System. HPDC workshop on "Next Generation Distributed Data Management", May 2006, Paris, France (2006)