A Framework for Distributed Preservation Workflows

Rainer Schmidt, Ross King
AIT Austrian Institute of Technology
Donau-City-Strasse 1, Vienna, Austria

Andrew Jackson, Carl Wilson
The British Library
Boston Spa, West Yorkshire, UK

Fabian Steeg, Peter Melms
University of Cologne
Albertus-Magnus-Platz, Cologne, Germany

Abstract

The Planets project is developing a service-oriented environment for the definition and evaluation of preservation strategies for human-centric data. It focuses on the question of logically preserving digital materials, as opposed to the physical preservation of content bit-streams. This includes the development of preservation tools for the automated characterization, migration, and comparison of different types of digital objects as well as the emulation of their original runtime environment in order to ensure long-time access and interpretability. The Planets integrated environment provides a number of end-user applications that allow data curators to execute and scientifically evaluate preservation experiments based on composable preservation services. In this paper, we focus on the middleware and programming model and show how it can be utilized in order to create complex preservation workflows.

Introduction

There is a vital need to electronically preserve our cultural heritage as well as the digital outcomes of today's research. Ensuring long-term access to the plethora of existing digital file formats has become an important research challenge, in particular in the context of memory institutions. In addition to the physical preservation of the content bit-streams, a preservation system must ensure the interpretability of the digital objects with current and future applications in order to prevent a loss of information. The development of preservation plans and automated workflows represents a major research goal in this area. The Planets project develops an integrated environment for the development and evaluation of preservation strategies for cultural heritage data. It focuses on preservation requirements faced by cultural heritage institutions, in particular those of national libraries and archives (Farquhar and Hockx-Yu 2007).

A major problem is imposed by the diversity and richness of the human-centric data. A large amount of the digital information produced is for example stored in the form of word processor documents. In order to preserve this sort of data, one must be able to cope with a diversity of document formats. These formats often exist in various versions, are application-specific and/or depend on underlying operating systems in order to be successfully interpreted. In addition to rich formatting information, documents may contain and reference a variety of other objects like fonts, embedded metadata, document history, images and the like. Even if an archivist only considers a relatively small class of objects types for preservation (e.g. eprints), it can require considerable effort to transform the digital items into a format suitable for archiving, while verifying the authenticity and ensuring that no significant information had been lost during the transformation. This requires the deployment, evaluation, and operation of a large number of individual data processing tools, rendering environments, and software platforms. The preservation of digital materials can therefore become a labor-intensive and tedious process for the responsible data curators.

These shortcomings motivate the development of an integrated environment that supports the design, evaluation, and execution of preservation strategies. The Planets integrated environment provides a number of end user applications that allow preservation experts to conduct experiments based on and a large number of preservation services. The system integrates existing content repositories, preservation tools, and services into a distributed research infrastructure. It allows data curators to import digital collections, assemble and execute complex workflows, and evaluated the results. Here, we outline the underlying software infrastructure, known as the Planets Interoperability Framework (IF), which integrates the various heterogeneous preservation components into a coherent preservation system, based on a service-oriented architecture. Such components range from command-line utilities, software libraries, and online services, to emulated hard and software environments, for examples refer to (van der Hoeven 2007).

The IF workflow programming environment implements a component model that is specifically designed for the development of preservation processes. We provide an extensible set of preservation components that can be easily assembled into complex executable workflows. The data flow between the components is reflected by a digital object model that is capable of wrapping up different types of content (files, streams, hierarchical data structures) as well as encapsulate relationships, provenance information, and other metadata. In this paper, we outline some of the basic operations that are implemented by (currently more than 50) Planets preservation services. The interfaces are compatible

with each other and operate based on a minimal data abstraction outlined later in this paper.

Service-Oriented Approach

Managing the ever-increasing volumes of data provides a research problem across many academic areas. Examples are data produced in areas like science, engineering, and the arts and humanities (Hey and Trefethen 2003). Digital libraries and archiving services are responsible for the organization, preservation, and publication of data products and primary data. An important factor for the scalability of digital repositories is the automation of data management policies (Rajasekar et al. 2006). An example for the implementation of digital curation strategies using rule-based service enforcement is provided by the iRods system (Hedges, Hasan, and Blanke 2007). Another effort is undertaken by the Clarin project [1], which develops a service-oriented infrastructure for the automated processing of linguistic resources. However, it is clear that preservation management for digital repositories cannot be fully automated but requires the continuous effort of data curators. Research infrastructures, however, can greatly enhance the capabilities of their participants by fostering collaboration and access to remote and heterogeneous resources. Here, we present a system that assists preservation experts in the development of preservation strategies. It allows users to utilize and assess a large range of preservation tools and access remote data collections based on a service-oriented system.

The Planets Interoperability Framework provides an environment that is implemented based on a number of core software components (King et al. 2009) providing the technical backbone of a Planets instance. These components include authorization and authentication, workflow execution, service discovery, data and metadata management. As a whole, the framework is capable of creating an integrated environment that supports, controls, and secures the interaction of user applications with the distributed backend service infrastructure. The system is accessed by practitioners through a set of web-based applications that aid the user in the development and execution of preservation experiments. Two applications that make use of the preservation framework are the Testbed application [2] (Aitken et al. 2008) and the Preservation Planning Tool[3] *Plato* (Becker et al. 2008). These applications provide graphical interfaces for the evaluation of preservation strategies and decision support for preservation planning, respectively.

Figure 1 illustrates the high-level system architecture which is here separated into three distributed layers. The central component of the architecture is provided by the Planets gateway server (GS) which provides a controlled environment that operates on top of a large number of preservation and other services (S1...Sn). The gateway basically provides two communication substrates: firstly, a range of web services interfaces (Portal Services) that provide the user applications with the required means to utilize the service infrastructure in a secured, controlled, and reproducible way; secondly, an API (WFlow) for defining preservation workflows based on a set of unified components, which are outlined later in this paper.

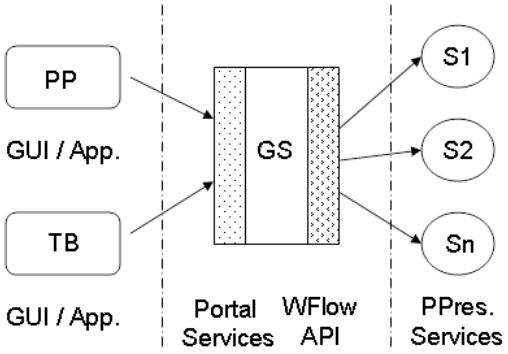

Figure 1: Preservation Gateway Layers

Digital Object Model

The Planets service API [4] employs a generic data abstraction, the Planets *Digital Object*. Digital Objects encapsulate the concept of single digital entities. They may be composed of one or more byte-streams and be associated with metadata. The data abstraction is used to represent digital entities that are consumed and/or produced by the services within the Planets infrastructure. For example, the workflow system can be used to obtain a digital object from a repository service, apply a preservation service, and deposit the resulting object via the Data Registry Service. It is important to note that the digital object abstraction does not implement a full-blown repository data model. The digital object model is intended to provide a minimal data abstraction that can be mapped against records retrieved from existing repository systems and vice versa. Moreover, the object model needs to be sufficiently expressive to reflect the preservation process and its outcomes, for example events, manifestations, object characteristics.

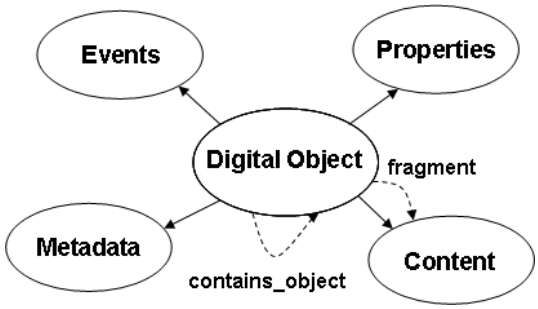

Figure 2: Schematic representation of the Digital Object Model

[1] www.clarin.eu

[2] http://testbed.planets-project.eu/testbed/

[3] http://www.ifs.tuwien.ac.at/dp/plato/intro.html

[4] a Java-based Application Programming Interface

Figure 2 shows a schematic representation of the Planets **digital object** implementation. In general, a digital object represents a referenceable content entity and its associated metadata. Digital objects can hold a set of bibliographic default **properties** like a name (mandatory), author, or a human readable description. They provide space for basic metadata, like repository URL and format, and also for arbitrary tagged **metadata** chunks. This form avoids any need to explicitly prescribe the nature of the high-level data model, while still allowing such metadata to be associated with an object. File formats are specified in the form of URIs based on the PRONOM ID schema (Brown) (e.g. info:pronom/fmt/122 for EPS version 1.2). The Planets Technical Registry provides a utility to translate between the PUID schema and the more general MIME types. An example of technical metadata is a checksum algorithm and value. Planets defines a basic type set of **events**; examples are creation, characterization, modification events. An event typically includes an actor, a timestamp, and a number of specific name-value pairs. The event model has been designed to comply with the event definition provided by the PREMIS schema (Online Computer Library Center (OCLC) May 2005). **Content** data-streams are associated with a digital object based on a reference (typically a repository URL) or can be directly embedded within the object, if desired. **Relationships** to other digital objects for example of types like "contained" or "derived from" are required for expressing multiple manifestations and for creating composite objects. The digital object abstraction provides a recursive concept that can represent compound object types that are composed from many different underlying objects. For example, consider a digital object that represents the root node of a file tree. If the digital object is passed to a service, "contained" child elements could be incrementally downloaded and processed. Another relationship is provided by the concept of *fractions*, which allow reference to content parts (e.g. frames of a video, files in a compressed package). Work is ongoing on serializing digital objects while making use of metadata formats like RDF, METS and PREMIS.

Repository Integration

Many memory institutions, like national libraries, and archives already have archiving systems in place. These are often custom solutions or based on commercial systems. Replacing such environments is neither feasible nor desirable. Therefore the Interoperability Framework was designed to integrate with and complement existing archive systems; it is in no way meant to replace them or even to provide archiving functionality. However, it can not be assumed that the Planets software has any control over an institutional repository and/or that it can be granted permission to automatically deposit materials there. Hence, a less intrusive approach has been implemented. Interoperability between existing digital object management systems and the Planets infrastructure has been based on a "mutual access" strategy. The basic scenario involves the following steps: (1) Digital assets are retrieved based on the particular public interfaces/protocols provided by a repository system. (2) The obtained records are converted to the Planets digital object model and ingested through the Data Registry Service. After this stage, the obtained *digital objects* are available for processing within the Planets environment. (3) The outcomes of a preservation experiment/process are made accessible through a Data Registry. Current implementations are based on Apache Jackrabbit [5] and the Fedora Commons [6] repository software. A user may download the resulting data/metadata entities, which can be subsequently deposited into an institutional archiving system.

Repository Access

As described earlier; in order to make data available for experiments, it is required to first retrieve the digital objects from a managed repository environment. Mechanisms to access a repository, the internal data model, and representation depend very much on the particular system in place. Other variations result from the type of data that is being archived and the organization with custody. In the area of memory institutions, a commonly supported standard is provided by the Protocol for Metadata Harvesting (OAI-PMH) [7]. Other access mechanisms that have been integrated include Web services (REST and SOAP-based), native APIs, and file-based exchange. As a matter of fact, it was necessary to rely on existing and individual interfaces provided by these systems to retrieve content and metadata. Another major obstacle for providing seamless access amongst different repositories and collections is imposed by a great variation in metadata usage - both syntactically and semantically. Therefore, a major requirement for the preservation system was the development of pluggable access components that unify the different data sets and encodings of the various data sources (refer to figure 3).

Digital Object Managers

The digital object managers implement the functionality for retrieving Planets digital objects from individual repositories and/or storage systems. A simplified version of the object manager interface is shown in figure 4 . When a collection is registered through the Data Registry Service, the individual data items are mapped to the digital object model and stored within the metadata repository. The system would, for example, map the title and description of a Dublin Core [8] record directly to the corresponding digital object attributes. Some repositories also embed technical information such as checksums and algorithm within an retrieved record. Metadata that is not interpreted can be still associated as tagged metadata chunks within a digital object.

Evaluation

During the past project year, a selected set of sample repositories and online data sources were chosen for integration - each with different characteristics and varying degrees of

[5] http://jackrabbit.apache.org/
[6] http://www.fedora-commons.org/
[7] http://www.openarchives.org/pmh/
[8] http://dublincore.org/

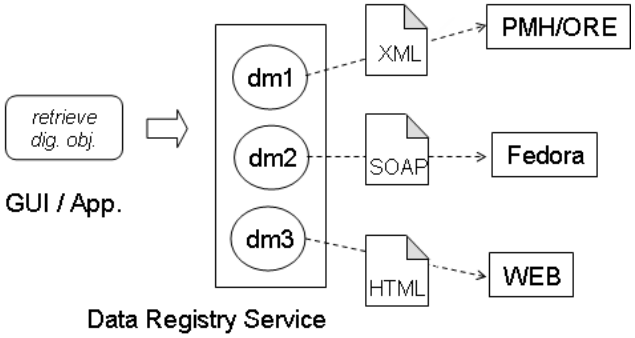

Figure 3: Content from different remote data sources can be referenced and accessed through the Planets data registry. A common interface to the client application is provided by bespoke Digital Object Managers (dm1..dmn). In order to access different repositories, the object managers translate the requests or queries to the respective interfaces and protocols exposed by the respective data sources.

```
interface DigitalObjectManager {
  void store(URI, DigitalObject)
  boolean isWritable(URI)
  List<URI> list(URI)
  DigitalObject retrieve(URI)
  public List<Query> getQueryTypes()
  public List<URI> list( URI, Query)
}
```

Figure 4: A unified interface for retrieving Digital Objects from different and distributed data resources. The basic functionalities are query, list, and retrieve. Write access for depositing experiment results is supported by the Planets Data Registry.

standards compliance. These data sources were made available within the Planets Testbed through the use of specific Digital Object Managers. Ingesting a digital collection has been accomplished through a graphical data registry browser (figure 5) and a corresponding digital object manager. This allows experimenters to dynamically retrieve remote data items and utilize the digital objects as part of an experimentation workflow. Future work in this area will deal with the inclusion of OAI-ORE *resource maps* (Maslov et al. 2009) in order to enhance the structuring of the data products that are disseminated through the Data Registry.

Preservation Services

The standardization of service interfaces for basic, atomic preservation actions is of crucial importance to this service-oriented approach. These definitions are provided as annotated Java interfaces, and any service developer needs only to implement a single interface in order to create a Planets-compatible preservation service. This means that Planets services are easy to swap or combine, making it simpler to create software that is capable of invoking many different

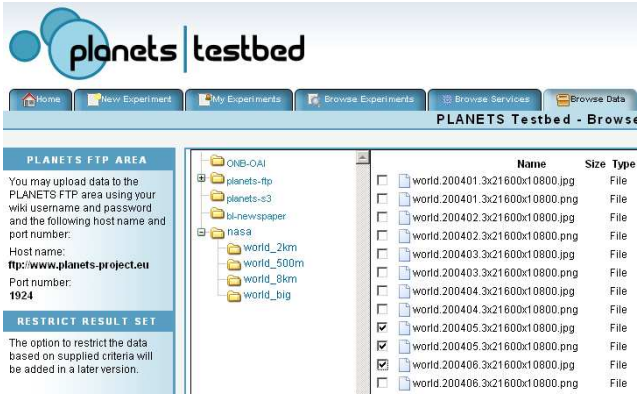

Figure 5: The Repository Browser implemented by the Testbed application. Users can select from different data sources, browse collections, and select objects. Incorporated sources include for example an repository of the Austrian National Library (ONB), the Amazon S3 storage service, Web resources, and a collection of digitized newspapers [10] of the British Library.

preservation tools.

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. These *level-one* service interfaces define atomic digital preservation verbs, for example:

- *Characterize* provides a generic interface for different characterization tools like JHOVE [11], the XCEL Extractor[12], and the New Zealand Metadata Extractor[13].
- *Compare*: Compares different objects based on metadata, object properties, or a normalized representation.
- *Identify*: Provides an interface to wrap format identification tools, e.g. DROID [14], or the unix file service, returning a URI format identifier.
- *Migrate*: Provides a generic interface for format migration tools.
- *Modify*: A component that modifies digital objects (e.g. enrich, corrupt, repair, crop), but doesn't change their format.
- *Validate*: Validates digital objects against file format specifications and schema definitions.
- *CreateView*: Renders a digital object, e.g. by utilizing an emulated environment.

[11] http://hul.harvard.edu/jhove/
[12] http://planetarium.hki.uni-koeln.de/public/XCL/
[13] http://meta-extractor.sourceforge.net/
[14] http://sourceforge.net/projects/droid/

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: the operations are atomic, Planets service data types are used for parameters and returns, and binary data is handled implicitly using a Planets Digital Object instance.

Service Discovery

In addition to a messaging interface (the WSDL document), Planets preservation services must expose a defined metadata document that describes the functionality implemented by the service. Once a service is registered with the preservation system, a rich service descriptor for each preservation service endpoint is generated and automatically registered with the Service Registry. The IF service registry provides a fine-grained service discovery mechanism including an extensible, schema- and taxonomy-based service categorization system. Moreover, the registry maintains information including tool identifier, accepted file formats, pathways, and/or default parameters. The service registry is accessible via a graphical as well as a programmatic interface.

Service Orchestration

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 a repository of individual templates 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 and data types. 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 institutions digital repository.

Workflow Environment

A crucial requirement of the programming environment is to allow data curators and archivists to assemble and deploy the preservation workflows they require, without forcing them to understand the underlying technical details. It is therefore important to abstract away the complexity of the underlying architecture. This can be done by structuring the system into different abstraction layers and by employing higher-level workflow representations. Here, we present an approach that basically distinguishes between two user groups; *developers* that implement workflows and *experimenters* that apply workflows. Our approach provides a separation of concerns, so that not every party intending to use the preservation system needs to understand the entire communication and data model. Figure 6 outlines the workflow enactment process and involved concepts.

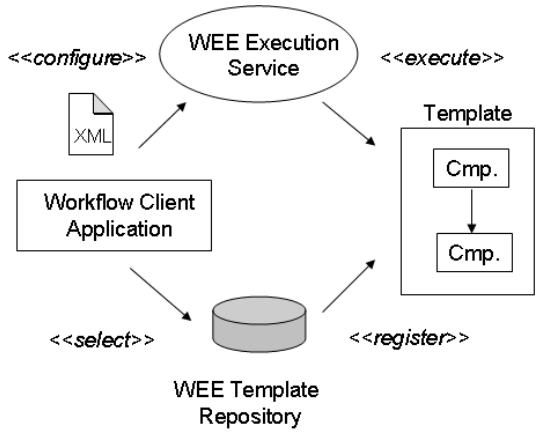

Figure 6: Workflow Enactment: Pre-defined *Workflow Templates* define the service orchestration based on composed high-level components. Clients can choose from registered templates and parameterize them based on XML descriptor files. Workflow instances are scheduled for execution using the workflow execution engine.

Workflow Templates

Workflow templates are pre-defined workflow fragments that solely implement abstract process logic but do not specify concrete services, tools, or their parameterization. We provide an extensible set of preservation components (based on a Java API) that can be used to easily implement complex workflows specifications. The higher-level and trusted workflow components operate on top of the lower level preservation Web services (the level-one services outlined in the previous section) and encapsulate details like messaging and metadata.

Parameterization

Workflow templates implement reusable patterns that are made available to Planets users through import into a Template Repository. The Template Repository provides service interfaces to register, browse, and retrieve workflow template definitions. In order to schedule a workflow execution, a user submits a descriptor document as well as a pointer to the data registry to the Workflow Execution Service. A descriptor basically contains the identifier of a template and defines its parameterization, as shown in figure 7.

Workflow Execution

The workflow execution engine (WEE) provides a web service for the execution and monitoring of workflow instances. The main purpose of the WEE is the provision of a controlled environment for the specification and execution of preservation processes. It implements an enactor that governs the orchestration of the various preservation components, which encapsulate functionalities like communication, state management, and preservation metadata handling.

```
<workflowConf xmlns:xsi="...">
  <template>
    <class>templates.MigrateByValue</class>
  </template>
  <services>
    <service id="migrate">
      <endpoint>
        http://saturn.ait.ac.at/Mdb2SiardMigrate?wsdl
      </endpoint>
      <parameters>
        <param>
          <name>planets:service/migration
            /input/migrate_to_fmt</name>
          <value>planets:fmt/ext/siard</value>
        </param>
      </parameters>
    </service>
  </services>
</workflowConf>
```

Figure 7: Example workflow configuration file for the invocation of a migration service. A workflow template that implements the a simple migration process logic is selected. The template is configured by specifying a migration service (Mdb2SiardMigrate) a well as a target format (siard). The available workflow templates can be browsed and inspected using the workflow repository service.

The WEE performs workflow execution asynchronously and may deliver status information to the user based on inquiry and email notification capabilities. It currently provides a generic web browser client or can be accessed by end user applications using web service or native interfaces.

Workflow Control Panel

In general, the workflow execution engine is accessed by the preservation applications through its Web service API. Additionally, we provide a generic graphical client application for the workflow environment, called the Workflow Control Panel (WCP). The WCP (figure 8) provides a graphical user interface that allows one to choose from various abstract workflow scenarios (templates). The workflow templates are then rendered and visualized. Selected workflow templates are configured using the GUI representation (e.g. drop-down boxes) and can be executed and monitored using the Web application. Additionally, users can upload/generate an XML representation of the actual workflow instance. In general, the WCP is used for testing and so called “informal experiments”. Client applications that implement a planning methodology or scientific experimentation process use the WEE implicitly and provide a custom user interface representation.

Conclusion

In this paper, a we have presented a prototype environment for the execution of digital preservation strategies based on distributed preservation services. We argue that preservation systems in particular have strong dependencies on legacy applications and third party services. Therefore, research on unified preservation interfaces, standardized service profiles, and programming models is crucial to the interoperability

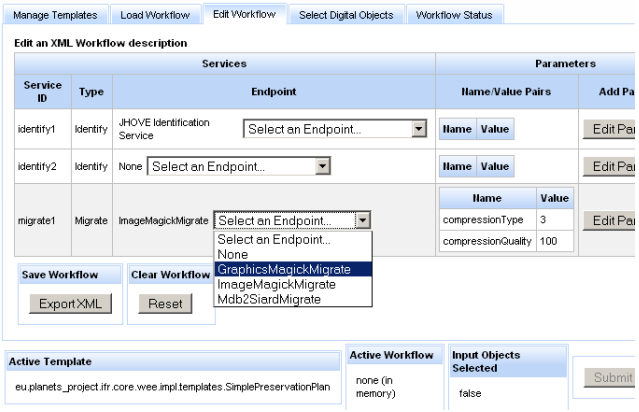

Figure 8: Dynamically generated representation of a workflow template by the workflow control panel.

and reusability of current and future preservation tools and components. Future work on the workflow environment will deal with resource management and scalablity issues.

Acknowledgments

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

References

Aitken, B.; Helwig, P.; Jackson, A. N.; Lindley, A.; Nicchiarelli, E.; and Ross, S. 2008. The Planets Testbed: Science for Digital Preservation. In *Code4Lib*, volume 1(5).

Becker, C.; Kulovits, H.; Rauber, A.; and Hofman, H. 2008. Plato: a service oriented decision support system for preservation planning. In *JCDL ’08: Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries*, 367–370. ACM.

Brown, A. The PRONOM PUID Scheme: A scheme of persistent unique identifiers for representation information. Technical report.

Farquhar, A., and Hockx-Yu, H. 2007. Planets: Integrated Services for Digital Preservation. In *International Journal of Digital Curation*, volume 2(2), 1746–8256.

Hedges, M.; Hasan, A.; and Blanke, T. 2007. Curation and Preservation of Research Data in an iRODS Data Grid. In *E-SCIENCE ’07: Proceedings of the Third IEEE International Conference on e-Science and Grid Computing*, 457–464.

Hey, A. J. G., and Trefethen, A. E. 2003. The Data Deluge: An e-Science Perspective. In Berman, F.; Fox, G. C.; and Hey, A. J. G., eds., *Grid Computing - Making the Global Infrastructure a Reality*, 809–824. Wiley and Sons.

King, R.; Schmidt, R.; Jackson, A.; Wilson, C.; and Steeg, F. 2009. The Planets Interoperability Framework - An

Infrastructure for Digital Preservation Actions. In *Proceedings of the European Conference on Digital Libraries (ECDL 2009)*.

Maslov, A.; Mikeal, A.; Phillips, S.; Leggett, J.; and Mark, M. 2009. Adding OAI-ORE Support to Repository Platforms. In *Proceedings of the 4th International Conference on Open Repositories (OR09), Atlanta, GA, USA.*

Online Computer Library Center (OCLC). May 2005. Data Dictionary for Preservation Metadata: Final report of the PREMIS working group. Technical report.

Rajasekar, A.; Wan, M.; Moore, R.; and Schroeder, W. 2006. A prototype rule-based distributed data management system. In *in: HPDC workshop on "Next Generation Distributed Data Management, Paris, France.*

van der Hoeven, J. 2007. Dioscuri: Emulator for Digital Preservation. In *D-Lib Magazine*, volume 13 (11/12).