2006

Synchronization and Multiple Group Server Support for Kepler

K. Maly
Old Dominion University

M. Zubair
Old Dominion University

H. Siripuram
Old Dominion University

S. Zunjarwad
Old Dominion University

Yannis Manolopoulos (Ed.)

See next page for additional authors

Original Publication Citation

Maly, K., Zubair, M., Siripuram, H., Zunjarwad, S. (2006). Synchronization and multiple group server support for Kepler. In Y. Manolopoulos, J. Filipe, P. Constantopoulos, & J. Cordeiro (Eds.), *Proceedings of the Eighth International Conference on Enterprise Information Systems*. SciTePress. https://doi.org/10.5220/0002495401110117

SYNCHRONIZATION AND MULTIPLE GROUP SERVER SUPPORT FOR KEPLER

K. Maly, M. Zubair, H. Siripuram, S. Zunjarwad
Old Dominion University, Computer Science Department, Norfolk,VA-23508,USA

Keywords: Digital Libraries, Open Archives Initiative.

Abstract: In the last decade literally thousands of digital libraries have emerged but one of the biggest obstacles for dissemination of information to a user community is that many digital libraries use different, proprietary technologies that inhibit interoperability. Kepler framework addresses interoperability and gives publication control to individual publishers. In Kepler, OAI-PMH is used to support "personal data providers" or "archivelets".". In our vision, individual publishers can be integrated with an institutional repository like Dspace by means of a Kepler Group Digital Library (GDL). The GDL aggregates metadata and full text from archivelets and can act as an OAI-compliant data provider for institutional repositories. The basic Kepler architecture and it working have been reported in earlier papers. In this paper we discuss the three main features that we have recently added to the Kepler framework: mobility support for users to switch transparently between traditional archivelets to on-server archivelets, the ability of users to work with multiple GDLs, and flexibility to individual publishers to build an OAI-PMH compliant repository without getting attached to a GDL.

1 INTRODUCTION

In the last decade literally thousands of digital libraries have emerged; one of the biggest obstacles for dissemination of information to a user community is that many digital libraries use different, proprietary technologies that inhibit interoperability. Building interoperable digital libraries will allow communities to share information across institutional and geographic borders. One major effort that addresses interoperability is the Open Archive Initiative (OAI) (Lagoze, 2002) that has developed a framework to facilitate the discovery of content stored in distributed archives.

One of the efforts in this direction is (Maly, 2001), for the overall framework see Figure 1, that gives publication control to individual publishers, supports rapid dissemination, and addresses interoperability. In Kepler, the OAI framework is used to support what we call "personal data providers" or "archivelets". At this time, we have implemented the system that lets any user create an OAI-compliant archivelet using simple, self-contained, self-installing software. As a demonstration, we have setup a registration service and a service provider at Old Dominion University (Kepler, 2005). Once an archivelet registers with our registration service, the service provider can harvest metadata from it. We believe Kepler is a small step toward our long-term vision of providing tools and software for communities to easily deploy digital libraries that are customized for their needs, can be easily populated, managed, and are "open" for development of future services. We define a communal digital library as a federation of smaller group based digital libraries. In our model, a community represents users that wish to share digital objects of common interest. A group is a sub-set of community users that share the same publication and management process. For example, a department dealing with water pollution in a research laboratory is a group and several such groups in other laboratories and universities form a community.

The architecture of just the archivelet is summarized in (Maly, 2001); (Maly, 2003) reports on the general system architecture with a number of new features added (see Figure 2 for the Kepler user interface). These include the concept of a server-side archivelet, persistent URLs, NAT/proxies,

Maly K., Zubair M., Siripuram H. and Zunjarwad S. (2006).
SYNCHRONIZATION AND MULTIPLE GROUP SERVER SUPPORT FOR KEPLER.
In *Proceedings of the Eighth International Conference on Enterprise Information Systems - DISI*, pages 111-117
DOI: 10.5220/0002495401110117

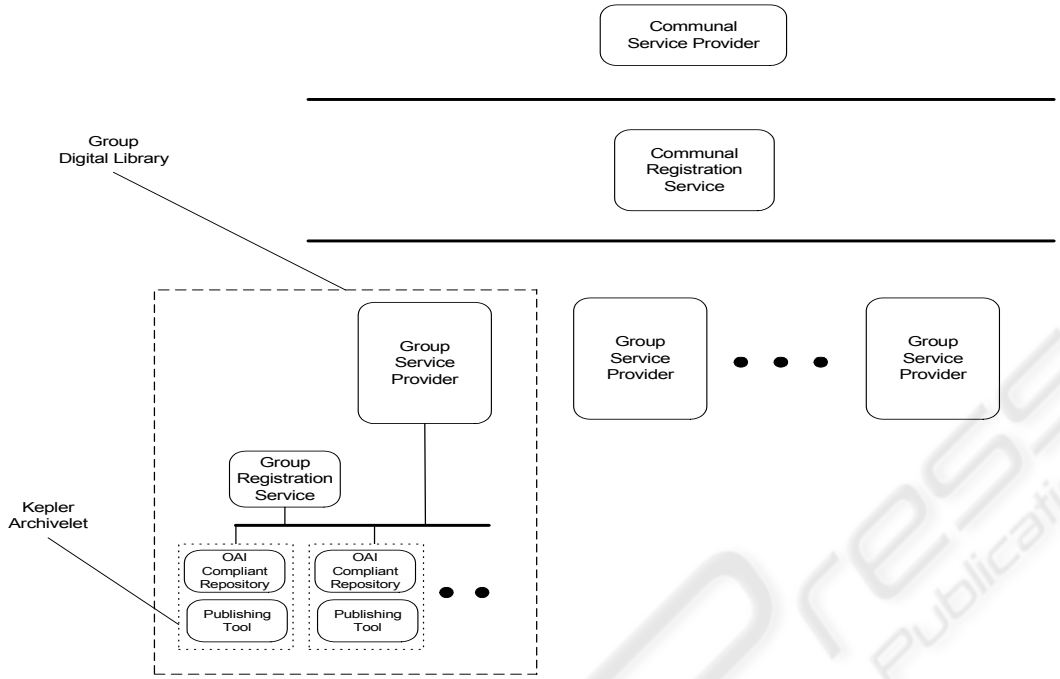

Figure 1: Kepler framework.

import/export, caching and a validation tool. The server-side archivelet addresses the problem of allowing users away from their machine that runs the archivelet to publish and manage their digital library. Persistent URLs and NAT/proxies address the problem that arise when archivelets are behind a router that hides the network addresses. The import/export solves the problem of moving entire subsets of documents from one archivelet to another. Caching – done at the group server that harvests an archivelet - solves the problem of a document not being available to a search user as the archivelet that stores it is not on-line. Finally, the validation tool enables a community to set standards for the process of entering metadata and having these standards enforced at publication time.

In this paper we report on three new major steps in making the Kepler system more usable: synchronization, multiple community registration, and the generic archivelet. Synchronization, reported in Section 3, addresses the issue of synchronizing the server-side and the local archivelets. That is, whenever a document is published in either the local or the server-side archivelet, the system ensures that it appears in the other archivelet. Multiple community support, described in Section 4, means that a user can select which of her papers are registered to what group server (community). Only the papers registered to a group server are harvested by that group server. The generic archivelet, reported on in Section 5, is for the user who wants to use the Kepler archivelet as an independent, small, and light-weight digital library that is OAI-compliant. That is, it can be harvested by any OAI service provider as it has a persistent URL.

2 BACKGROUND

Today, there exists many open source institutional repository software, for example the eprints.org software from the University of Southampton has been widely adopted; and DSpace (Smith, 2003) is created by Hewlett Packard and the MIT Libraries, but has been widely adopted outside of MIT. All of these systems feature significant features and capability for building large-scale digital libraries and institutional repositories. All also use the OAI-PMH as a core technology. However, they are in contrast to Kepler in that they are institutional centric and not individual publisher centric. The choice of a community of institutional digital library is an extremely important one, and we

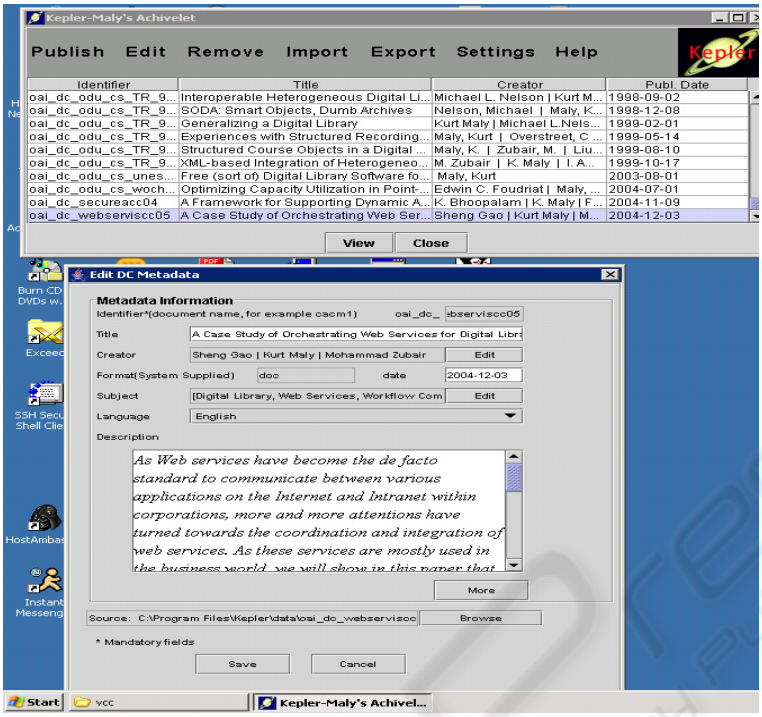

Figure 2: Kepler archivelet user interface.

recommend surveys such as (Brogan, 2003) as guides in determining which DL suits your needs.

Kepler as developed over the last few years draws from a significant base of existing OAI-PMH projects, developed both at Old Dominion and throughout the community. In addition to the original Kepler project (Maly, 2001), some of the features draw from the Arc (Arc, 2005) project. The OAI-PMH Static Repository format (Hochstenbach, 2003) was adopted when the need for archivelet importing and exporting was addressed. The persistent URL work is similar to the Extensible Repository Resource Locators (ERRoLs) project currently underway at OCLC (Young, 2005).

3 SYNCHRONIZATION OF ON-SERVER AND LOCAL ARCHIVELETS

In the previous version of the Kepler system, the group server, being a reliable, scalable digital library in itself, maintained a data base of the harvested records. The local archivelet and the server-side archivelet were not connected and had their own storage system. The problems we are facing when we try to have the two archivelets represent the same collection virtually all the time are twofold. One problem is to maintain consistency of the collections visible in each archivelet in the face of the possibility that the user publishes in both places. In turn the group server should reflect that same consistent state in its visible collection of metadata records. The second problem is one of efficiency and low latency, that is, we want to minimize redundant work and we want to minimize the time where the state is inconsistent. Our approach to the problem, guided by similar problems in distributed databases was to first combine the separate storage system at the server by having one database of metadata records for both the group server and the server-side archivelet (see the top part of Figure 3). That allows the server-side archivelet to see the records published at the local archivelet immediately after harvesting. However, we still have the problem of synchronizing traditional archivelet publications with the server-side archivelet. This is an active research problem and is being addressed by database community for example, (Barbara, 1990). There are several commercial databases that support synchronization, also some time referred as replication (Agrawal, 1990). Some example issues we face in synchronization are: synchronization

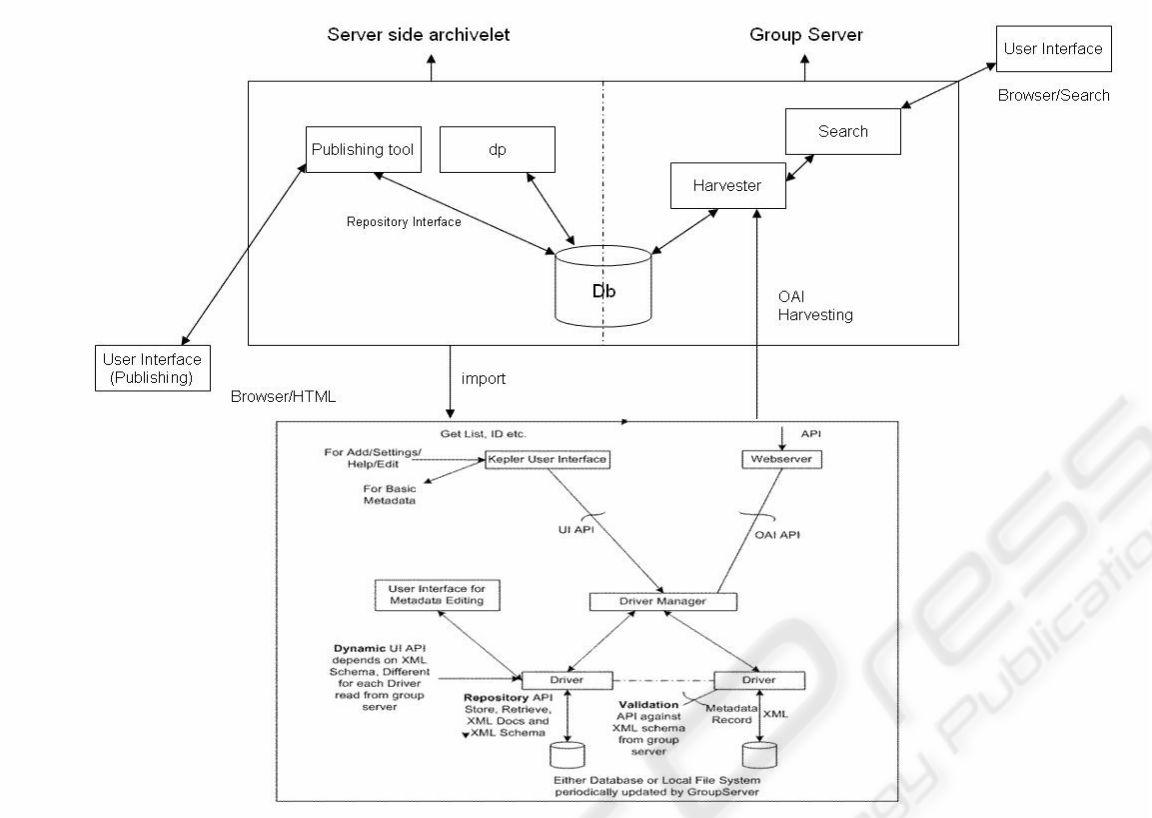

Figure 3: Synchronization architecture.

granularity (field level, record level, table level, etc.); we implement synchronization at the record level and use OAI date stamp for efficient management of synchronization. To simplify our implementation, we decided to use the import feature to transmit changes at the server-side to the local archivelet (as opposed to build an OAI harvester on top of the archivelet), again see Figure 3.

When the user publishes the record on the local (also called traditional) archivelet, the service provider (group server) is notified to start a harvester thread. If the OAI server at the traditional archivelet is up, the records will be harvested and stored in the database of the service provider.

As the service provider and the server-side (also called on-server) archivelet share the same database, the harvested records are now reflected at the on-server archivelet account. The harvester checks the date stamp of the harvested record with the record currently available before updating the database. This guarantees that to always have the latest modified records at the on-server archivelet. When the user publishes/edits records at the on-server archivelet, it creates an 'export' xml file representing the latest records on the on-server archivelet. We send a notification (and an URL) for importing this xml file to the tradition archivelet. If the OAI server is running the local archivelet will then 'import' the file immediately. If the local archivelet is down, it will check with the group server when it comes up again for any notification messages and act on them. In either case the import routine at the local archivelet checks for freshness of the record by comparing the date stamp.

The two routines (import and harvesting) need to take care of a number of issues. They include: What happens when one archivelet does not yet exist? What happens when a record gets changed versus newly created? What happens when a record gets deleted on either archivelet? How can we ensure that the records are the freshest possible? Within the major decisions of having a combined database at the server and to use export/import these are issues we dealt with mainly by using notification and time stamp.

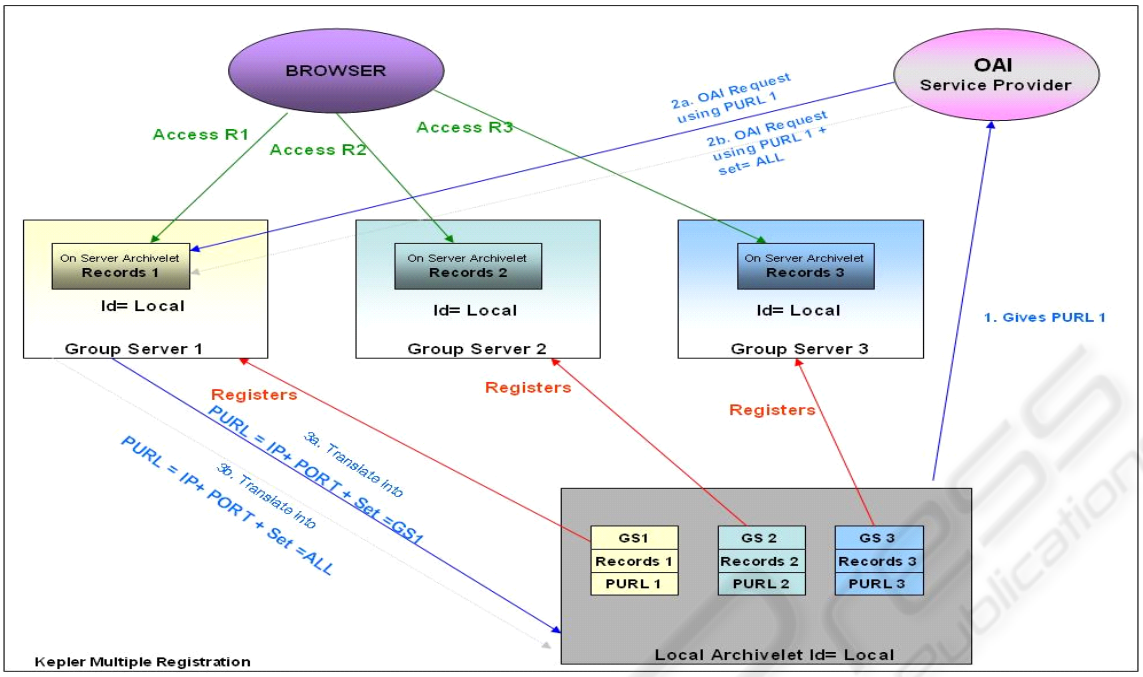

Figure 4: Kepler support of multiple communities.

4 MULTIPLE COMMUNITIES' SUPPORT

The earlier release of Kepler framework allowed a Kepler archivelet to work with only the original group server - the server where the user downloads the archivelet from. In the new release, which is the focus of this paper, we eliminated this constraint and gave flexibility to users to register their archivelet with multiple group servers. The motivation for adding this feature is to address individual publishers who are active in multiple areas of research and need to publish their work with different group servers. To support this feature, we need to address two major issues: harvesting of an archivelet and maintaining the simplicity of the user interface. For harvesting, we use the SET feature of OAI-PMH to make distinction between different group servers. We now discuss in some details our approach in addressing these issues.

Harvesting. We need to provide support for selective harvesting of records from an archivelet based on the group server association. For example, a group server A should be able to harvest all records published in the archivelet only for group server A. At the same time we also wanted to support cases where an OAI service provider (outside the realm of Kepler framework) may be interested in harvesting "All" records published in an archivelet.

For selective harvesting of records, we use the SET feature of OAI-PMH, which is an optional construct for grouping items for the purpose of selective harvesting. The archivelet organizes records into sets, where each set corresponds to a group server. An archivelet has as many sets as the number of group servers with which it has registered. A typical OAI-PMH request from a group server A looks like: http://.../oai?verb=ListRecords&from=2005-05-18&metadataPrefix=oai_dc&set=A

On receiving such a request, the archivelet returns all records published under group server A. An OAI service provider (typically outside the realm of a group server), harvesting all records, issues an OAI-PMH request with "set = ALL". As we are supporting persistent URL (PURL) by maintaining a table in the group server, all OAI-PMH requests for an archivelet actually go to a registered group server for getting the current IP address of the archivelet. When a harvesting request is originating from a registered server, we discover the current address of the archivelet using the PURL table maintained in the group server. An OAI-PMH request originating from an OAI service provider that is not a registered group server, the translation is handled by the group server pointed to by the base URL publicized by the archivelet owner. We illustrate this in the Figure 4.

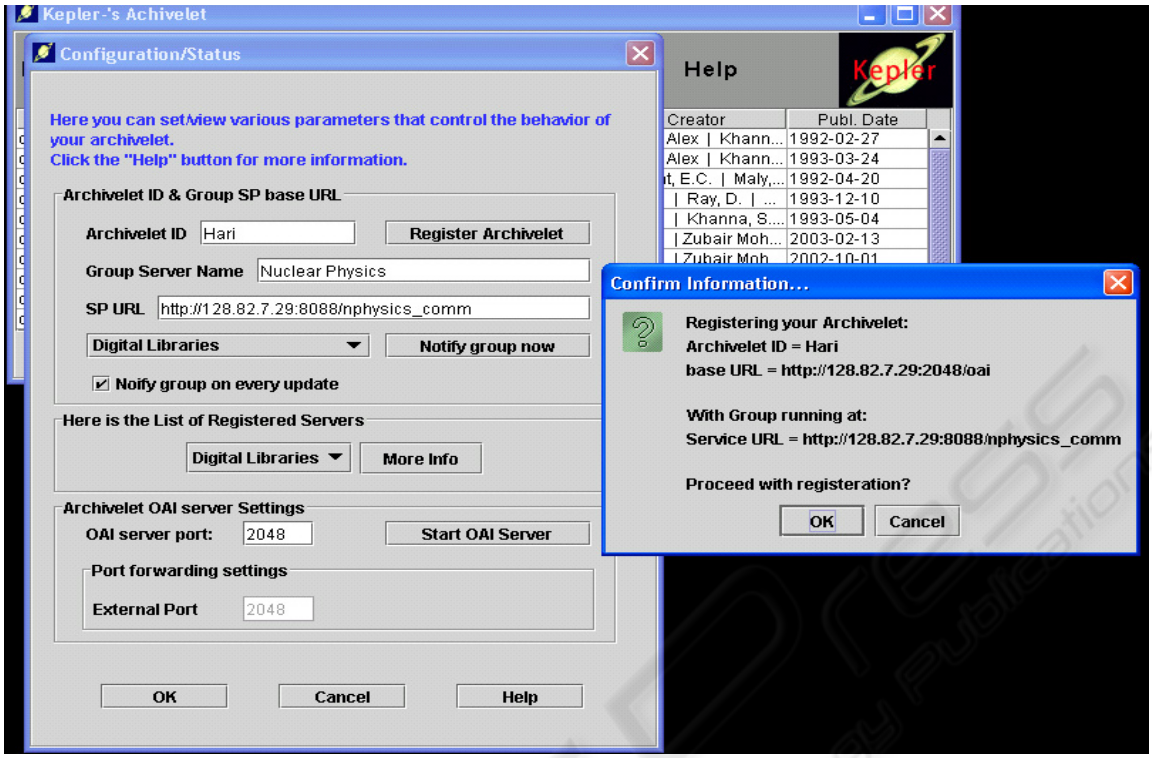

Figure 5: Registration interface for multiple communities.

User Interface. At the user interface level, we need to support publication, listing all groups, and import/export features in presence of multiple groups. To keep the user interface simple, we introduce a concept of "location" on the user interface, which is basically a pull-down menu that lists the registered group server. A user can select a location (a registered group server) from this list and then all the publication, listing, and importing/exporting of records is constrained to the selected group server. We also introduce a special location "All", which when selected in listing mode, lists all records published in the archivelet. Before a user can publish a document or use import/export, the user has to select one of the registered group servers.

5 GENERIC ARCHIVELET

The objective of supporting generic archivelet is to provide flexibility to individual publishers that do not want to be associated with a group server, yet want to build an OAI compliant repository. They wish to expose their collection to OAI service providers but not necessarily to group service providers in a Kepler framework. One of the major issues is the question of where to support the PURL table that provides the current address of the archivelet. For registered archivelets, the Kepler group server(s) maintains the PURL table. For a generic archivelet we do not have any registered group server. To support generic archivelets, we introduce a PURL resolver service that is maintained by the Old Dominion University digital library research group. A generic archivelet register itself to this PURL service and when the archivelet is starts it contacts the PURL service to update its current address.

The other major issue is that we wanted – in order to be able to maintain the code easily - to use the same code package for generic archivelet as well as traditional archivelets that are associated with Kepler group servers. A registered archivelet has constraints on publishing metadata defined and enforced by the community through the group server. For example, a group server can make the "abstract" metadata field mandatory, thus requiring all registered archivelet to enter content for this metadata field when publishing. On the other hand, a generic archivelet comes with minimal default

constraints that are preset and can not be changed. To support this feature with the same code package, we have introduced a "generic archivelet" mode in the archivelet configuration component. Thus, an archivelet becomes generic depending on a configuration file that is part of the download package used.

6 STATUS OF SYSTEM

We have implemented the complete Kepler package using Java technology and is available for download at sourceforge net site

(http://sourceforge.net/project/showfiles.php?group_id=107191&package_id=115659&release_id=233083).

The group server package is supported both on Unix and Windows platform. The package consists of a group server along with support for distributing traditional archivelet software. Once the group server is deployed, a user can download the traditional archivelet from the group server's web site.

After downloading the archivelet, a user can register the archivelet with multiple group servers using the setting menu as illustrated in Figure 5. The user interface supports the location pull down menu that allows user to select one of the registered group servers for the purpose of publication, listing, and importing/exporting of records.

7 CONCLUSION AND FUTURE WORK

In this paper we have described three significant additions to the Kepler system that makes it attractive to be deployed in different environments. We allow research communities to overlap and researchers to have multiple interests and hence papers on different group servers. The second addition was the interconnection of local archivelets and server-side archivelets and to keep them consistent within themselves and with the group server(s). Finally we described the generic archivelet that allows researcher not affiliated with any community to publish with a very lightweight digital library tool and be OAI-compliant. All these features have been implemented and thoroughly tested and made available on Sourceforge under Opensource. We also provide in our lab a PURL server for the generic archivelets that can be downloaded from our website (http://kepler.cs.odu.edu).

The key item in our future work will be to test the user acceptance of these features and perform field tests with several communities. In doing so , we may have to create additional drivers for other metadata sets. We are, in particularly interested in a driver for complex objects that will allow users to publish composite objects and provide hierarchical labeling of these objects.

REFERENCES

Agrawal, D. AND El-Abbadi, A., 1990. The tree quorum protocol: An efficient approach for managing replicated data. In *Proceeding of the Sixteenth International Conference on Very Large Databases*.

Arc, 2005. http://arc.cs.odu.edu

Barbara, D. AND Garcia-Molina, H., 1990. The case for controlled inconsistency in replicated data. In *Proceedings of the IEEE workshop on replicated data*.

Brogan, M.L.,2003. A Survey of Digital Library Aggregation Services. The Digital Library Federation Council on Library and Information Resources, Washington, DC. Available at *http://www.diglib.org/pubs/brogan/*

Hochstenbach, P., Jerez, H., Van de Sompel, H., 2003. The OAI-PMH Static Repository and Static Repository Gateway. In *Proc. of the third ACM/IEEE Joint Conference on Digital Libraries, Houston TX, pp. 210-217*.

Kepler, 2005. http://kepler.cs.odu.edu/

Lagoze, C., Van de Sompel, H., Nelson, M., Warner, S., 2002.The Open Archives Initiative Protocol for Metadata Harvesting. Available at *http://www.openarchives.org/OAI/openarchivesprotocol.htm*

Maly, K., Zubair, M., Liu, X., 2001. Kepler - An OAI Data/Service Provider for the Individual. D-Lib Magazine 7(4), Available at *http://www.dlib.org/dlib/april01/maly/04maly.html*

Maly, K., Nelson, M., Zubair, M., Amrou, A., Kothamasa S., Wang L., Luce, R., 2003. Light-Weight Communal Digital Libraries. In *Proc. of the fourth ACM/IEEE Joint Conference on Digital Libraries, Tucson AZ, pp.237-238*

Smith, M., Barton, M., Bass, M., Branschofsky, M., McClellan, G., Stuve, D., Tansley, R., Harford W. J.,2003. DSpace - an open source dynamic digital repository. D-Lib Magazine, 9(1), Available at *http://www.dlib.org/dlib/january03/smith/01smith.html*

Young, J.A., 2005. Extensible Repository Resource Locators (ERRoLs) for OAI Identifiers. Available at *http://www.oclc.org/research/projects/oairesolver/default.htm*.