Representing LEAD Experiments in a FEDORA digital repository

You-Wei Cheah, Beth Plale
Indiana University
Bloomington, IN
{yocheah, plale}@cs.indiana.edu
IU-CS TR666

Abstract

In this paper, we discuss the representation of heterogeneous collections of scientific data in a Fedora digital repository. We discuss and demonstrate our approach, identify the problems we encountered, and outline our solutions.

1 Introduction

Scientific data has numerous relationships and attributes (metadata), which may be specific to its domain. Projects such as the Linked Environments for Atmospheric Discovery (LEAD) [1], a cyber infrastructure for mesoscale meteorology research and education provides these scientific data. LEAD data, for example, has relationships in the form of projects, experiments, collections and files, as can be seen in Fig.1[1]. This figure shows a workspace of a typical LEAD user, which consists of files and collections generated by workflows. Metadata about a science data collection is needed to share the collection with others, or even for a scientist to find and use a collection some months or years after he or she generated it. Often metadata generated in a scientific domain is described using one or more community metadata schemas. These XML schemas represent, among other information, application specific attributes of a data collection and sometimes encode a community vocabulary. LEAD metadata schema (LMS) [2] is a profile of FGDC. FGDC adheres to the ISO 19115 standard (ISO international metadata standard). Fig.2[2] shows a sample of the LMS.

LEAD experiments are nested collections of LEAD objects. LEAD experiments may contain any number of files and collections, or even experiments (although this currently does not exist in an experiment). Collections (and experiments) may further contain nested collections and files.

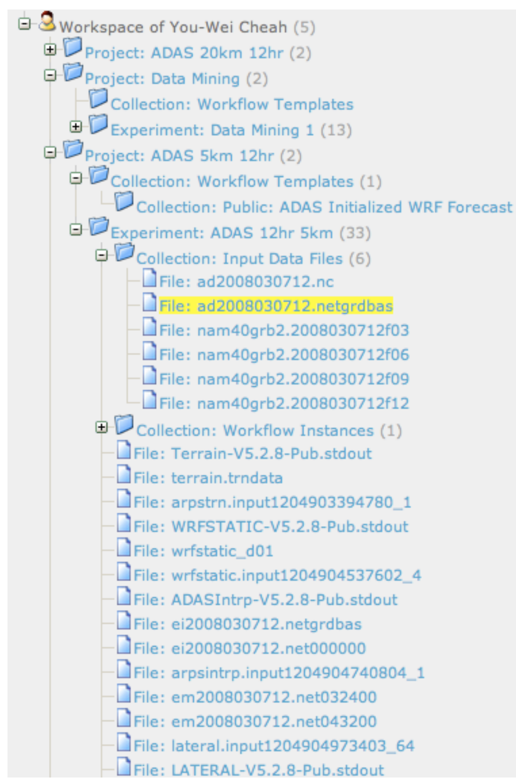

Fig. 1 Workspace in LEAD

[1, 2] Fig. 1, 2 obtained from http://portal.leadproject.org

All LEAD objects have a LEAD metadata representation. Experiments and collections are represented solely by metadata. Files are a little different, because, in addition to having a metadata representation, files also have actual content. This content may be in any form of data, such as XML, binary, ASCII, etc.

In this paper we investigate the problem of representation of rich collections of science data in a Fedora [3] repository. Fedora has support for representing metadata, but its suitability in representing scientific collections is less well understood. Specifically, *how can this basic support be used to our purposes? Does Fedora aid or detract from the need of scientists to represent and view data in their metadata schemas? Finally, how well suited is Fedora to ingesting the large data volumes typically found in science data?* We address these questions in this paper.

Our approach to understanding the representation of LEAD experiments in a Fedora repository is to examine the tradeoffs of modeling a single LEAD experiment first as a single digital object and then by modeling a LEAD experiment as multiple digital objects.


```
      <fgdc:metd>20080504</fgdc:metd>
    - <fgdc:metstdn>
        LEAD Profile of FGDC Content Standard for Digital Geospatial Metadata
      </fgdc:metstdn>
      <fgdc:metstdv>FGDC-STD-001.LEAD_1_2-2005</fgdc:metstdv>
    </lead:metainfo>
  - <lead:geospatial>
    - <lead:idinfo>
      - <lead:timeperd>
        - <fgdc:timeinfo>
          - <fgdc:rngdates>
              <fgdc:begdate>20080501</fgdc:begdate>
              <fgdc:begtime>000000000</fgdc:begtime>
              <fgdc:enddate>20080502</fgdc:enddate>
              <fgdc:endtime>000000000</fgdc:endtime>
            </fgdc:rngdates>
          </fgdc:timeinfo>
          <fgdc:current>unknown</fgdc:current>
        </lead:timeperd>
      - <lead:spdom>
        - <fgdc:bounding>
            <fgdc:westbc>-180.0</fgdc:westbc>
            <fgdc:eastbc>180.0</fgdc:eastbc>
            <fgdc:northbc>90.0</fgdc:northbc>
            <fgdc:southbc>-90.0</fgdc:southbc>
          </fgdc:bounding>
        </lead:spdom>
      </lead:idinfo>
    </lead:geospatial>
  - <lead:distinfo>
    - <fgdc:distrib>
      - <fgdc:cntinfo>
        - <fgdc:cntorgp>
            <fgdc:cntorg>UCAR/UNIDATA</fgdc:cntorg>
          </fgdc:cntorgp>
        - <fgdc:cntaddr>
            <fgdc:addrtype>mailing</fgdc:addrtype>
            <fgdc:city>Boulder</fgdc:city>
```


Fig. 2 Metadata from a LEAD file

We discuss the first approach in this paper and show experimental results of ingest, retrieval and purge times under the single digital object solution. Our ongoing work is to examine search costs and expressiveness under the two approaches.

The remainder of the paper is organized as follows. Section 2 gives an overview of Fedora (version 3.0 beta 1), the Fedora Digital Object Model and Fedora Object XML. Section 3 details the data sets that we used in our experiments. In section 4, we talk about our test bed and experimental goals. Section 5 details our approach related to our representation of a LEAD experiment as a single digital object model. In Section 6, we discuss our results for our experiments with modeling a LEAD

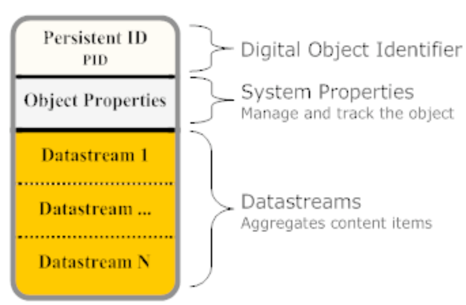

Fig. 3 Fedora Digital Object Model

experiment as a single digital object. In Section 7, we have our conclusions, which is followed by the future work section in Section 8.

2 FEDORA

Fedora is an extensible framework for the storage, management and dissemination of complex objects and the relationships among then. It is implemented as a web service. REST and SOAP interfaces are used to expose all aspects of the complex object architecture and its related management functions.

The Fedora Digital Object Model (Fig. 3[3]) [4] is a fundamental building block of the Content Model Architecture and all other Fedora-provided functionality. It is a generic digital object model that can be used to express different kinds of objects including documents, datasets, metadata, etc.

A Fedora digital object has three basic components:

1) PID
2) Object Properties
3) Datastream(s)

A PID is assigned for every digital object. Each PID is unique and persistent to that digital object.

Object properties in a digital object are a set of system-descriptive properties that are necessary to manage and track that object in a repository.

Datastreams represent data sources in a digital object. Fig. 4[4] provides a detailed view of the datastreams in the Fedora Digital Object Model. Every digital object will have a Dublin Core datastream (Fedora will create one if one is not provided). An Audit Trail datastream is maintained by the Fedora repository service to record all changes made to that object. This datastream is system managed and is not editable. The RELS-EXT datastream is an optional datastream, which may be used to detail relationships between digital objects. A digital object may have any number of additional custom datastreams, in addition to these reserved datastreams. Fedora datastreams have control groups, which pertain to the bytestream content. It can be defined as one of four types:

1) Internal XML Metadata

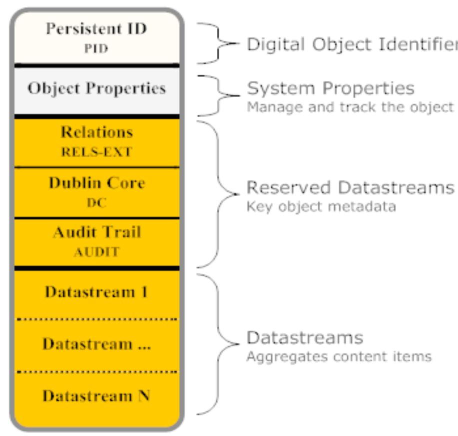

Fig. 4 Fedora Digital Object Model with detailed view of datastreams

```
<digitalObject PID="uniqueID">
      <!-- there are a set of core object properties -->
      <objectProperties>
            <property/>
            <property/>
            ...
      </objectProperties>
      <!-- there can be zero or more datastreams -->
      <datastream>
            <datastreamVersion/>
            <datastreamVersion/>
            ...
      </datastream>
      !-- there can be zero or more disseminators -->
      <disseminator>

            <disseminatorVersion/>
            <disseminatorVersion/>
            ...
      </disseminator>
</digitalObject>
```

Fig. 5 Sketch of FOXML

[3, 4] Fig. 3, 4 obtained from "The Fedora Digital Object Model (Fedora Release 3.0 Beta 1)" http://www.fedora-commons.org/documentation/3.0b1/userdocs/digitalobjects/objectModel.html

- Datastream will be stored as XML that is actually stored inline within the digital object XML file.

2) Managed Content
   - The datastream content will be stored in the Fedora repository. An internal identifier to that stream will be stored in the digital object XML.
3) External Referenced Content
   - The datastream content is stored outside of the Fedora repository and a URI to that datastream is stored in the digital object. Fedora mediates access to this content.
4) Redirect Referenced Content
   - The datastream content is stored outside of the Fedora repository and a URI to that datastream is also stored in the digital object. When access to that datastream is requested, Fedora redirects to that URI.

Digital objects are stored internally in a Fedora repository using the Fedora Object XML (FOXML) [5] format (Fig. 5[5]). FOXML is a simple XML format that directly expresses the Fedora digital object model. It is also used for ingesting and exporting objects to or from a Fedora repository.

3 Data Sets

We used various LEAD data in our experiments for comparison, in particular, three LEAD experiments, two LEAD files and two LEAD collections. The data that we used are as follows:

LEAD Experiments

- ADAS Experiment (5km, 12-hour)
  - Total size: ~5.0GB
  - Size of LEAD metadata: ~404.6kB
  - Consists of a total of 3 collections and 37 files.

  Comment: The ARPS Data Assimilation System (ADAS) experiment is executed as a workflow of 10 tasks that includes an execution of a 12-hour weather forecast computed over an analysis grid of resolution 5 km in the horizontal.

- NAM Experiment (5 km, 12-hour)
  - Total size: ~3.8GB
  - Size of LEAD metadata: ~396.1kB
  - Consists of a total of 3 collections and 37 files.

  Comment: The NAM (North American Model) experiment is a similar workflow to the ADAS experiment but uses NAM data as input. It also has 10 tasks.

[5] Fig. 5 obtained from "Introduction to Fedora Object XML (FOXML) (Fedora Release 3.0 Beta 1)" http://www.fedora-commons.org/documentation/3.0b1/userdocs/digitalobjects/introFOXML.html

- Data Mining Experiment
  - Total size: ~1.7MB
  - Size of LEAD metadata: ~101.1kB
  - Consists of a total of 3 collections and 12 files.

  Comment: This experiment is a workflow of 8 tasks that detects storms in NEXRAD Level II data using Algorithm Development and Mining System (ADAM) Storm Detection Algorithm.

LEAD Files

- ADAS File
  - Total size: ~ 927MB
  - Size of LEAD metadata: ~ 6.20kB
- namelist.input
  - Total size: ~ 8.44kB
  - Size of LEAD metadata: ~ 4.48kB

LEAD Collections

- Input Data Files Collection from ADAS Experiment (5 km, 12 hour)
  - Total size: ~ 1.22GB
  - Size of LEAD metadata: ~ 31.5kB
  - Consists of 6 files.
- Workflow Templates
  - Total size: ~ 6.3kB
  - Size of LEAD metadata: ~ 6.3 kB
  - Consists of 1 collection.

4 Test Bed & Experimental Setup

For our experimentation purposes, we used Fedora 3.0 beta 1 (released December 21, 2007), which was installed on a Dell PowerEdge 2850. This machine consists of two 3GHz Xeon processors with Hyper Threading and 1MB of cache each. A total of 4 virtual CPU's were allocated. This machine had 6GB of RAM, 139GB RAID 1 Array for its System Disk, 430GB RAID 5 Array for data, but had only a 100Mbps Ethernet card. This would prove to be a bottleneck during ingests operations. The operating system installed was Linux Red Hat 3.4.6-8.

In addition, we used MyLEAD agent/2.10.4, which is running on the production stack (tyr01) for retrieving LEAD data.

Our client for testing was written in Java and we ran it on Bitternut server (Dell PowerEdge 6950). All client communications between the MyLEAD agent and Fedora repository were done using SOAP calls.

The goal with our experimental evaluation is to better understand the total time it takes to transfer a LEAD experiment from MyLEAD into Fedora. We did this by measuring the retrieval time for LEAD objects from MyLEAD, the ingest time of LEAD objects into Fedora and the purge time of LEAD objects in Fedora.

For each LEAD object, the metadata of an object is first retrieved from MyLEAD agent and a FOXML document is formed based on the data retrieved. The FOXML document is then ingested into Fedora, where Fedora will store the FOXML document and fetch file contents from locations, which are provided in the FOXML document. After this, we proceed to purge the object we ingested. Time measurements for retrieval, ingest and purging were recorded during a single run. During our experiment, we ran 10 consecutive runs of retrieval, ingest and purging for each object. We then removed the data of our first run to eliminate the time spent by the client for setting up communication between MyLEAD agent and Fedora. We did this twice and merged both data sets.

We then plotted box and whisker plots for retrieval times, ingest times and purging times for the LEAD object. The mean of our data is represented by ◇ and their values are indicated on the graphs. ✗ and ○ are used to represent outliers.

5 Lead Experiment Modeled as a Single Digital Object

Our model of a LEAD experiment as a single digital object is very much similar to the Fedora Digital Object Model. This object consists of a unique PID, Object Properties, a Dublin Core datastream and a number of datastreams that correspond to LEAD experiment metadata, collection metadata, file metadata, as well as file content. So, for example, modeling our ADAS experiment (5 km, 12 hour) (3 collections and 37 files) yields a single digital object that consists of a unique PID, Object Properties, a Dublin Core datastream and a total of 78 datastreams for LEAD related data. We would have 1 datastream for the experiment metadata, 3 datastreams for the 3 collections metadata, 37 datastreams for file metadata and 37 datastreams for file content. Nevertheless, a LEAD experiment loses its hierarchy when modeled in this manner, since Fedora does not provide a way of expressing relationships between datastreams.

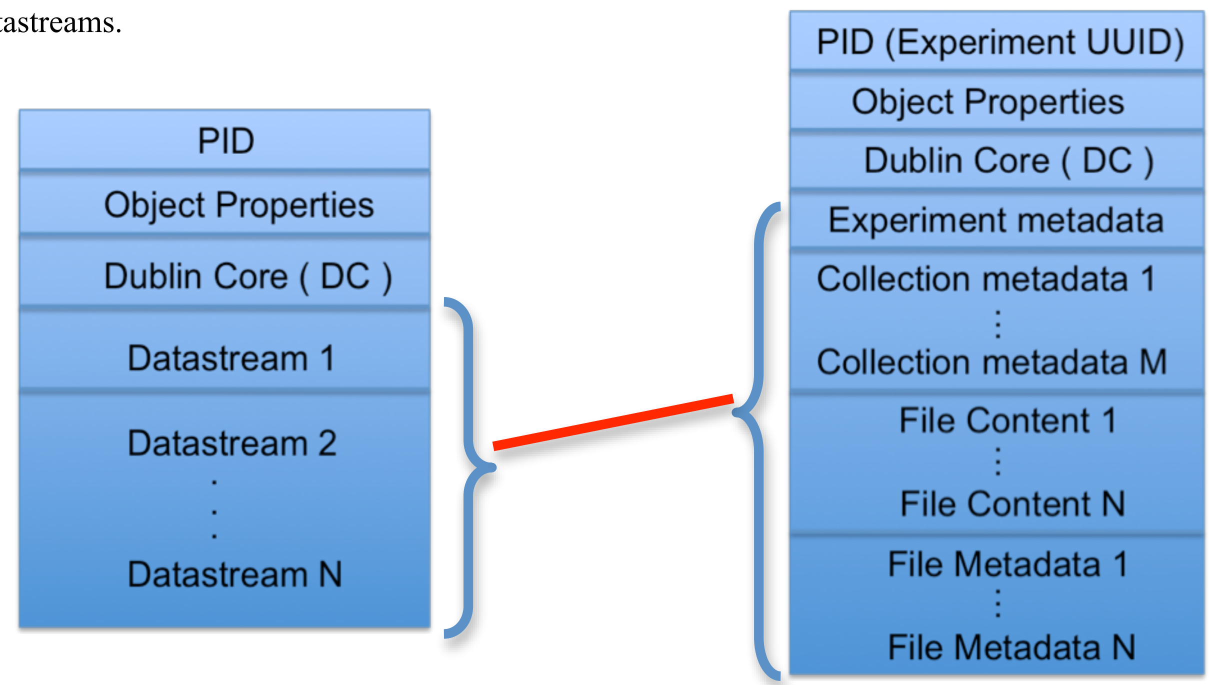

Fig. 6 Comparison between the Fedora Digital Object Model on left and our model of a LEAD Experiment modeled as a single digital object on right.

6 Results for a LEAD Experiment Modeled as Single Digital Object

Fig.7 shows the retrieval time for each LEAD object from MyLEAD agent. In this plot, we notice that although an ADAS experiment's metadata has a slightly larger size (404.6 kB) than the NAM experiment's metadata (396.1 kB), it takes a longer average retrieval time for a NAM experiment than for a ADAS experiment. Nevertheless, the difference in magnitude is small (< 500ms). Another thing worth noting is that the retrieval time for the metadata of the Workflow Templates collection (6.3 kB) is slightly longer than the ADAS (5 km, 12-hour) Input Data Files Collection (31.5 kB), and it has an outlier that takes about 5 times more than the average retrieval time.

Fig. 8 shows the time needed to ingest each LEAD object along with its

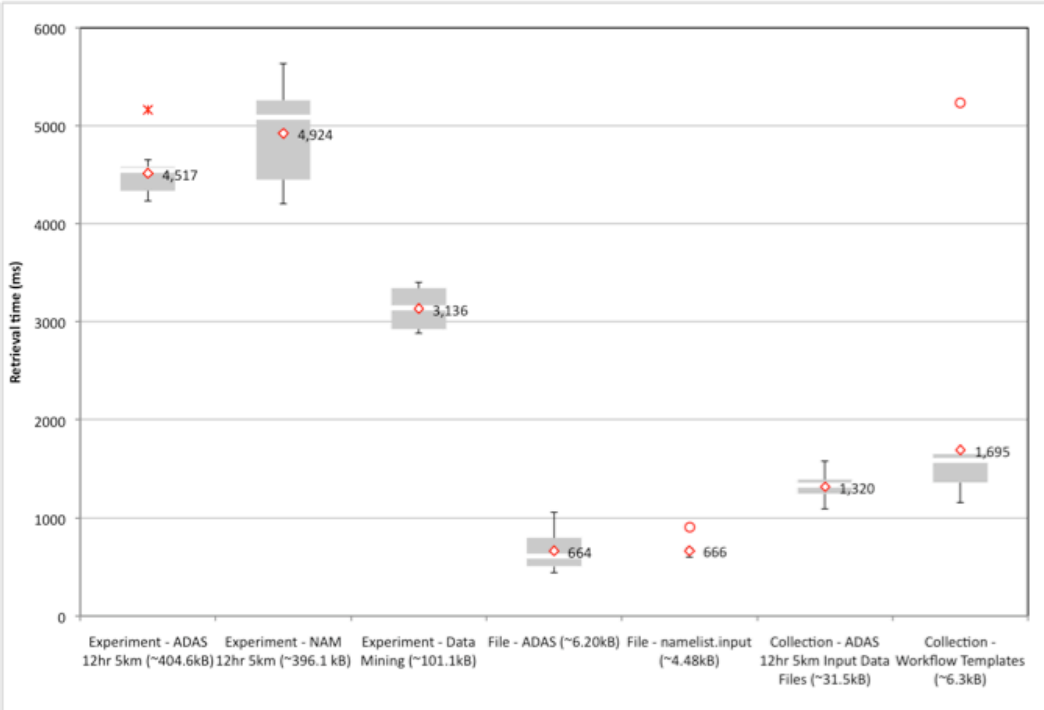

Fig. 7 Retrieval time

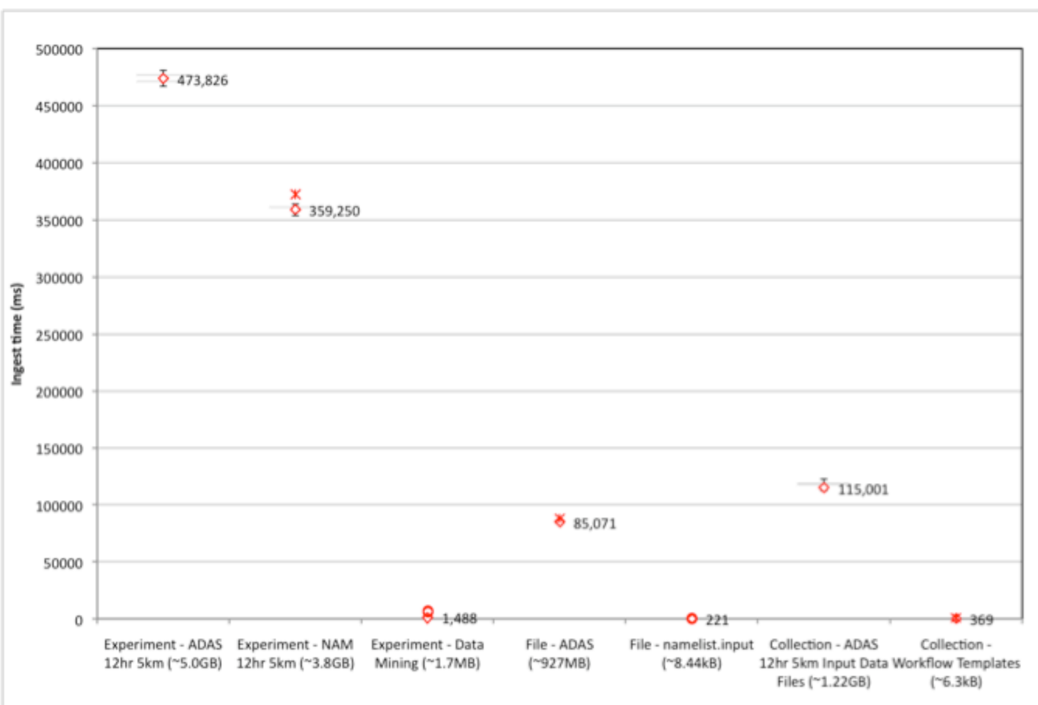

Fig. 8 Ingest time (Files stored in MyLEAD)

FOXML document. It is worth noting that due to a XML namespace bug in Fedora, we have to first write the FOXML document to disk and reference the document via a URL before it can be ingested. As can be seen, the ingest times of larger LEAD objects are really slow, such as the ADAS (5 km, 12-hour) experiment, the NAM (5 km, 12-hour) experiment and the ADAS file. This can be attributed to the 100Mbps Ethernet connection on the server hosting the Fedora repository. Since files are stored in MyLEAD, the connection becomes a bottleneck.

In order to prove that this is indeed the case, we went ahead and stored our test data on the server hosting the Fedora repository. We then ran our tests in the same fashion to measure the new ingest times.

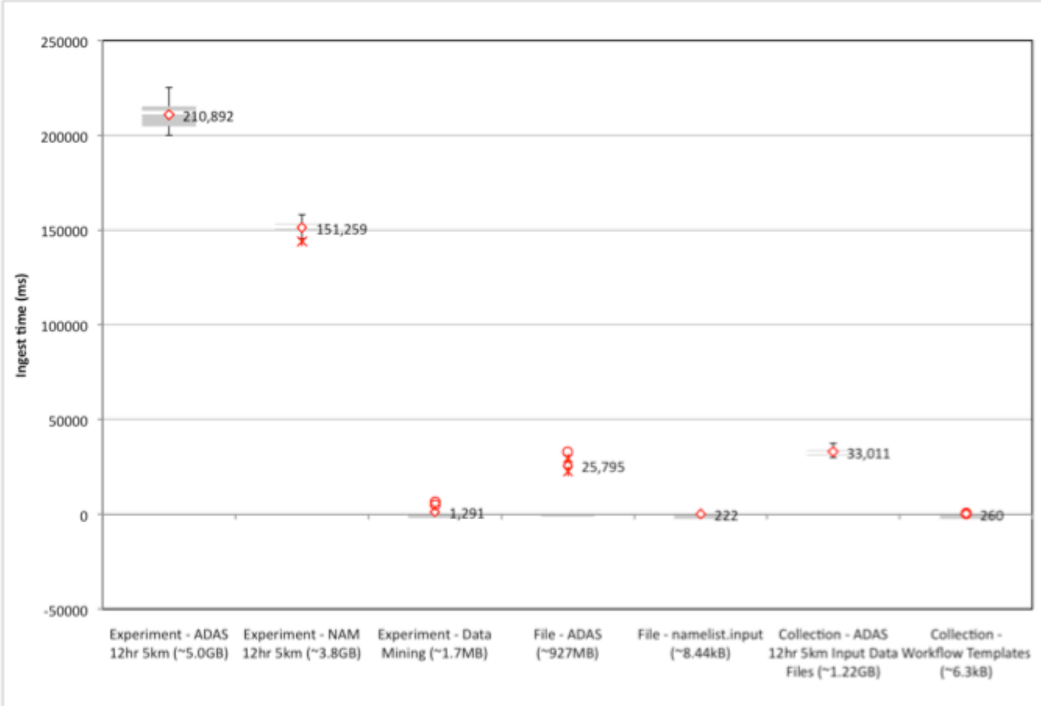

Fig. 9 Ingest time (Files stored on local host)

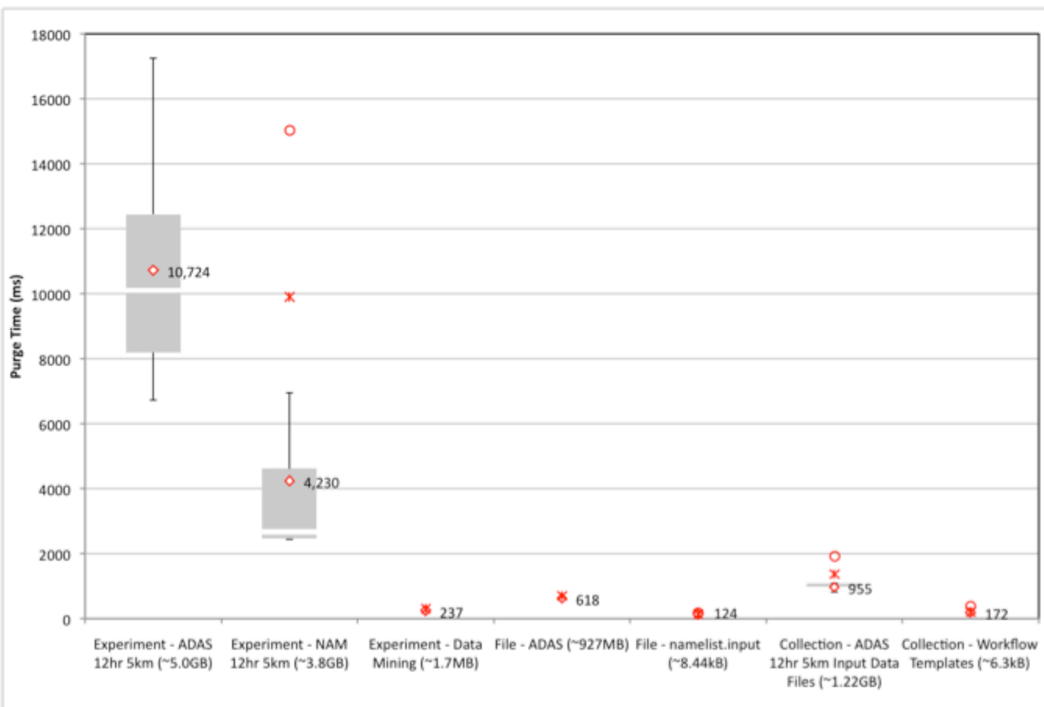

Fig. 10 Purge time

Fedora requires that files be fetched using a http connection, hence we are ensured that files are not being transferred directly through the file system.

Fig. 9 shows the new ingest times for the same LEAD objects and FOXML documents used in our previous ingest tests. As expected, ingest times are reduced significantly while using this approach for LEAD objects of large sizes. As can be seen, the average ingest times for an ADAS experiment drops from 473,826 ms to 210,892ms, which is a reduction of more than 50%. For LEAD objects of smaller sizes, this is not so evident. The average ingest times for our namelist.input file only had a 1ms decrease.

Fig. 10 shows that purge times are consistent with the sizes of LEAD objects. As these object sizes increases, so does their ingest times for their corresponding digital objects. In general, purge times are not that consistent, with outliers occurring frequently. The outliers are amplified as the sizes of objects increases. This is obvious for the NAM experiment (5 km, 12-hour) case. Purge times for the digital object of a NAM experiment (5km, 12-hour) are quite irregular; with one of its outliers almost four times longer than the average purge time.

7 CONCLUSIONS

Through our project, we notice that a LEAD experiment loses its initial hierarchy if modeled as a single digital object. Our experiments with retrieval time, ingest time and purge time have also given us a better idea of the time needed to perform these operations for different types of LEAD data. We also proved that ingest operations are heavily dependent on the network connections, since ingests are performed entirely through http connections.

8 FUTURE WORK

Fedora 3.0 beta 2 will soon be released. This release we hope will include fixes to bugs that we have encountered in our study, such as the XML namespace bug mentioned above. By using the RELS-EXT datastream and modeling a LEAD experiment as multiple digital objects, we should be able to solve the problem of a LEAD experiment losing its hierarchy when modeled as a single digital object. Further uses of the RELS-EXT datastream should also be investigated. Tools such as ORE provider capitalize on the use of RELS-EXT datastream to generate atom feeds. We should also look at modeling a LEAD science data product as a single data object, as well as modeling a collection of public forecast model products. Scientific data outside of LEAD should also be modeled. This will enable us to better understand how to model other kinds of scientific data.

REFERENCES

[1] K. Droegemeier, K. Brewster, M. Xue, D. Weber, D. Gannon, B. Plale, D. Reed, L. Ramakrishnan, J. Alameda, R. Wilhelmson, T. Baltzer, B. Domenico, D. Murray, A. Wlson, R. Clark, S. Yalda, S. Graves, R. Ramachandran, J. Rushing, E. Joseph, "Service-oriented environments for dynamically interacting with mesoscale weather", *Computing in Science and Engineering*, IEEE Computer Society Press and American Institute of Physics, Vol. 7, No. 6, pp. 12-29, 2005

[2] Beth Plale, Rahul Ramachandran, and Steve Tanner, Data Management Support for Adaptive Analysis and Prediction of the Atmosphere in LEAD, *22nd Conference on Interactive Information Processing Systems for Meteorology, Oceanography, and Hydrology (IIPS)*, January 2006.

[3] Fedora Open Source Repository Software: White Paper, October 2005 http://www.fedora.info/documents/WhitePaper/FedoraWhitePaper.pdf

[4] The Fedora Digital Object Model (Fedora Release 3.0 Beta 1) http://www.fedora-commons.org/documentation/3.0b1/userdocs/digitalobjects/objectModel.html

[5] Introduction to Fedora Object XML (FOXML) (Fedora Release 3.0 Beta 1) http://www.fedora-commons.org/documentation/3.0b1/userdocs/digitalobjects/introFOXML.html