Peer Reviewed

Title:
Mainstreaming Preservation through Slicing and Dicing of Digital Repositories: Investigating Alternative Service and Resource Options for ContextMiner Using Data Grid Technology

Author:
Lee, Christopher, University of North Carolina
Marciano, Richard, University of North Carolina
Hou, Chien-Yi, University of North Carolina
Shah, Chirag, University of North Carolina

Publication Date:
10-05-2009

Series:
iPRES 2009: the Sixth International Conference on Preservation of Digital Objects

Publication Info:
iPRES 2009: the Sixth International Conference on Preservation of Digital Objects, California Digital Library, UC Office of the President

Permalink:
http://www.escholarship.org/uc/item/9tw130cc

Multimedia URL:
http://www.cdlib.org/services/uc3/iPres/video.html?file=ipres/CamLee&title=Christopher%20Lee%3A%20Mainstreaming%20Preservation%20through%20Slicing%20and%20Dicing%20of%20Digital%20Repositories%3A%20Investigating%20Alternative%20Service%20and%20Resource%20Options%20for%20ContextMiner%20Using%20Data%20Grid%20Technology

Abstract:
A digital repository can be seen as a combination of services, resources, and policies. One of the fundamental design questions for digital repositories is how to break down the services and resources: who will have responsibility, where they will reside, and how they will interact. There is no single, optimal answer to this question. The most appropriate arrangement depends on many factors that vary across repository contexts and are very likely to change over time. This paper reports on our investigation and testing of various repository "slicing and dicing" scenarios, their potential benefits, and implications for implementation, administration, and service offerings. Vital considerations for each option (1) efficiencies of resource use, (2) management of dependencies across entities, and (3) the repository business model most appropriate to the participating organizations.

Supporting material:
Presentation

iPRES 2009

THE SIXTH INTERNATIONAL CONFERENCE ON THE PRESERVATION OF DIGITAL OBJECTS

Proceedings

October 5-6, 2009
Mission Bay Conference Center
San Francisco, California

California Digital Library

Mainstreaming Preservation through Slicing and Dicing of Digital Repositories: Investigating Alternative Service and Resource Options for *ContextMiner* Using Data Grid Technology

Christopher A. Lee, Richard Marciano, Chien-Yi Hou, Chirag Shah

School of Information and Library Science
University of North Carolina
Chapel Hill, NC 27599-3360
{callee, marciano, chienyi, chirags}@email.unc.edu

Abstract

A digital repository can be seen as a combination of services, resources, and policies. New online environments for digital collections are often created to provide a relatively focused set of services. If a digital collection environment proves useful over time, those responsible for managing the environment often begin to confront issues of interoperability, sustainability and scalability. A fundamental design question for digital repositories is how to break down the services and resources: who will have responsibility, where they will reside, and how they will interact. The most appropriate arrangement depends on many factors that vary across repository contexts and are likely to change over time. We report on efforts to integrate content and functionality of a feature-rich collecting environment (*ContextMiner*) into a robust data curation environment (iRODS). *ContextMiner* is a web-based service for building collections, through the execution and management of "campaigns" (i.e. sets of queries and parameters to harvest content). iRODS (integrated Rule-Oriented Data System), is adaptive policy-driven data grid middleware, which addresses aspects of growth, evolution, openness, and closure – fundamental requirements for digital preservation. This paper reports on our investigation of various repository "slicing and dicing" scenarios, their potential benefits, and implications for implementation, administration, and service offerings.

Introduction and Motivation

A digital repository can be seen as a combination of services, resources (required to carry out those services and supported by the services), and policies that determine how the services should be implemented. No two repositories will have the exact same services, resources or policies. New innovative online environments for digital collections are often created in order to provide a relatively focused set of services (e.g. management and presentation of a specific type of digitized materials; author submission and annotation of pre-print articles; harvesting and dissemination of content from the Web). For purposes of simplicity, services and resources are often co-located under the control of a single entity. If a digital collection environment proves useful over time, those responsible for managing the environment often begin to confront issues of interoperability, sustainability and scalability. In short, they move from developing and supporting a specialized set of tools to developing and supporting a long-term digital repository. The NSF Cyberinfrastructure Council (NCC) points out that "research collections [originally developed to serve only short-term work group needs] may evolve over time to become resource and/or reference collections," which have longer periods of retention and thus require higher long-term stewardship commitments (2006). Making this transition successfully is one of the main ways in which digital preservation will be "mainstreamed."

One of the fundamental design questions for digital repositories is how to break down the services and resources: who will have responsibility, where they will reside, and how they will interact (Sierman, Van Diessen and Lee 2008). There is no single, optimal answer to this question. The most appropriate arrangement depends on many factors that vary across repository contexts and are very likely to change over time. Not only is the external environment of technology and users subject to change, but so are the services, resources and policies of the repository itself. It is, therefore, desirable to explore multiple options for "slicing and dicing" a given repository, in order to (1) increase the chances of settling on an arrangement that is appropriate to the given context, and (2) formulate long-term strategies that are amenable and robust to changes in the arrangement over time. When "services make the repository" (Chavez et al. 2007), moving raw data from one location to another will often not be sufficient.

The NCC offers the following principle: "Provide a framework that will sustain reliable, stable resources and enable the integration of new technologies and research developments with a minimum of disruption to users." A CI must "evolve" over time (2006). Long-term preservation will be served through "robust design" (Hargadon and Douglas 2001), which is effective in the short-term but also sufficiently flexible to remain

effective in a wide range of possible future contexts. Limiting the interdependencies between subsystems can also make a design more robust against disruptions from the environment (Simon 1962).

Long-term repositories should not be locked into one particular combination of hardware and software, but should instead make extensive use of redundancy (Maniatis et al. 2005); diversity in both technological approaches (Rosenthal et al. 2005) and business models (NSF Cyberinfrastructure Council 2006); abstraction; virtualization (Marciano and Moore 2005); detailed descriptive and administrative metadata beyond that which is required for immediate use; and the development and adoption of open standards in way that is attentive to the need for flexibility (Hanseth, Monteiro, and Hatling 1996; Monteiro 1998; Egyedi 2001).

System evolution, sustainability and innovation can also be greatly facilitated through modularity (Langlois and Robertson 1992). Modular design "creates a new set of modular operators, which open new pathways of development for the design as a whole." (Baldwin and Clark 2000) Curators of digital collections can pre-empt future costly and problematic system migration efforts by integrating collections into environments specifically designed to support long-term preservation, scalability and interoperability (Aschenbrenner et al. 2008).

We report on an integration of content and functionality of a feature-rich collecting environment (ContextMiner) into a robust data curation environment (iRODS). This work contributes to the emergence of policy-based digital preservation environments, which will be essential for the development of a robust cyberinfrastructure to support current and future users of digital resources (Beagrie et al. 2008; Berman 2008) We hope to illustrate options for growth in frameworks such as ContextMiner, when repositories reach a critical mass that requires re-architecting with regards to storage, archiving, and scalability and wish to make the research findings generalizable to other classes of repositories.

ContextMiner

ContextMiner is a web-based service for building collections, through the execution and management of "campaigns" (i.e. sets of associated queries and parameters to harvest content over time). Campaigns can collect information from a variety of sources, including blogs, YouTube, Flickr, Twitter, and the open Web. *ContextMiner* takes advantage of various site-specific APIs to collect specific data elements. Figures 1-3 provide screenshots of a collecting campaign, emphasizing data collected from YouTube.

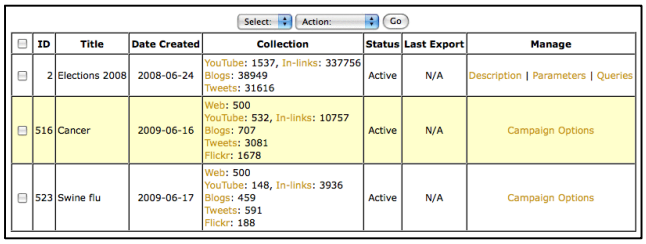

Figure 4 - Viewing Collecting Campaigns in ContextMiner

Figure 5 - Listing Items from YouTube within a Collecting Campaign in ContextMiner

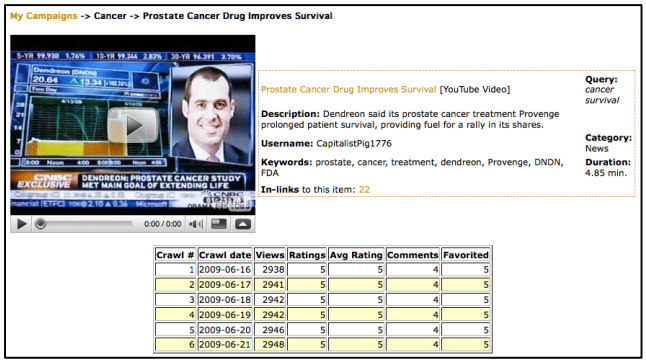

Figure 6 – Viewing Detailed Metadata for a Video from YouTube

When creating collecting campaigns, users of *ContextMiner* can specify a set of queries and associated parameters, including how often the queries are executed, the number of results to harvest, and the primary use environment hosts (web sites) that should be queried.

We use the term "crawl" to indicate one instance of executing the following two sets of activities: 1) submitting all queries associated with a campaign and then collecting data from a specified number (up to 1000) of results fore each query based on YouTube's search option of sorting by "relevance"; and 2) collecting updated dynamic metadata for each video that has been "discovered" through any instance of step 1. When a video is first discovered within a crawl, *ContextMiner*

collects static metadata (video ID, title, contributor, date added, description and tags) and dynamic metadata (number of views, ratings, number of honors, and number of times favorited) associated with the video. Then the video is added to a list of "discovered" videos associated with each query. In step 2 of subsequent crawls, the dynamic metadata for each video is collected again. Each time data are captured for a video, a time-stamp is recorded. By including the YouTube ID within the database records for each video, *ContextMiner* allows a user to track the rank of a given video across time independently for each query and identify multiple instances of the same video within or across campaigns.

The VidArch project has developed and used the *ContextMiner* framework and services for harvesting YouTube videos and associated contextual information on a variety of topics, including energy, epidemics, health, natural disasters, truth commissions, and the 2008 U.S. presidential election (Shah and Marchionini 2007; Capra et al. 2008; Marchionini et al 2009). *ContextMiner* runs each of the queries on YouTube every day, and it extracts the top 100 results, based on YouTube's relevance ranking.

After creating and initiating a collecting campaign, the user of *ContextMiner* can carry out various campaign maintenance activities, including changing the campaign description, queries, and some of the parameters; pausing, resuming, or deleting entire campaign or specific queries; and adding new queries. *ContextMiner* supports both information discovery and selection for purposes of collection development. The curator of a collection may determine that only a subset of the items identified from crawls warrant ingest into a repository. In order to support such determinations, users of *ContextMiner* can apply judgments (relevant, non-relevant, neutral) to crawled items and also delete those items that he/she does not wish to retain.

In July 2008, a public beta of ContextMiner was released, allowing anyone to run similar crawls. There are now nearly 300 users, and this population continues to grow. Users have created more than 600 campaigns, and collected millions of digital objects. These campaigns and their uses span a large spectrum. One example is the cancer research team at University of Wisconsin at Madison that has been using *ContextMiner* to run a campaign on how people produce and consume cancer-related information in digital media sites, such as YouTube and Flickr. They have collected more than 700 YouTube videos and nearly 40,000 images from Flickr, along with associated contextual information.

Growth Pains of *ContextMiner*

The current implementation – based on a single MySQL database and associated code – has served its intended purposes very well, but it is not a scalable or sustainable basis for offering wide-scale collecting services in support of the diverse array of potential users and use cases. Below are several major challenges and opportunities for the future of *ContextMiner*.

Storage

All the data collected by running queries and capturing associated metadata are currently populating a single MySQL database on the same server. Because *ContextMiner* continues to run all the processes associated with a campaign, the data from a campaign continues to grow over time. Given that there are now about 300 users, running more than 600 campaigns with more than 1000 queries almost every day, this creates an increasing challenge for processing and storage.

Collaboration

Professionals responsible for the collection and curation of digital resources can benefit from collaborating in their efforts. Several users of *ContextMiner* have expressed a desire to collaborate with other users of the system. This could involve sharing campaign queries and parameters; data and metadata collected within campaigns; relevance judgments; and humanly-generated metadata. Collaborative filtering, tagging and other interactive tools could also allow users to further collaborate in their application of selection judgments and determinations of whether and how many copies of items to ingest into their respective repositories.

Secure Sharing

In order to support various collaboration and sharing scenarios, they must have associated interfaces, storage facilitators, and services. This could be supported by existing software for authentication, access permissions and control of profile information.

Passive Users

Not all users of *ContextMiner* visit the site or their campaigns frequently. Some have simply created their campaigns and let *ContextMiner* run the automated processes that can keep collecting the data for them. Figures 4-6 show the number of user accounts, campaigns and logins to *ContextMiner*. New users continue to create many new campaigns, but they are not revisiting their created campaigns with similar frequency.

It would be beneficial to create and implement policies to handle such *passive users*. For instance, a policy could specify that, when a user has not logged into *ContextMiner* for more than a month, her campaigns will be paused, she will receive an email notification, and her campaigns will be deleted after another month has passed.

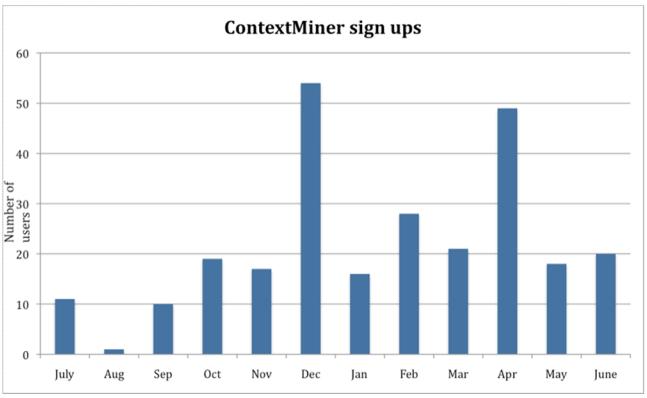

Figure 4 - ContextMiner sign ups in the first year (starting July 2008)

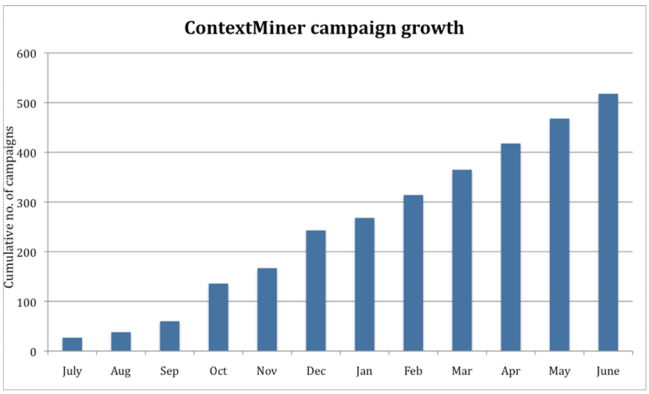

Figure 7 - Cumulative Number of Campaigns in ContextMiner (starting July 2008)

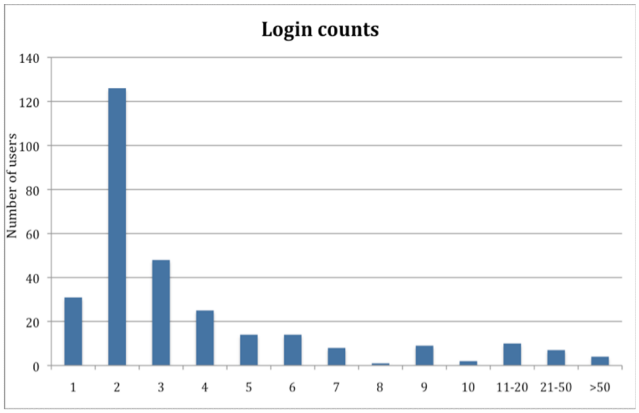

Figure 8 - Number of Logins to ContextMiner User Accounts

Preservation and Sustainability

The users of *ContextMiner* may desire long-term storage and replication of the content generated from their campaigns. This creates technical and policy-related challenges. The current *ContextMiner* user interface does not directly support users' downloading of videos from YouTube; it captures, generates, manages and hosts metadata associated with videos. It is not possible for the School of Information and Library Science at the University of North Carolina, Chapel Hill (the host of *ContextMiner*) to take on the role of collecting and preserving collections of content identified in all *ContextMiner* campaigns. However, it is very appropriate to provide (1) direct hooks into software that can be used to download YouTube videos, blog pages, or other content associated with campaigns, and (2) interfaces from *ContextMiner* to other storage and repository environments (e.g. data grids, Fedora or DSpace instances) where content can be managed over time.

ContextMiner allows users to specify various parameters for the scheduling of campaigns, queries and crawls. It also provides a basic data export function, which allows users to generate copies of their campaigns' data as either Extensible Markup Language (XML) or comma-separated values (CSV). There is great potential for combining these two features in various ways in order to allow *ContextMiner* users to ingest and replicate campaign-related data based on designated trigger events or pre-defined schedules.

integrated Rule-Oriented Data System (iRODS)

iRODS (integrated Rule-Oriented Data System), is adaptive policy-driven data grid middleware, which addresses aspects of growth, evolution, openness, and closure – fundamental requirements for digital preservation (Thibodeau 2008). iRODS currently scales to hundreds of millions of files, tens of thousands of users, and petabytes of data. It operates in a highly distributed environment with heterogeneous storage resources and allows for growth through federation. iRODS supports evolution through the virtualization of the underlying technology and supports changing business requirements through customization of repository behaviors. It supports openness through treatment of content that is agnostic to data type.

iRODS is designed to support data virtualization (storage system independence), trust virtualization (administration independence), and management virtualization (policy independence). This makes it a unique platform to study repository integration. It allows resources, services and policies to be separated or combined in many different ways. The coupling of iRODS with other repository software can create both new efficiencies and new types of repository services.

iRODS can be instrumented with policies that support the management of the lifecycle of digital assets. One key feature is the automation of policy enforcement across distributed data that have been organized into a shared collection.

The rule engine schedules and executes rules which are expressed as the following sequence: *event, condition, action set*, and *recovery procedure*. Once the condition stands, the rule will be triggered to execute the action set. An action set includes a chain of micro-services or rules. Micro-services are small procedures/functions that perform specific tasks.

Slicing and Dicing Options

ContextMiner provides an interface for users to specify the criteria to collect and crawl web content and YouTube videos. iRODS is a very flexible environment that can potentially support or directly implement various aspects of *ContextMiner*. We have been investigating various repository "slicing and dicing" scenarios, their potential benefits, and implications for implementation, administration, and service offerings.

1. Transfer of Data

A relatively simple scenario involves moving data (called "persistent state information" in iRODS terminology) from *ContextMiner* to iRODS, in order to take advantage of the scalability, data integrity and replication features of iRODS. This transfer could be carried out only once or periodically, based on trigger events or pre-defined schedules. This approach allows the initial application and all its associated scripts to reside in its natural habitat, but provides the capability of now issuing metadata queries from the iRODS repository itself directly to the iRODS the metadata catalog of iRODS (called iCAT). A set of data grid services (rules) is added to the ruleset, where the video content is managed.

An important consideration is what data to include in the transfer. The VidArch collection at UNC, for example, includes (1) video files harvested from YouTube, (2) static metadata for each video, collected from YouTube the first time the video was encountered, and (3) extensive metadata about both the individual videos and the collecting campaigns over time. We have moved copies of all three types of data from the VidArch project into iRODS. This allows delegation to the data grid and management in a scalable, distributed environment, with automated management through iRODS preservation rules. However, one might instead choose to transfer only one or two of the above categories of data to iRODS.

Our initial transfer into iRODS effectively treated the data as a large, undifferentiated bitstream, i.e. it did not break the data up into distinct data elements. Many further advantages can be gained within iRODS by mapping the fields from the MySQL database of *ContextMiner* into the internal data structures of iRODS, known as attribute, value, units (AVUs). We have been investigating such a low-level transfer of data through two different mechanisms: export of XML from *ContextMiner* and import into iRODS; and the Rule-oriented Database Access (RDA) system, which provides rule-driven access to arbitrary databases through iRODS.

Rather than waiting to receive data submissions, iRODS could instead take a more active role in the transfer. iRODS could query and obtain data from the *ContextMiner* database. One could create a "collection" within iRODS for each campaign and store videos with associated metadata under this collection. iRODS could then query the *ContextMiner* database periodically (e.g. once a week). If iRODS discovered a new campaign within *ContextMiner*, it could create a new collection for that campaign and ingest the associated metadata.

Data transfer also opens up numerous arrangements in which iRODS mediates storage of the data and metadata in different places, based on who is assigned responsibility for storage services (e.g. the collecting institution, a consortial data center, a private-sector storage provider).

2. Transfer of Features and Functions from ContextMiner to iRODS

More complex scenarios involve the use of iRODS as a middleware layer to move, federate, and further enhance the collection building and user services currently offered by *ContextMiner*. For example, data grid technology has been used to manage a large collection of crawled web resources, with Fedora serving as the basis for end-user access to the collection (Marciano, Moore and Zhu 2009). Moore and Zhu (2008) have also used iRODS to implement policy-driven web crawls. In the case of *ContextMiner*, iRODS rules can be used to execute continuous web harvesting after a collecting campaign has been initiated, using the harvesting software that is considered most appropriate. iRODS rules can also implement user account actions based on customized policies (e.g. disabling crawls after a given period of inactivity).

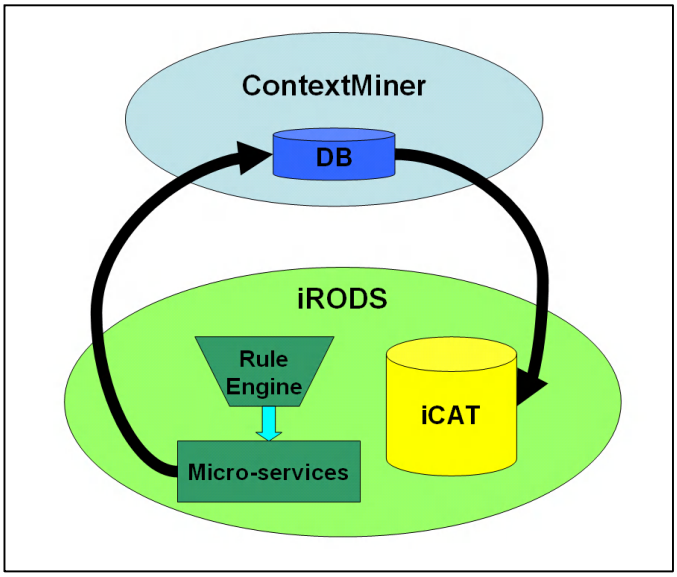

Figure 9 - A Combined ContextMiner/iRODS Architecture

Some of the experimentation we have carried out so far covers the following. In order to connect *ContextMiner* and iRODS, a first approach was to design a set of rules to retrieve information from *ContextMiner* into iCAT. Figure 7 illustrates the combined architecture.

The set of rules we use to retrieve information from *ContextMiner* can be divided into two groups. The first group is for atomic rules. When putting a file into iRODS, a matching rule will be triggered immediately. The rule will grab information associated with the object just uploaded to iRODS and ingest it to iCAT as metadata. We will use part of the result to define our second group of periodic rules. These rules usually run asynchronously, such as once a week or once a month. In *ContextMiner*, users specify the frequency of crawls. We can use this information to design rules to synchronize the information between *ContextMiner* and iCAT. Beyond the above rules, we can also design additional rules to specify preservation policies within iRODS.

Usage Scenario

Consider a case in which a user wants to use *ContextMiner* to collect videos from YouTube related to the 2008 U.S. presidential election and preserve the videos within iRODS. He creates a campaign, with a set of associated queries to be issued to YouTube every Sunday after the initial query.

Rule 1 in the iRODS rule base can be used to get the initial set of metadata (as XML exported from *ContextMiner*), parse the metadata, creation collections within iRODS for each collecting campaign represented in the ContextMiner data, and ingest the metadata into iCAT as user-defined metadata. **Rule 2** is designed to download the videos associated with a collecting campaign, once every 7 days. **Rule 3** is designed to query the *ContextMiner* database, in order to get any new metadata associated with the videos, once every 7 days.

Rule 1 – Get, Parse and Ingest Initial Metadata into iRODS Collections:
acPostProcForPut | $objPath like /ContextMiner/* | msiParseContextMinerForCollection($objName,result)#msiCreateCampaign(result) | nop#RollBack

Rule 2 – Download Videos Once/Week:
getVideoRule || delayExec(<EF>7d</EF>, misParseContextMinerForVideo(XMLfile,result)#msiGetVidoe(result),nop#nop) | nop

Rule 3 – Update Metadata Once/Week:
updateVideoMetadata || delayExec(<EF>7d</EF>, misParseContextMinerForMetadata(XMLfile,result)#msiIngestMetadata(result),nop#RollBack) | nop

Two further rules relate to actions that take place entirely within the context of the iRODS data grid (i.e. do not involve interaction between *ContextMiner* and iRODS). **Rule 4** changes permissions so that data are available to the public. **Rule 5** makes one backup copy of the data.

Rule 4 – Make Data Available to the Public:
acPostPut | $objPath like */ContextMiner/** | msiModifyACL(public) | RollBack

Rule 5 – Replicate Data Once for Backup:
acPostProcForPut | $objPath like /ContextMiner/* | DelayExec(<PLUSET>1h</PLUSET>, msiReplicate($objName, newResource),nop) | nop

Rules 1 and 4 are atomic rules which act immediately upon being triggered. **Rules 2 and 3** are periodic rules which run every week. **Rule 5** is a deferred rule which runs one hour after being triggered.

The small set of rules provided above are intended to illustrate a few significant actions that one would be likely to perform on the ContextMiner data within iRODS. The scenarios we have been considering involve a more extensive set of rules.

Conclusions and Future Directions

This paper reports on our early efforts to explore slicing and dicing options for ContextMiner, as an example of internally complex collection environment. Further investigations should consider the following considerations for each option: (1) efficiencies of resource use, (2) management of dependencies across entities, and (3) the repository business model most appropriate to the participating organizations.

The issues and strategies explored in this paper have major relevance beyond the specific case of *ContextMiner*. Many collection building and collection management environments have reached considerable sophistication and internal complexity. However, they are often not designed to support significant shifts in scope, scale or underlying computing platforms. Members of the DICE group have been approached by various communities associated with such collection environments, who wish to integrate their collection management services with underlying scalable storage services and emerging preservation services.

Cross-repository integration frameworks are being researched to respond to the challenges of the lifecycle of repository spaces, where required services can be delegated to underlying cyberinfrastructure, and integration prolongs the life of the initial repository.

This paper illustrates initial experimentation and mechanism for automated management of both metadata and content through rule-based policy-driven mechanisms. This work informs the Distributed Custodial Archival Preservation Environments (DCAPE) project, which is funded by the National Historical

Publications and Records Commission (NHPRC). DCAPE is developing a ruleset of preservation services for state and university archives. The *ContextMiner*-iRODS integration effort is helping us to identify additional rules that may be applicable to transfer of data or functionality between other collecting environments

The research summarized in this paper has also highlighted the potential value of incorporating hooks directly from the user interfaces of repository and collection management environments into iRODS. By adding a few additional toggles, check boxes and text entry boxes to the *ContextMiner* interface, for example, one could allow the user to establish, schedule or invoke numerous rules through iRODS. These could include choices such as "replicate my campaign data X times in Y locations," "verify the integrity of my campaign data by running a checksum every X days," "notify me through email if my campaigns are about to be disabled," "pause my campaign if it grows beyond X bytes," or "every X hours, harvest the blog pages identified in my campaign using wget and store the videos in the following Y locations."

The user could apply such settings without having to master iRODS rule syntax or command-line skills. The potential for rule-oriented data curation will be greatly advanced by the development of user interfaces – for both repository professionals and parties who are submitted content – that can define and enact rules, while hiding many of the implementation details.

Acknowledgements

This work has been supported by grants from the National Science Foundation for the VidArch Project (#IIS 0455970 DigArch Program), and the National Historical Publications and Records Commission for the Distributed Custodial Archival Preservation Environments (DCAPE) project (NAR08-RE-10010-08).

References

Aschenbrenner, A.; Blanke, T.; Flanders, D.; Hedges, M.; and O'Steen, B. 2008. The Future of Repositories? Patterns for (Cross-)Repository Architectures. *D-Lib Magazine* 14(11/12).

Baldwin, C.Y. and Clark, K.B. 2000. *Design Rules. Vol. 1: The Power of Modularity*. Cambridge, MA: MIT Press.

Beagrie, N.; Semple, N.; Williams, P.; and Wright, R. 2008. Digital Preservation Study, Part 1: Final Report October 2008. Charles Beagrie Limited.

Berman, F. 2008. Got Data? A Guide to Data Preservation in the Information Age. *Communications of the ACM* 51(12): 50-56.

Capra, R.; Lee, C.A.; Marchionini, G.; Russell, T.; Shah, C.; and Stutzman, F. 2008. Selection of Context Scoping for Digital Video Collections: An Investigation of Youtube and Blogs. In Proceedings of the 8th ACM/IEEE Joint Conference on Digital Libraries, 211-20. New York, NY: ACM Press.

Chavez, R.; Crane, G.; Sauer, A.; Babeu, A.; Packel, A.; and Weaver, G. 2007. Services Make the Repository. *Journal of Digital Information* 8(2).

ContextMiner. http://www.contextminer.org

Distributed Custodial Archival Preservation Environments (DCAPE) project. http://dcape.org

Egyedi, T. 2001. Infrastructure Flexibility Created by Standardized Gateways: The Cases of XML and the ISO Container. *Knowledge, Technology & Policy* 14(3): 41-54.

Hanseth, O,; Monteiro, E.; and Hatling, M. 1996. Developing Information Infrastructure Standards: The Tension between Standardisation and Flexibility. *Science, Technology and Human Values* 21(4): 407-26

Hargadon, A.B.; and Douglas, Y. 2001. When Innovations Meet Institutions: Edison and the Design of the Electric Light. *Administrative Science Quarterly* 46(3): 476-501.

Langlois, R.N.; and Robertson, P.L. 1992. Networks and Innovation in a Modular System: Lessons from the Microcomputer and Stereo Component Industries. *Research Policy* 21(4): 297-313.

Maniatis, P., et al. 2005. The LOCKSS peer-to-peer digital preservation system. *ACM Transactions on Computer Systems* 23(1): 2-50.

Marchionini, G.; Shah, C.; Lee, C.A.; and Capra, R. 2009. Query Parameters for Harvesting Digital Video and Associated Contextual Information. In *Proceedings of the 9th ACM/IEEE-CS Joint Conference on Digital Libraries*, 77-86. New York, NY: ACM Press.

Marciano, R.; and Moore, R. 2005. Technologies for Preservation. In *Managing Electronic Records*, 81-100. London: Facet Publishing.

Marciano, R.; Moore, R.; and Zhu, B. 2009. Enabling Inter-Repository Access Management between iRODS and Fedora. Presented at the 4th International Conference on Open Repositories, Atlanta, Georgia, May 18-21, 2009.

Monteiro, E. 1998. Scaling Information Infrastructure: The Case of Next Generation IP in Internet. *The Information Society* 14(3): 229-45

Moore, R. and Zhu, B. Archiving Websites with iRODS: Ford.com Project. Presented at the Society of American Archivists Annual Meeting, San Francisco, California, August 26-30, 2008.

NSF Cyberinfrastructure Council. 2006. NSF's Cyberinfrastructure Vision for 21st Century Discovery. National Science Foundation.

Rosenthal, D.S.H., et al. 2005. Requirements for Digital Preservation Systems: A Bottom-up Approach. *D-Lib Magazine* 11(11).

Shah, C., and Marchionini, G. 2007. Preserving 2008 US Presidential Election Videos. Presented at the International Web Archiving Workshop (IWAW), Vancouver, BC, Canada.

Sieman, B.; Van Diessen, R. and Lee, C.A. 2008. Component Business Model for Digital Repositories. In Proceedings of the Fifth International Conference on Digital Preservation (iPres), London, England, September 29-30, 2008.

Simon, H.A. 1962. The Architecture of Complexity. *Proceedings of the American Philosophical Society* 106: 467-82.

Thibodeau, K. 2008. Architectural Issues in Preservation. Sun Preservation and Archiving Special Interest Group meeting. (Baltimore, November 20, 2008).