Policy-driven ingest – a Graphical User Interface to iRODS

Mike C. Conway, Jewel H. Ward, Antoine de Torcy, Hao Xu, Arcot Rajasekar, Reagan W. Moore; DICE; Chapel Hill, North Carolina

Abstract

A trusted digital repository requires the enforcement and validation of high-level policies. Within the integrated Rule-Oriented Data-management System (iRODS), these policies may be encoded as machine-actionable rules that control the execution of workflows at each storage location. Policies that manage the objects in the repository are augmented with policies to manage ingest and extraction of required metadata. This paper describes a system that provides a graphical environment for policy definition that specifies policy elements as modules, composes policies from these elements, and monitors the application of policies at ingest.

Introduction

The long term preservation of digital information is a challenging endeavor. Frameworks such as the Open Archival Information System [OAIS] Reference Model [1] have provided a useful conceptual model of the digital objects, metadata, and processes that characterize a digital repository. Systems such as Fedora [2] and DSpace [3] have adopted the OAIS model and vocabulary, creating platforms that manage the life-cycle of digital objects.

In order to ensure long-term preservation, repositories must exhibit the qualities of a "trusted digital repository". The RLG-OCLC working group defined a trusted digital repository as one whose mission is "to provide reliable, long-term access to managed digital resources to its designated community, now and in the future" [4]. Beyond compliance with the OAIS Reference Model, the RLG-OCLC identified many attributes of a trusted digital repository, including enterprise level, archive level, collection level, and item level policies. Examples included regulatory requirements, archive management framework, collection deposition process, and item format. The ISO MOIMS-RAC effort has developed a formal ISO standard that specifies criteria for the audit and certification of digital repositories [5]. The criteria cover organizational infrastructure, digital object management, and technical infrastructure. Approximately 100 policies can be extracted from the criteria, of which about half can be turned into machine-actionable rules.

Background

The theory of digital preservation has described a "minimal set of preservation policies that are needed to implement management policies", as well as a "minimal set of preservation metadata needed to evaluate assessment criteria" [6]. These preservation policies, and the metadata generated by application of such policies, have been encoded as computer-actionable rules within systems such as iRODS.

iRODS (the Integrated Rule-Oriented Data System), an open-source, policy-driven data repository, "manages a highly controlled collection of distributed digital objects, while enforcing user-defined Management Policies across the multiple storage locations" [7]. The iRODS rule engine has provided a model for encoding component services as micro-services, written in the C language. The rule engine has used a specialized language to chain these micro-services together into a workflow, along with execution conditions and error recovery processing. These rules have been bound to events within the data grid, such as "after ingest of a file", or "before dissemination of a file". Rules can also be run on a periodic, or delayed execution basis [8]. Through the configuration of the rule-base within iRODS, it has been possible to explicitly specify policies and maintain the necessary metadata attributes that verify the correct application of specified policy.

The PLEDGE project examined the policies, rules, and preservation metadata used by SRB and DSpace. PLEDGE researchers investigated how the trustworthiness of a repository could be automatically verified by validating assessment criteria generated by the application of rules within a repository. In going through this mapping process, they observed the difficulty in mapping assessment criteria to policies directly, and found it useful to map the policies to preservation capabilities in the underlying repository, and then map from these capabilities to the underlying rules [9]. It is this idea of developing preservation capabilities to meet the criteria of a trusted digital repository, and mapping these capabilities into computer-actionable rules enforceable by iRODS that motivates this paper.

While building on this idea of mapping from high-level policies to preservation capabilities, and from preservation capabilities to computer actionable rules, we saw a need to develop abstractions that can bridge the gap between high-level policies and assessment criteria, and computer-actionable rules and preservation metadata. The purpose of this project has been to research this middle ground with prototype interfaces oriented towards archivists. Collectively, this effort is termed "Arch", and it is a system that will "allow the composition of policies and verification of policy enforcement by the archivist" [10].

Method

The Arch project demonstrated several aspects of a system that would allow the specification and verification of high-level policies by archivists. This work involved the development of graphical interfaces for individuals, who have various roles in the system, as well as software abstraction layers to accomplish the mapping between high-level policy and computer-actionable rules.

Abstracting Policy

Important abstractions we investigated within Arch included:

- The abstraction of preservation environment capabilities into generalized policy components.
- The composition of policy components into policy templates.
- The customization and binding of policy templates to record collections.

- The dynamic resolution and activation of policies based on life-cycle events, or periodic evaluation of preservation metadata.

The policies of the iRODS preservation environment have existed as rules that control the execution of micro-services. The rules are encoded in the *core.irb* file, where they can be customized and applied to various event hooks (policy enforcement points) within iRODS. iRODS maintains a wide range of system metadata and audit trail data that reflects the application of specified policy. Arch researchers proposed that these rules could be reified, essentially moving from fine-grained encoding of policy to coarse-grained preservation capabilities, where customization would be accomplished through the setting of flags and input parameters. In effect, a policy could be characterized by the set of input parameters that are needed by the micro-services that implement the associated procedure. The input parameters could be associated with the collection (or record series) into which a record was deposited. On deposition, a rule could check which input parameters have been defined, and then execute the associated rules.

Policies within Arch were treated as templates, and were stored within policy repositories as XML and associated iRODS metadata. The policy template, in part, aggregated configured rule mappings from the rule mapping repositories, along with associated customizations. Additional elements were specified in the policy template. For example, specifications for required metadata were added to the prototype. The required metadata information was specified such that clients could interrogate it, and dynamically generate interfaces for metadata gathering at ingest. Such required metadata information would then be available at a later stage for the purpose of validation.

Once policies were configured and placed in a repository, they were available for application to a record series. In Arch, an interface allowed an archivist to create a record series, and select a policy that would be bound to the record series. The process of binding the policy to the record series involved a lookup of the specified policy template. Here the rule mappings and other data could be used to create dynamic interfaces. For purposes of the prototype, the policy template was limited to a specified series of policies for checksum validation, replication, malware scanning, and setting of retention dates. Once a series was created, and a policy is bound to it, a set of metadata mappings and XML policy descriptors exist such that a policy can be resolved for data objects placed into a series. The constellation of components appears as in Figure 1.

The maintenance of the metadata associated with policy abstraction was a challenging part of Arch. Existing facilities in iRODS were used as the platform, so that the metadata artifacts that describe policy could themselves be managed by iRODS. In essence, a set of shadow directories were defined at the level of the archive, for a rule mapping repository, policy mapping repository, and record series. The various XML documents that describe a policy were placed in these shadow directories. iRODS AVU (Attribute-Value-Unit) metadata capabilities were used to tag the various components such that they could be discovered by an opaque name. Since policies could be discovered and resolved programmatically, it also becomes possible to create representations of assembled policies that could be preserved with the record series they were bound to.

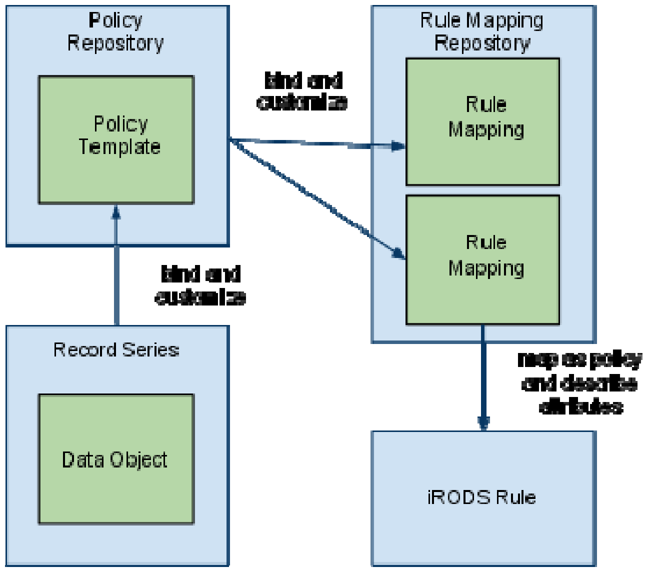


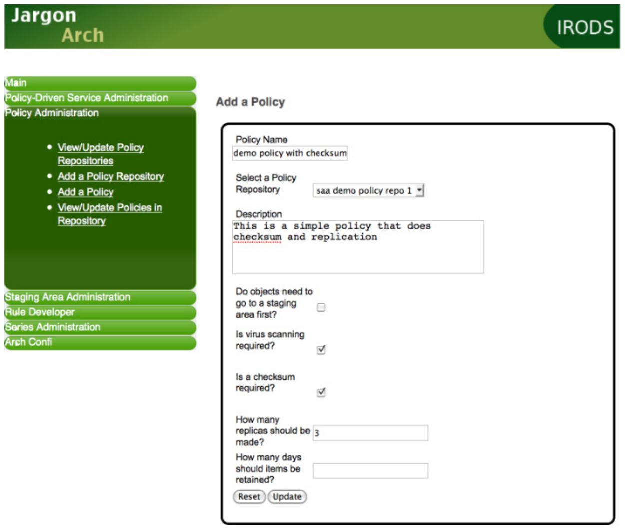
***Figure 1**. Policy components in Arch*

Using the abstractions and mappings described, a policy can be stored within iRODS. When events occur within iRODS, there then is a need for a policy resolution and activation infrastructure within the rule engine. In this prototype, the narrow scope of ingest activities allowed a straight-forward integration point between iRODS events, and the firing of a policy resolution rule. This integration point is specified using the acPostProcForPut event hook within the core.irb file. The policy resolution rule used in Arch is a simplification of a target capability within iRODS that can:

- Recognize an event within the repository.
- Recognize that a policy applies to the event.
- Dynamically resolve the policy to be applied by consulting the Arch policy metadata.
- Activate necessary policies, and update metadata that reflects the outcome of any actions.

Interfaces for Archivists

A pair of interfaces were developed as part of Arch. The first is a web application meant to manage the various policy components. Figure 2 shows one view that adds a policy to a policy repository. This web interface allows creation of an overall archive, with necessary global configuration, as well as rule mapping and policy repositories. Functions are provided to configure policy templates, and to create new record series bound to a specified template. The interface supports round-trip viewing and updating of policies. Importantly, changes in policy can be dynamically detected by the resolution process, as the policy is resolved at invocation.


Jargon
Arch
IRODS
Main
Policy-Driven Service Administration
Policy Administration
View/Update Policy Repositories
Add a Policy Repository
Add a Policy
View/Update Policies in Repository
Staging Area Administration
Rule Developer
Series Administration
Arch Confi

Add a Policy

Policy Name: demo policy with checksum

Select a Policy Repository: saa demo policy repo 1

Description: This is a simple policy that does checksum and replication

Do objects need to go to a staging area first? ☐

Is virus scanning required? ✓

Is a checksum required? ✓

How many replicas should be made? 3

How many days should items be retained?

Reset Update

***Figure 2**. Arch Interface*

To allow viewing of policy status for the ingest process, functionality was added to a desktop transfer client GUI called iDrop. The iDrop GUI allows data objects on a local workstation to be ingested into iRODS. Extensions to iDrop created what we termed a "policy aware" client. The policies bound to record series were sensed by the client, and this was depicted in the interface, as shown in Figure 3.

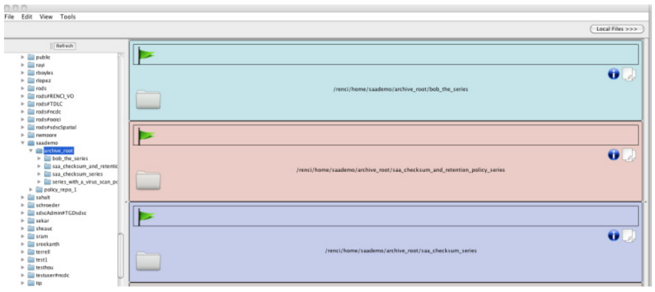


***Figure 3**. iDrop policy detection*

In addition to the detection of bound policy, iDrop was able to interrogate metadata generated by policies, and to depict the status of policy application via icons and other cues in the GUI. For example, failure of a required malware scan resulted in warning icons displayed in iDrop.

Another aspect of policy, the collection of required metadata values at ingest time, was integrated into iDrop. When a file was ingested, the policy was detected, and required metadata values were identified. Basic type information, plain language prompts, and the ability to statically or dynamically represent possible values were also implemented. In this way, a policy could interact with clients at ingest time. This may have wider implications if combined with tools such as the Curator's Workbench, under development by the Carolina Digital Repository [11].

Results and Conclusion

iRODS is a unique distributed data management platform for creating a trusted preservation environment [12]. In particular, the ability to enforce policies across administrative domains through the use of a distributed rule engine is essential. Each policy can be implemented as a rule that is automatically enforced. The iRODS rule engine, with its distributed nature, and with its ability to integrate computer-actionable rules with events occurring throughout the repository, provides a foundation for a system of policy abstraction and dynamic resolution.

The process of creating policy templates and binding them to record series was demonstrated through a graphical web interface. The demonstration implemented a limited policy set associated with ingestion policies. This limited set of policies operated on a limited portion of the data-object life cycle. Even with these limitations, Arch demonstrated many qualities of a system of policy composition and binding.

Future iterations of Arch require development of standard policy-resolution mechanisms within the iRODS rule engine. An important aspect of policy resolution is the an ability to understand events coming from iRODS in the context of the life cycle of documents in the repository. An abstraction of events is as important as an abstraction of policy. iRODS low-level events (policy enforcement points), such as the creation of a collection or storage of a data object, take on different meanings depending on the context. For example, initial ingest in a staging area, and the movement of data objects from a staging area into long-term storage, can each be interpreted as a "postProcForPut" event. Policy components within Arch will need to include bindings to abstract life cycle events so that resolution can take place.

An important quality that Arch demonstrated is the ability to dynamically resolve policy at the point of activation. This points to a new style of rules within iRODS, where there is a late binding between the policy activation points and the actual policy that is invoked. This provides a great deal of flexibility, and a level of isolation between different record series that are subject to differing policy constraints. It may be that elaboration of late-binding and rule resolution mechanisms within iRODS represents a "stepping stone" to the goal of high-level policy abstraction. Arch has been a valuable tool to investigate the qualities of a system that serves as the 'middle ground' between high-level policies and low level computer-actionable rules.

Acknowledgement

The ARCH project was done under the NARA grant: NSF Grant OCI-0848296, "NARA Transcontinental Persistent Archive Prototype"

References

[1] Consultative Committee for Space Data Systems. *Reference model for an Open Archival Information System (OAIS) (CCSDS 650.0-B-1)*. Washington, DC: National Aeronautics and Space Administration (NASA), 2002.

[2] Fedora http://fedora-commons.org

[3] DSpace http://www.dspace.org

[4] Beagrie, Neil. et al. Trusted Digital Repositories: Attributes and Responsibilities, RLG-OCLC Report, 2002, available at http://www.rlg.org/longterm/repositories.pdf

[5] Consultative Committee for Space Data Systems. *Audit and Certification of Trustworthy Digital Repositories* (CCSDS 652.0-R-1), 2009. Washington, DC: National Aeronautics and Space Administration (NASA), available at http://wiki.digitalrepositoryauditandcertification.org/pub/Main/WebHome/652x0r1candidate-update-typoscorrected.doc

[6] Reagan Moore. "Towards a theory of digital preservation." *International Journal of Digital Curation* 3, no. 1 (2008): 63-75.

[7] iRODS Introduction available at https://www.irods.org/index.php/What_is_the_iRODS_Data_System

[8] Reagan Moore, Arcot Rajasekar, Michael Wan, Wayne Schroeder, Policy-Guide Large-scale Data Management System, available at https://www.irods.org/pubs/DICE_Policy_iRODS-2pg2.pdf

[9] MacKenzie Smith, Reagan W. Moore. "Digital Archive Policies and Trusted Digital Repositories", The International Journal of Digital Curation Issue 1, Volume 2 (2007): 92-101.

[10] Mike C. Conway, Jewel H. Ward, Antoine De Torcy, Hao Xu, Arcot Rajasekar and Reagan W. Moore, Policy-based Preservation Environments: Policy Composition and Enforcement in iRODS

[11] Curator's Workbench https://github.com/UNC-Libraries/Curators-Workbench

[12] Arcot Rajasekar, Reagan Moore, Mike Wan, Policy-based Distributed Data Management Systems, (2010) , Journal of Digital Information 11(1): Open Repositories 2009, available at http://journals.tdl.org/jodi/issue/view/91

Author Biography

Mike Conway is a Research Assistant at the DICE Center at UNC-Chapel Hill. He is the Java Architect and Interface developer for the iRODS Data Grid. He is currently in the Masters of Science in Information Science program at the School of Information and Library Science at UNC Chapel Hill. His interests include policy abstraction, as well as the development of the iRODS platform.

Jewel H. Ward is a doctoral student in the School of Information and Library Science at the University of North Carolina at Chapel Hill. She has been a research assistant with the Data Intensive Cyber Environments (DICE) research group since 2007. She worked at the University of Southern California (USC) as the Program Manager for the USC Digital Archive from 2004-2006. Her research interests include machine-level policy implementation, and, data and metadata transfer preservation systems and infrastructures including iRODS and the OAI-PMH.