Synthesis Lectures on Information Concepts, Retrieval, and Services

Editor
Gary Marchionini, University of North Carolina, Chapel Hill

iRODS Primer: Integrated Rule-Oriented Data System

Online Multiplayer Games
William Sims Bainbridge

Meredith Ringel Morris and Jaime Teevan

Multimedia Information Retrieval
Stefan Rüger

Information Architecture: The Design and Integration of Information Spaces
Wei Ding and Xia Lin

Hypermedia Genes: An Evolutionary Perspective on Concepts, Models, and Architectures
Nuno Guimarães and Luis Carriço

Reading and Writing the Electronic Book
Cathy Marshall

Understanding User – Web Interactions via Web Analytics
Bernard J. (Jim) Jansen

XML Retrieval
Mounia Lalmas
Faceted Search
Daniel Tunkelang

Introduction to Webometrics: Quantitative Web Research for the Social Sciences
Michael Thelwall

Automated Metadata in Multimedia Information Systems: Creation, Refinement, Use in Surrogates, and Evaluation
Michael G. Christel

Exploratory Search: Beyond the Query-Response Paradigm
Ryen W. White and Resa A. Roth

New Concepts in Digital Reference
R. David Lankes
iRODS Primer

Integrated Rule-Oriented Data System

Arcot Rajasekar
Reagan Moore
Chien-Yi Hou
Christopher A. Lee
Richard Marciano
Antoine de Torcy

Data Intensive Cyber Environments Center and School of Information and Library Science,
University of North Carolina at Chapel Hill

Michael Wan
Wayne Schroeder
Sheau-Yen Chen
Lucas Gilbert
Paul Tooby
Bing Zhu

Data Intensive Cyber Environments Center and Institute for Neural Computation,
University of California, San Diego

SYNTHESIS LECTURES ON INFORMATION CONCEPTS, RETRIEVAL, AND SERVICES #12

MORGAN & CLAYPOOL PUBLISHERS
Policy-based data management enables the creation of community-specific collections. Every collection is created for a purpose. The purpose defines the set of properties that will be associated with the collection. The properties are enforced by management policies that control the execution of procedures that are applied whenever data are ingested or accessed. The procedures generate state information that defines the outcome of enforcing the management policy. The state information can be queried to validate assessment criteria and verify that the required collection properties have been conserved. The integrated Rule-Oriented Data System implements the data management framework required to support policy-based data management. Policies are turned into computer actionable Rules. Procedures are composed from a Micro-service-oriented architecture. The result is a highly extensible and tunable system that can enforce management policies, automate administrative tasks, and periodically validate assessment criteria.

**KEYWORDS**

data life cycle, data grid, digital library, preservation environment, policy-based data management, rule engine, iRODS, metadata catalog, assessment criteria, policies, micro-services
Acknowledgments

This research was supported by:

- NSF SDCI 0721400, SDCI Data Improvement: Data Grids for Community Driven Applications (2007–2010)

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the National Archives and Records Administration, the National Science Foundation, or the U.S. Government.
# Contents

1. **Introduction** ............................................................................................................... 1

2. **Integrated Rule-Oriented Data System** ................................................................. 5
   2.1 Data Grid Overview .......................................................................................... 6

3. **iRODS Architecture** ............................................................................................. 15
   3.1 Virtualizations in iRODS ................................................................................ 21
   3.2 iRODS Components ....................................................................................... 22

4. **Rule-Oriented Programming** .................................................................................. 25
   4.1 Session State Variables ................................................................................... 29
   4.2 Persistent State Information Variables ......................................................... 33
   4.3 User Environment Variables .......................................................................... 33

5. **The iRODS Rule System** ........................................................................................ 41
   5.1 The iRODS Rule Architecture ....................................................................... 42
   5.2 Rules .............................................................................................................. 43
   5.3 Rule Grammar ............................................................................................... 44
   5.4 RuleGen Language ........................................................................................ 48
      5.4.1 Using the RuleGen Parser ..................................................................... 48
      5.4.2 Example Rule Build ............................................................................. 48
   5.5 iRODS Rule Engine ....................................................................................... 50
   5.6 Default iRODS Rules .................................................................................... 51
   5.7 Session Variables Available for Each Rule .................................................. 55

6. **iRODS Micro-services** ............................................................................................. 59
   6.1 Micro-service I/O Arguments ...................................................................... 74
   6.2 Naming Conventions ..................................................................................... 77
      6.2.1 Variable Naming Conventions ............................................................... 77
      6.2.2 Constant Naming Conventions ............................................................. 77
6.2.3 Function Naming Conventions ............................................................. 77
6.2.4 File Naming Conventions ................................................................. 78

7. Example Rules .......................................................................................... 79
7.1 File Manipulation Rules ........................................................................... 80
  7.1.1 List All Files in a Collection ............................................................. 80
  7.1.2 List Checksums for All Files in a Collection ..................................... 81
  7.1.3 Verify Checksum of All Files in a Collection ................................... 82
  7.1.4 Recompute Checksum of All Files in a Collection ............................ 83
  7.1.5 Copy Files from Source to Destination Collections ........................... 84
  7.1.6 Make a Replica of Each File in a Collection ....................................... 85
  7.1.7 Trims the Number of Replicas of a File ............................................. 86
7.2 User Interaction Rules ............................................................................. 88
  7.2.1 Send E-mail to Specified E-mail Address .......................................... 88
  7.2.2 Periodically Verify Checksum of Files .............................................. 88
  7.2.3 Remove Expired Files ....................................................................... 90
7.3 Rule Manipulation .................................................................................. 90
  7.3.1 Print Rules Being Used by the Rule Engine ....................................... 90
  7.3.2 Change the Rules in the core.Irb File .............................................. 91
  7.3.3 Change Back to the Rules in the core.Irb.Orig File ............................ 92
  7.3.4 Pre-Pend Rules ............................................................................... 92
  7.3.5 Pre-Pend Rules and Logical Name Mappings .................................... 93
  7.3.6 Append Rules and Logical Name Mappings ..................................... 94
  7.3.7 Replicate Each File in a Collection ................................................. 95
7.4 System Testing ...................................................................................... 96
  7.4.1 Test Parametric Variable ................................................................. 96
  7.4.2 Test $-Variable ............................................................................. 97
  7.4.3 Test “While” Loop Execution .......................................................... 99
  7.4.4 Test “For” Loop Execution .............................................................. 99
  7.4.5 Test “If-Then-Else” Execution ....................................................... 100
  7.4.6 Test Writing to Stdout and Stderr Buffers ....................................... 101
  7.4.7 Test Sending E-Mail ...................................................................... 102
  7.4.8 Test “For Each” Loop for Comma-Separated List ............................. 102
  7.4.9 Test “For Each” Loop on a Query Result ........................................ 103
  7.4.10 Test Remote Execution of Micro-Service Writes ............................... 104
# 7.4.11 Test Remote Execution of Delayed Writes

- Page 106

# 7.5 Resource Selection Example

- Page 107

# 8. Extending iRODS

- Page 109

## 8.1 Changing the IP Address

- Page 109

## 8.2 How to Interface a Micro-Service

### 8.2.1 How to Interface a Micro-Service as a Module

- Page 110

## 8.3 Web Services as Micro-Services

### 8.3.1 First Step (Done Only Once)

- Page 116

### 8.3.2 Second Step (Done for Each Web Service)

- Page 117

## 8.4 iRODS FUSE User Level File System

- Page 120

## 8.5 Mounted iRODS Collection

### 8.5.1 Building and Linking Libtar

- Page 123

## 8.6 iRODS Development Information

- Page 123

# Appendix A: iRODS Shell Commands

- Page 125

# Appendix B: Rulegen Grammar

- Page 131

# Appendix C: Exercises

- Page 137

# Author Biographies

- Page 141
Recent decades have seen a rapid rise in collaborative activities in scientific research, and more broadly across many sectors of society. Driven by new information technologies such as the Web as well as the increasing complexity and interdisciplinary nature of today’s research problems, from climate change to the world’s increasingly integrated economies, the need for technologies, sometimes called “cyberinfrastructure,” that enable researchers to collaborate effectively continues to grow rapidly. The Integrated Rule-Oriented Data System (iRODS) is a state-of-the-art software that supports collaborative research, and, more broadly, management, sharing, publication, and long-term preservation of data that are distributed.

A tool for collaboration, iRODS is itself the product of a fruitful collaboration spanning more than a decade among high performance computing (HPC), preservation, and library communities, whose real-world needs have driven and shaped iRODS development. The computational science and HPC communities are inherently interdisciplinary, generating and using very large data collections distributed across multiple sites and groups. The massive size of these data collections has encouraged development of unique capabilities in iRODS that allow scaling to collections containing petabytes of data and hundreds of millions of files.

The preservation community brings the need for long-term preservation of digital information, a challenging problem that is still an active research area to which iRODS research activities have made significant contributions. Interestingly, there turned out to be significant commonalities in the requirements for preserving digital data in time and collaborative sharing of distributed data collections across space, whether geographic, institutional, disciplinary, etc.

The third community that has contributed to iRODS development is the library community, with expertise in descriptive metadata that is essential for management, discovery, repurposing, as well as controlled sharing and long-term preservation of digital collections.

In collaborating with these communities, iRODS research and development has been characterized by close attention to the practical requirements of a wide range of users, resulting in pioneering architecture and solutions to numerous distributed data challenges that now form the
iRODS Data System for managing, sharing, publishing, and preserving today’s rapidly growing and increasingly complex digital collections.

iRODS is a software middleware, or “cyberinfrastructure,” that organizes distributed data into a sharable collection. The iRODS software is used to implement a data grid that assembles data into a logical collection. Properties such as integrity (uncorrupted record), authenticity (linking of provenance information to each record), chain of custody (tracking of location and management controls within the preservation environment), and trustworthiness (sustainability of the records) can be imposed on the logical collection. When data sets are distributed across multiple types of storage systems, across multiple administrative domains, across multiple institutions, and across multiple countries, data grid technology is used to enforce uniform management policies on the assembled collection. The specific challenges addressed by the iRODS Data Grid include:

- Management of interactions with storage resources that use different access protocols. The data grid provides mechanisms to map from the actions requested by a client to the protocol required by a specific vendor supplied disk, tape, archive, or object-relational database.
- Support for authentication and authorization across systems that use different identity management systems. The data grid authenticates all access, and authorizes and logs all operations on the files registered into the shared collection.
- Support for uniform management policies across institutions that may have differing access requirements such as different institutional research board approval processes. The policies controlling use, distribution, replication, retention, disposition, authenticity, integrity, and trustworthiness are enforced by the data grid.
- Support for wide-area-network access. To maintain interactive response, network transport is optimized for moving massive files [through parallel input/output (I/O) streams], for moving small files (through encapsulation of the file in the initial data transfer request), for moving large numbers of small files (aggregation into tar files), and for minimizing the amount of data sent over the network (execution of remote procedures such as data subsetting on each storage resource).

In response to these challenges, iRODS is an ongoing research and software development effort to provide software infrastructure solutions that enable collaborative research. The software systems are implemented as middleware that interacts with remote storage systems on behalf of the users. The goal of the iRODS team is to develop generic software that can be used to implement all distributed data management applications, through changing the management policies and procedures. This has been realized by creating a highly extensible software infrastructure that can be modified without requiring the modification of the core software or development of new software
code. This publication describes the data grid technology in Chapter 2, the iRODS architecture in Chapter 3, the Rule-Oriented Programming model in Chapter 4, the iRODS Rule system in Chapter 5, the iRODS Micro-services in Chapter 6, examples of iRODS rules in Chapter 7, and extensions to iRODS in Chapter 8. Appendix A lists the iRODS i-Commands that can be executed interactively, Appendix B lists the rulegen grammar, and Appendix C lists exercises that can be used to test knowledge of the iRODS policy-based data management system.

Documentation for iRODS is continually being updated by the growing iRODS open source community on the iRODS wiki at http://www.irods.org, covering topics such as installation, how to use iRODS, administration, and development information. The wiki also contains iRODS-related publications for further reading.
iRODS is software middleware that manages a highly controlled collection of distributed digital objects, while enforcing user-defined Management Policies across multiple storage locations. The iRODS system is generic software infrastructure that can be tuned to implement any desired data management application, ranging from a Data Grid for sharing data across collaborations, to a digital library for publishing data, to a preservation environment for long-term data retention, to a data processing pipeline, to a system for federating real-time sensor data streams.

The iRODS technology, which is developed by the Data Intensive Cyber Environments (DICE) group, is distributed between the University of North Carolina at Chapel Hill (UNC) and the University of California, San Diego (UCSD). The DICE Center has been established at UNC to coordinate development, application, and use of the iRODS Data Grid in cyberinfrastructure. The team at UCSD is associated with the Institute for Neural Computation.

The ideas for the iRODS project have existed for a number of years, and became more concrete through the National Science Foundation-funded project, Constraint-Based Knowledge Systems for Grids, Digital Libraries, and Persistent Archives, which began in the fall of 2004. The development of iRODS was driven by the lessons learned in nearly 10 years of deployment and production use of the DICE Storage Resource Broker (SRB) Data Grid technology and through applications of theories and concepts from a wide range of well-known paradigms from computer science fields such as active databases, program verification, transactional systems, logic programming, business rule systems, constraint-management systems, workflows, and service-oriented architecture. The iRODS Data Grid is an adaptable middleware, in which management policies and management procedures can be dynamically changed without having to rewrite software code.

The iRODS Data Grid expresses management policies as computer actionable Rules, and management procedures as sets of remotely executable Micro-services. The Rules control the execution of the Micro-services. The state information generated by the Micro-services is stored in a metadata catalog (iCAT). The iRODS Data Grid manages input and output information from the