Introduction: Large-scale Data Management Systems (LDMS) facilitate management of large collections (Peta-bytes and more) of data that are geographically distributed across multiple administrative domains. LDMSs provide a data sharing interface to applications that is highly scalable, vendor-agnostic and adaptive to technology change. In addition to being fault-tolerant and network latency aware, an LDMS provides facilities for multiple-levels of authentication and authorization, facilities for auditing and accounting, support strategies for data placement, replication, backup, archiving and versioning, interfaces for rapid data ingestion and access, visualization and third-party data movement, support for server-side workflows for data assimilation, analysis, transformation and fusion, distributed recovery procedures, and uniform data access. An LDMS supports data access from workstations, to large-scale compute clusters, and supercomputers. Since an LDMS brokers hundreds of millions of files, it also supports rich semantic data discovery across system, descriptive and normative metadata.

Administration and control of an LDMS can be very challenging not only because of the scalability and heterogeneity of the user-base, resources, data types, and methods and protocols for data access, but also because of the diversity and complexity of the administrative policies one needs to enforce in such a system. For example, the need for different levels of access authorization to a collection of files, layered with the need for changes in access policy over time interspersed with a need for rapid reaction to policy changes based on current events requires a dynamic environment that can respond without human intervention. Multiply this complexity with policies that need to be specific for different collections of data or data streams, different types of users and services, and different types of resources and networks. The challenge becomes overwhelming for “one size fits all” type of data management systems.

Integrated Rule-Oriented Data Systems: In the Data Intensive Cyber Environments [DICE] group, based at UNC and UCSD, we have been conducting research and prototype development on policy-governed large scale data management systems (pLDMS) called the integrated Rule-Oriented Data Systems [iRODS]. IRODS is adaptive middleware that provides a flexible, extensible and customizable data management architecture. Workflow functions are encoded as sets of micro-services that are controlled through user-defined and administrator-defined rules. The functionality control can be decided by the users within constraints imposed by the system and application developers. Hence, changes to a particular process or policy can be easily constructed by the user and tested and deployed without the aid of system and application developers. The user can also define conditions when these rules get triggered thus controlling application of different rules and rule sets based on current events and operating conditions. The programming of rules in iRODS can be viewed as lego-block type programming. The building blocks for the iRODS are micro-services - small, well-defined procedures/functions that perform a specific task. Users and administrators “chain” micro-services to implement a larger macro-level functionality (as rules) to be executed on triggered conditions. The rules can be published for use by others. For example, one may encode a rule that when accessing a data object from a collection C, additional authorization checks need to be made. These authorization checks can be encoded as a set of additional rules with different triggers that can fire based on current operating conditions. In this way, one can control access to sensitive data based on rules and can escalate or reduce authorization levels dynamically as the situation warrants. The iRODS rule engine design builds upon the application of theories and concepts from a wide range of well-known paradigms from fields such as active databases, transactional systems, logic programming, business rule systems, constraint-management systems, workflows, service-oriented architecture and program verification. Figure 1 provides an architectural diagram for the iRODS system.
Rules are invoked either by client operations (such as ingestion of a new file, access to a copy of a file) or by an event that changes the metadata state. The distributed rule engine fires the rules at the storage location of the data and executes the relevant micro-services that define the intended functionality. The micro-services can check for conditions, change the metadata state or perform an operation such as replicating a file, or broadcasting an email, etc. The Administrative interface provides a means for defining rules (a rich rule language supports atomic, deferred, and periodic execution of hierarchical rule sets), checking for rule base consistency, and managing the dynamic addition of users, user groups, storage resources, and storage resource groups.

The iRODS prototype has been developed based on our experience in building and deploying a first generation data grid system called the Storage Resource Broker [SRB]. The SRB manages collections in a wide variety of domains from bioinformatics, to astronomy sky surveys, to earthquake simulations, to sensor data streams. It has been shown to scale to petabytes of data with hundreds of millions of files. The main restriction of the SRB was hard-coded data management policies. If a user wanted to perform complex sets of operations on datasets, they had to program at the client level and move the data from the server to perform the operation. IRODS obviates the need for this through its flexible distributed rules and micro-services allowing operations to be performed at the data server.

Conclusion: The iRODS system provides a prototypical large-scale data management system. It allows encoding of user and administrative policies in a rule language based on the chaining of micro-services, leading to a flexible and extensible LDMS. Version 1.1 of iRODS has been distributed as open source software and is being applied in multiple projects including the NARA Trans-continental Persistent Archive Testbed [TPAP].

Acknowledgement: The work on iRODS is performed under a grant from the National Science Foundation SDCI (OCI-0721400) and a grant from the National Archives and Records Administration /National Science Foundation (OCI-0848296).

References:
[iRODS] integrated Rule-Oriented Data Systems: https://www.irods.org