A System for Managing Data Provenance in In Silico Experiments

Jarrod Trevathan, Ian M. Atkinson, Wayne W. Read, Nigel Sim and Chris Christensen

eResearch Centre,
James Cook University,
101 Angus Smith Drive, Douglas, Townsville, Queensland, Australia, 4811
Email: eResearch@jcu.edu.au

Abstract

In silico experiments use computers or computer simulation to speed up the rate at which scientific discoveries are made. However, the voluminous amounts of data generated in such experiments is often recorded in an ad hoc manner without regard to workflow, and often lacks rigorous business rules. The absence of stringent auditing and reporting policies makes it difficult to repeat experiments and largely denies independent parties the ability to verify study results. This paper presents a data provenance management system based on the utility of the ICAT metadata storage service as a viable schema for representing in silico experiments. The system provides a portal interface to integrate ICAT with job execution. We have built on a data repository which can handle arbitrary data size, complexity and type. This can be practically used to compare, validate and aid in the repetition of historic experiments. Furthermore, data can be verified via external repositories/sources which will ultimately enhance the scientific merit of in silico experimentation. Our proposed system augments existing applications and therefore does not require users to modify their current experimentation platform. A test case for a pharmacological study is presented to illustrate the proposed system's versatility for reporting and auditing of experiments and their results.

Keywords: Provenance, in silico, experiment management, workflow, version control

1 Introduction

Execution of computational, or *in silico* experiments follows the same methodical processes as any other scientific pursuit. Therefore, clear, accurate and detailed record keeping is equally important in a virtual environment as it would be in a physical laboratory. Since the information about an in silico experiment setup is already computerised, it is a much simpler task to capture this in a standardised way than in a comparable physical laboratory. Various disciplines specify standards and procedures for the collection and storage formats of scientific metadata. However, there are numerous inconsistencies between disciplines.

As Grid (Foster and Carl 1999, Fran et al. 2003) and Cloud (Armbrust et al. 2009, Buyya et al. 2008) computing systems become more widely adopted to perform in silico experiments, it is increasingly important to acknowledge that their heterogeneous nature poses a risk to experiment reliability and repeatability. Systems will vary in terms of both hardware and software. This makes it difficult to ascertain whether the outcome of an experiment will be the same when repeated on different platforms.

Extensive documentation regarding the main attributes of the system environment can help to overcome this by providing appropriate insight which can explain unusual trends in the data.

Collecting static machine information regarding hardware and software, sets up reference metrics which can be used to group and loosely compare execution runs on different machines. The same workflow run on a machine with identical memory and CPU should be very similar. Significant discrepancies would indicate there are issues relating to the hardware, software, or the application being executed. Then software dependencies could be investigated, ideally leading to the offending library, or methods being identified and rectified.

When analysing a dataset using computational techniques, it is easy and sometimes commodious to assume that the dataset simply came into existence. This is typically done without questioning the true origin, quality, or how/why it came to be in its intermediate and final representations. Simple *provenance* information is essential when trying to audit, review or reproduce findings (Buneman et al. 2000, Simmhan et al. 2005).

Existing *data provenance management systems* (DPMS) are built into some eScience tools. For example, experimentation environments such as Chimera (Foster et al. 2002), ^{my}Grid (Stevens et al. 2003, Yu and Buyya 2005) and Kepler (Altintas et al. 2006, Ludscher et al. 2005) provide various mechanisms for automatically capturing data provenance for experiments conducted using their frameworks. These environments use their own proprietary system for storing the provenance information in either an XML, RDF or relational form. Such systems are satisfactory when the target audience:

1. Is largely consuming existing services;
2. Is able to migrate to a new system; and
3. Does not desire to participate in the development of in silico simulations.

However, such an application-specific nature leads to a heterogeneous environment where each system does not interface conveniently with other external tools and systems. It is also undesirable to re-tool the practitioner with an entire new framework in an effort to improve their book keeping practises.

While the collection methods for provenance information will vary between implementations, it does

not need to occur within the main application. We propose that a *pre and post process* can be used to collect this information *without the need for modification to the running application.* This data is written to the document store, and returned along with the results of the main application. Therefore, a provenance management system must enable the *current users* of in silico experimentation to capture their provenance information without amending their existing systems. The provenance data can then be stored in a general-purpose repository where it can be verified by independent third parties and used to reproduce experiments if desired.

The physical science community utilises a number of *Laboratory Information Management Systems* (LIMS) which track physical experimentation. One such system from the crystallography community is the ICAT schema (Flannery et al. 2009). ICAT is a relational database implementation of the *Council for the Central Laboratory of the Research Councils* (CCLRC) Scientific Metadata Model. There are a number of *Service-Oriented Architecture* (SOA) implementations which present this schema to other applications. The generic and interoperable nature of ICAT makes it ideal as a repository for data storing provenance from in silico experiments in a verifiable manner.

This paper presents a DPMS based on the utility of the ICAT metadata storage service as a viable schema for representing in silico experiments. The system provides a portal interface to integrate ICAT with job execution. We have built on a data repository which can handle arbitrary data size, complexity and type. This can be practically used to compare, validate and aid in the repetition of historic experiments. Furthermore, data can be verified via external repositories/sources which will ultimately enhance the scientific merit of in silico experimentation. Our proposed system augments existing applications and therefore does not require users to modify their current experimentation platform. A test case for a pharmacological study is presented to illustrate the proposed system's versatility for reporting and auditing of experiments and their results.

This paper is organised as follows: Section 2 discusses related work on grid and cloud computing data management systems. Section 3 describes the architectural framework within which the system operates. Section 4 presents the design methodology for constructing our system. Section 5 describes the implementation of the proposed system. Section 6 gives a case study and discussion for the proposed system. Section 7 provides some concluding remarks and avenues for future work.

2 Related Work

The Computer Science and Informatics communities currently do not adopt a uniform experiment management regime. However, there exist opportunities and resources to facilitate data provenance management and enable other flow-on benefits to these research communities. This section presents related work that has been conducted on creating in silico workflow and data provenance management systems.

Bunemann et al (Buneman et al. 2000) outline some of the major issues with capturing data provenance, while Simmhan et al (Simmhan et al. 2005) discuss a number of models and uses of provenance metadata in eScience. Yu and Buyya (Yu and Buyya 2005) present a taxonomy of workflow management systems for Grid computing.

Chimera (Foster et al. 2002) is a virtual data system, which combines a virtual data catalogue for representing data derivation procedures and derived data, with a virtual data language interpreter The interpreter translates users' requests into data definition and query operations on the database. Chimera is coupled with distributed "Data Grid" services to enable on-demand execution of computation schedules constructed from database queries. The system has been tested on two problems. The first involved the reconstruction of simulated collision event data from a high-energy physics experiment. the second was the search of digital sky survey data for galactic clusters.

A novel use of provenance in Chimera is to plan and estimate the cost of regenerating datasets. When a dataset has been previously created and it needs to be regenerated (e.g., to create a new replica), its provenance guides the workflow planner in selecting an optimal plan for resource allocation.

^{my}Grid (Stevens et al. 2003) provides middleware in support of in silico (computational laboratory) experiments in biology, modelled as workflows in a Grid environment. myGrid services include resource discovery, workflow enactment, and metadata and provenance management, which enable integration and present a semantically enhanced information model for bio-informatics. myGrid is service-oriented and executes workflows written in XScufl language using the Taverna engine (Oinn et al. 2004). A provenance log of the workflow enactment contains the services invoked, their parameters, the start and end times, the data products used and derived, and ontology descriptions, and it is automatically recorded when the workflow executes. This process-oriented workflow derivation log is inverted to infer the provenance for the intermediate and final data products. Users need to annotate workflows and services with semantic descriptions to enable this inference and have the semantic metadata carried over to the data products.

Kepler (Ludscher et al. 2005) is an open-source scientific workflow system which enables scientists to design scientific workflows and execute them efficiently. Kepler provides emerging Grid-based approaches for access to distributed resources such as data and computational services, while hiding the underlying complexity of the Grid technologies. The Kepler system supports the automation of low-level data processing tasks so the focus can remain on the scientific questions of interest. The workflows that Kepler produces can be implemented in cross platform environments, provide documentation and visualisation of the processes and bring the power of distributed databases, computational Grid resources and applications to the desktop. Each workflow step is represented by actors, which are individual processing components that can be manipulated through a drag and drop method into a workflow, via Kepler's visual interface. The actors are then connected and organised according to the data flow, and the dependencies among them, to form the workflow.

Altintas et al (Altintas et al. 2006) present a framework for data and process provenance in Kepler. They outline the requirements and issues related to data and workflow provenance in a multi-disciplinary workflow and show how generic provenance capture can be facilitated by Kepler's actor-oriented workflow environment. They also describe the usage of the stored provenance information for efficient rerunning of scientific workflows.

Other DPMSs include the CMCS project (Myers et al. 2003, Pancerella et al. 2003) for collaboration and metadata-based management for multiscale science, the Earth System Science Workbench (ESSW) (Frew and Bose 2001) for metadata management and data storage for Earth Science applications, and Trio (Cui and Widom 2000, Widom 2005)

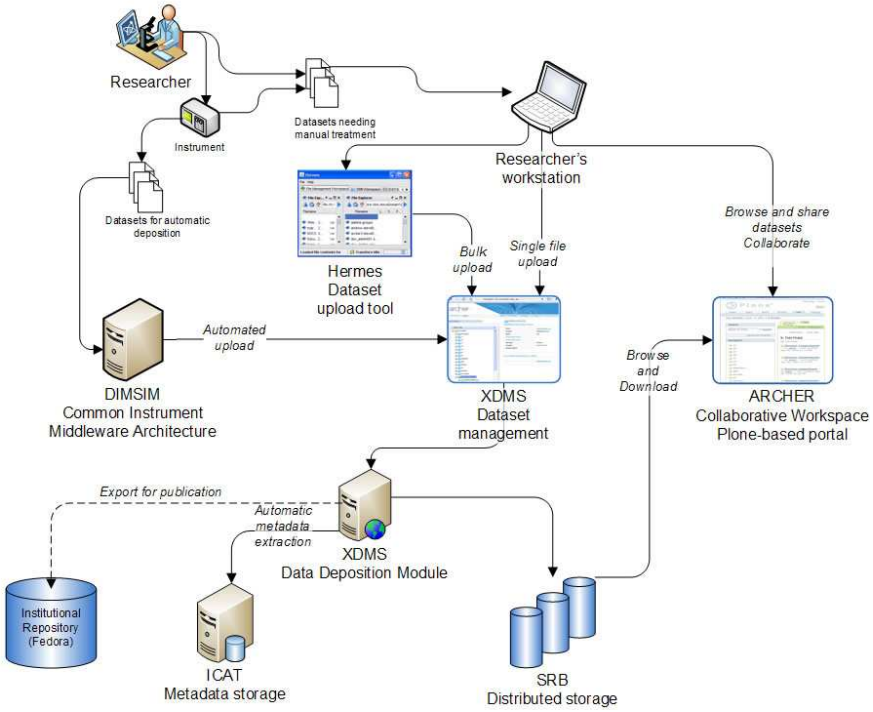

Figure 1: The ARCHER research data management Workflow

for tracing lineage information in data warehouses.

3 ARCHER

This section describes the ARCHER project and presents architecture within which the proposed DPMS operates.

The *Australian ResearCH Enabling enviRonment* [1] (ARCHER) project (Androulakis et al. 2009) is an Australian higher education initiative which has developed 'production-ready' software tools, operating in a secure environment, to assist researchers to:

- Collect, capture and retain large data sets from a range of different sources including scientific instruments;
- Deposit data files and data sets to eResearch storage repositories;
- Populate these eResearch data repositories with associated metadata;
- Permit data set annotation and discussion in a collaborative environment; and
- Support next-generation methods for research publication, dissemination and access.

3.1 Applications

Figure 1 illustrates the main software components and workflow of the ARCHER platform. Within this framework, ARCHER offers the following services:

* ARCHER Enhanced Plone - a collaborative workspace development tool for building websites where researchers can come together.
* HERMES - for managing research datasets from a desktop client application.

[1] www.archer.edu.au

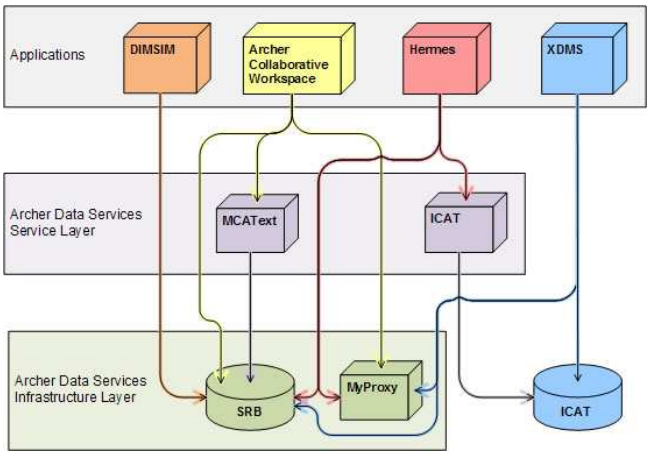

Figure 2: Layers in ARCHER

* HYDRANT - for managing workflow automations from a web application.
* DIMSIM - Distributed Integrated Multi-Sensor and Instrument Middleware for concurrent data capture and analysis.
* CIMA - Common Instrument Middleware Architecture.
* SAL - A Sensor Abstraction Layer to automate sensor network hardware configuration and simplify sensor instrument access and control (Trevathan et al. 2010).

3.2 Data Services Layer

Figure 2 illustrates the logical layers within the ARCHER system. At the top is the Application Layer that supports the aforementioned applications. Below this is the Data Services Layer. The *ARCHER Data Services* (ADS) infrastructure supports other components in the ARCHER Toolset, and incorporates both third-party and ARCHER-developed software.

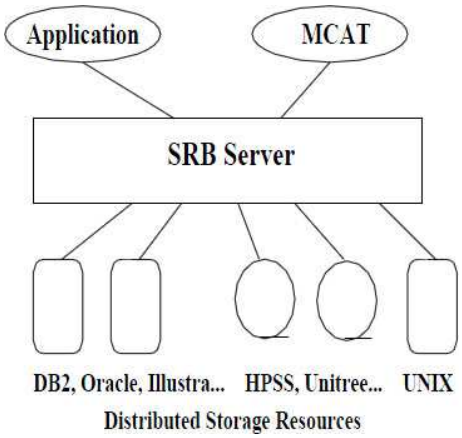

Figure 3: The SRB middleware (figure taken from (Baru et al. 1998))

It provides data storage via the *Storage Resource Broker* (SRB) (Baru et al. 1998), metadata storage with ICAT, and authentication through MyProxy.

ADS is made up of two layers:

- ADS Service Layer provides web service interfaces to the data and metadata storage.
- ADS Infrastructure Layer provides distributed storage, authentication, and an optional certificate authority.

Infrastructure Layer

The ADS Infrastructure Layer is a streamlined package of third party grid components such as Globus [2], MyProxy and SRB with customisations to allow authentication to an *Lightweight Directory Access Protocol* (LDAP) server. The components are all installed through automated deployment scripts with minimal configuration required.

SRB provides a uniform interface to heterogeneous data storage resources over a network (see Figure 3). It is a logical distributed file system based on a client-server architecture which presents users with a single global logical namespace or file hierarchy. As part of this, it implements a logical namespace (distinct from physical file names) and maintains metadata on data-objects (files), users, groups, resources, collections, and other items in a *Metadata Catalog* (MCAT). System and user-defined metadata can be queried to locate files based on attributes as well as by name.

SRB is middleware in the sense that it is built on top of other major software packages (various storage systems, real-time data sources, a relational database management system, etc.). It has callable library functions that can be utilized by higher level software. However, it is more complete than many middleware software systems as it implements a comprehensive distributed data management environment, including various end-user client applications. It has features to support the management and collaborative (and controlled) sharing, publication, replication, transfer, and preservation of distributed data collections.

SRB is a commercial product, free for use by academic institutions, and with full source code available. It is sometimes used in conjunction with computational grid computing systems, such as Globus Alliance, and can utilize the Globus Alliance Grid Security Infrastructure authentication system. SRB can store and retrieve data in archival storage systems such as HPSS and SAM-FS, on disk file systems (Unix, Linux, or Windows), as Binary Large Objects or tabular data in relational database management systems, and on tape libraries.

The *Virtual Data Toolkit* [3] (VDT) is a grid software packaging system that installs several grid components, including MyProxy and the Globus libraries.

MyProxy [4] is a service that listens on port 7512, generating short-lived certificates called "proxies" for users upon request. It serves these over the network, allowing users access to remote services. ARCHER adds LDAP authentication to the standard MyProxy installation.

Many parts of ARCHER Toolset require a common *Certificate Authority* (CA) to sign certificates. ARCHER Data Services can set up MyProxy to function as a CA. This is useful in a testing or development environment.

Service Layer

The ADS Service Layer is composed of two web applications – ICAT and MCAText.

ICAT is a metadata storage service that implements the CCLRC Scientific Metadata Model version 2 to record information about scientific experiments. The data from the experiments itself is stored on the SRB, while the metadata is held in the ICAT. ICAT's storage is implemented as a PostgreSQL database, which is installed through the Archer XDMS application.

MCAText is an ARCHER-developed web service layer over SRB and its MCAT database. It provides a high performance mechanism for other services to lookup authorisation information on content within SRB. It provides update notification to other services when content is modified, moved, or created. It is used by certain ARCHER tools, including the ICAT service and ARCHER Collaborative Workspace.

Figure 4 illustrates how all of the internal components interact in the standard ADS configuration.

3.3 Bringing it all Together

The main ARCHER project was completed in 2008. As a result of the work performed in ARCHER, researchers in general are now much closer to: having a place to collect, store and manage experimental data; deploying software tools focused on management of data and information; being able to easily customise a collaborative and adaptable portal web site relevant to their research field; having standardised and secure methods of storing, accessing, and analysing research results; and finding it easier to collaborate and share research datasets and information.

ARCHER has addressed many key issues in e-Research data management. It is enabling researchers to keep better track of their scientific data by organising them into intuitive and generic structures described by the CCLRC Scientific Metadata Model, and by making the research repository easily searchable. It is alleviating issues around the collaboration of large datasets by: supporting the storage of large research datasets, adopting a data-centric view; and by providing an initial set of collaborative tools that can be directly associated with the research data. It is also protecting the confidentiality and security of research data by limiting the access to a project team's research data.

[2] http://www.globus.org/

[3] http://vdt.cs.wisc.edu/

[4] http://www.my-proxy.com/

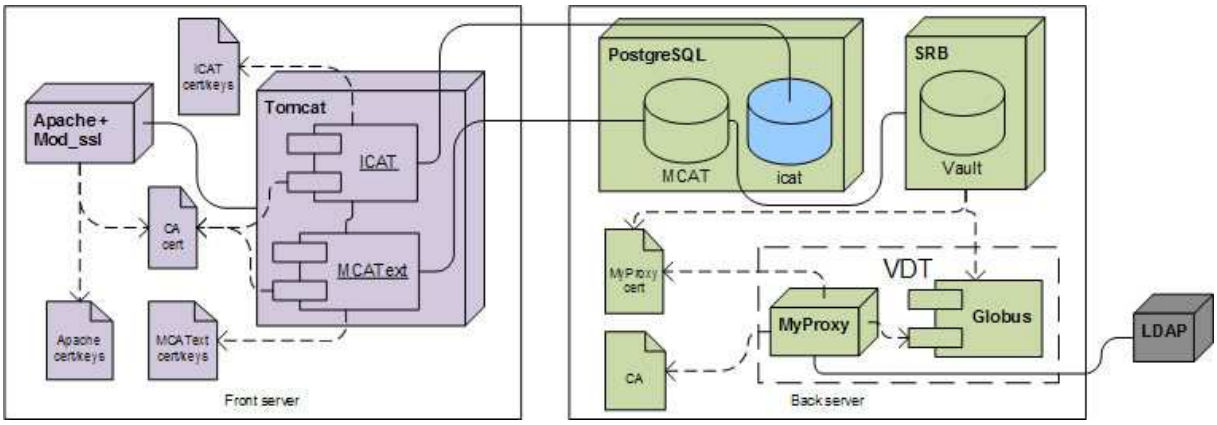

Figure 4: The standard tested ARCHER ADS configuration

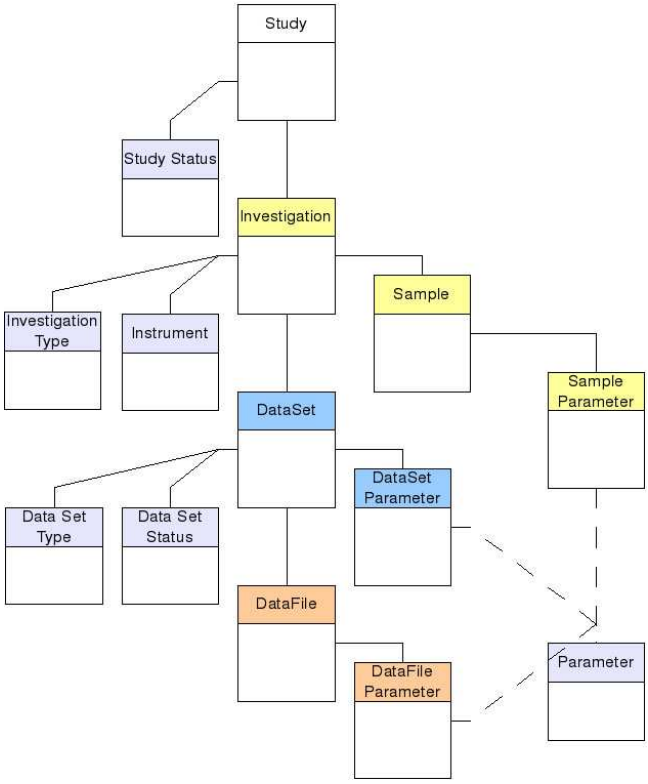

Figure 5: ICAT schema for result sets from physical automated experiments

4 Design Methodology

This section presents the design methodology for constructing the proposed system.

4.1 The ICAT Schema

The proposed DPMS is based on the continued work from the ARCHER implementation of the ICAT service. This implementation provides a web services interface with the schema, and includes tight integration with the SRB as a file repository.

ICAT is primarily concerned with result sets from physical, automated experimentation. Figure 5 illustrates the ICAT schema. Represented within the schema are *studies*, which form the highest grouping of research activities. Studies contain many *investigations* which are the individual experiments. Investigations involve zero or more samples, which are the focus of the investigation, and produce *datasets* and data files.

However, ICAT can also represent in silico experiments, by defining a *dataset type* as an input dataset and not defining any samples. If we consider an investigation to be any process which produces data output from samples or other datasets, then we can model any transformation as a sample. This makes sense especially if we want to capture failed experiments, such as descriptor calculations which fail due to invalid or out of bounds structures.

One of the key extensions provided by the ARCHER-ICAT services is the ability to store auxiliary documents against ICAT elements. It allows listing, insertion, updating and deletion of these documents, given the ICAT element type and element ID as the key. We use this feature to store an XML document for each of the in silico investigation types discussed here.

4.2 User Source Code Integrity

Principle among the requirements for effective data lineage and provenance collection for in silico experiments, is capturing the exact workflow or scripting used to transform or process the data. This is particularly evident when analysis is performed using custom software, or scripted environments such as MATLAB [5] or R [6]. The benefit which the user can leverage from these situations, which cannot necessarily be captured in a single set of configuration parameters, requires a more powerful approach.

Being able to identify the exact code used to perform a specific computation is essential. Given the ease with which small changes can be made to text-based scripts means robust and integrated methods of code validation are required.

Source code version control systems used in software development offer a capability to store, address and recall historic versions of a code base. Utilising this functionality within the provenance framework gives an opportunity to address the temporal software validation issue. However, the following constraints must be applied:

- The version control repository is maintained alongside the main provenance store. This ensures the same level of security from tampering, and also disaster tolerance.
- The code is always checked out from the repository by the execution system, and is not modified before execution.

[5] http://www.mathworks.com/products/matlab/
[6] http://www.r-project.org/

In the proposed system we consider the centralised version control application – *Subversion* [7]. Subversion stores text and binary files in a hierarchical directory tree. When a file is changed, added or deleted, it can be checked back into the central repository, creating a new revision. The user, or another process, can then check these files out at another location and they will be identical to the version used at the original file location. When changes are made, the user is able to view the difference between the working copy and those in the central repository, containing all previous revisions stored in that repository.

Provided the code is in a version control system, then the exact version of the software can be retrieved to process the dataset. The software revision is stored in a network accessible format within the provenance system as follows:

```
<protocol>://<host>[:<port>]/<path>@<revision>
```

The `<protocol>` will typically be http or https, and `<host>` is the network name of the subversion server. The `<port>` will be supplied if it is non-standard, `<path>` is the location of the code, and `<revision>` is what version of the code will be used for this experiment.

4.3 Reporting and Auditing

By collecting provenance information in the aforementioned manner, it becomes possible to produce new tools to assist in the monitoring and supervision of experiment-based research. Through providing a query interface, the ICAT service can enable supervisors to remotely, and continuously monitor students and subordinates. This helps to allow for rapid rectification of experimental errors.

Flagging of other interesting artefacts within results also becomes possible. The uniform storage of these investigations means such annotations can be extracted reliably, and without risking the loss of possibly interesting and insightful avenues for future investigation.

5 Implementation

This section describes how to implement the proposed DPMS using the ICAT schema, a source code version control system, and the related components of the ARCHER project.

5.1 Data Format

Within the ICAT schema, investigations and datasets both have a *type* attribute which is used to distinguish the specific function of the element. The available types are not stipulated by the ICAT specification, which allows the schema to be used for many purposes. The following *investigation types* are defined for the in silico use cases:

- *Retrieval* – This is either downloaded from a website, or queried from a database;
- *Calculation* – Includes the running of format conversion and the calculation of molecular descriptors; and
- *Analysis* – The running of a workflow or application script which processes the data.

The following *dataset types* are defined to complement the investigation types:

- *Input Dataset* – These are the source datasets for the defined Calculation and Analysis investigation types. The data files may or may not be populated, depending on whether this is the primary registration of the dataset, or if it is the culmination of data from other datasets.
- *Output Dataset* – These are the files produced by Calculation or Analysis. Data files need to be populated to describe what is in each file.

By using the dataset type and the investigation type, a complete history of the data back to the original collection can be constructed.

Two supplementary documents are also stored against the investigation which details the operation. These are used to fully capture the metadata about the configuration of the investigations. These documents are stored as XML, and the schema is specific to the investigation type.

The first document is referred to as the *Retrieval* document. The purpose of the retrieval operation is to capture the source information about the dataset. This allows verification to occur at a later date. Furthermore, it can be cited correctly when used in investigations. The format of the retrieval document is as follows:

Retrieval:

- URI
- Retrieval time
- License
- Dublin Core metadata record [8]

The second document is referred to as the *Analysis and Calculation* document. Analyses are computer applications which are run using the data. The metadata associated with them needs to be collected at the point of execution. This data includes the hardware and software environments which affect the execution of the analysis. Furthermore, application parameters also need to be captured so that the analysis can be repeated by an independent third party. As some applications require configuration files, these can be stored in the document, as can references to other input files. Input file validity is confirmed by also storing a SHA-256 [9] checksum of the files. The format of the Analysis and Calculation document is as follows:

Analysis and Calculation:

- Applications
  - Name
  - Version
  - Operating System
  - Runtime
  - Command - command line
    - Library Path (name =)
      - Library name
    - Input Parameters
      - Name
      - Value

[7] http://subversion.tigris.org/

[8] http://dublincore.org/

[9] Secure Hash Algorithm (SHA) is a family of hash functions used for verifying the integrity of a document. SHA-256 uses 32-bit words.

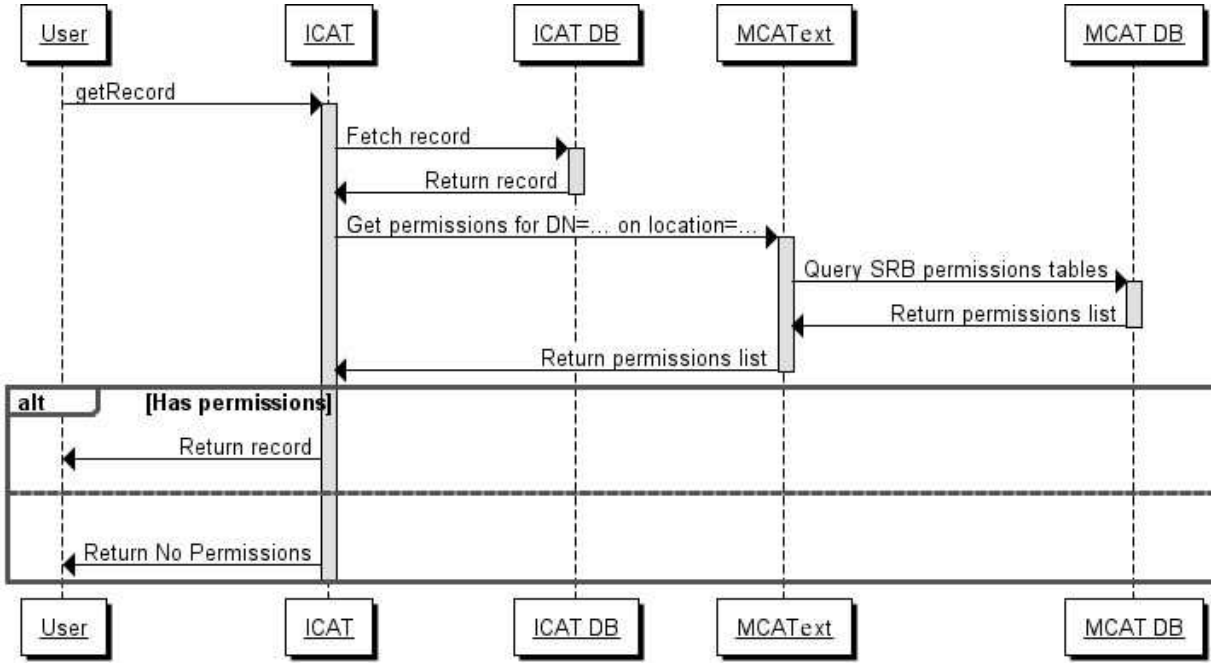

Figure 6: Interactions between ICAT, SRB and MCAText

* Input Files – Data files not already in ICAT
  - Name
  - Location
  - Hash
  - Data – Optional intended for configuration files

When the output files are available, they are moved off their initial storage into the data repository component of the system. This is achieved using a transfer agent which is started on the head node of the execution cluster. The agent is initialised with the details of the ICAT server, destination dataset, and the user's X.509 certificate [10]. It will copy the files into the data store and computes the checksum as this is happening. Upon completion, the details are written directly to the ICAT service against the dataset specified.

5.2 The ICAT Service and Security

The ICAT service exposes the ICAT schema via web services, providing basic creation, read, update, and delete operations for each of the elements. It also allows arbitrary XML documents to be stored against specific ICAT elements. The framework is presented in a manner that is easily extensible for adding new operations to the service, and altering some aspects of the existing service.

The ICAT-SRB services is an extension to the vanilla ICAT service, integrating file data storage using SRB. This implements a security model which is based around the object permissions within SRB. This is achieved by mirroring the core ICAT structure as folders and files within SRB, and then explicitly linking the ICAT records to these SRB objects using the location attribute on the given ICAT object. In the case of ICAT objects which have no direct mirror within the SRB hierarchy, the permissions are inherited from the closest parent which is mirrored. Lookup table objects are access controlled via a separate access control list in the service, as they are always world readable, and administered centrally.

Figure 6 illustrates the interactions between ICAT, SRB and MCAText. Achieving this integration requires both SRB and the ICAT services to be using X.509 authentication, so that the X.500 DN (Distinguished Name) [11] of the authenticated user is used to match the user's permissions via a call from the ICAT service to an SRB support service.

MCAText provides two key functions to authorised services:

- Read only lookups of SRB permission information; and
- Notification of file and folder changes within SRB.

This provides integration of the permissions systems, and also allows ICAT to be kept consistent with object changes within SRB.

5.3 Application Portal

The principle objective of this project was to capture more of the computational experiment information from the users during experimentation. To achieve

[10] X.509 is a cryptographic standard for a public key infrastructure for single sign-on and Privilege Management Infrastructure. X.509 specifies standard formats for public key certificates, certificate revocation lists, attribute certificates, and a certification path validation algorithm.

[11] The X.500 directory service is a global directory service. Its components cooperate to manage information about objects such as countries, organizations, people, machines, and so on in a worldwide scope. It provides the capability to look up information by name (a white-pages service) and to browse and search for information (a yellow-pages service). The information is held in a directory information base (DIB). Entries in the DIB are arranged in a tree structure called the directory information tree (DIT). Each entry is a named object and consists of a set of attributes. Each attribute has a defined attribute type and one or more values. The directory schema defines the mandatory and optional attributes for each class of object (called the object class). Each named object may have one or more object classes associated with it. The X.500 namespace is hierarchical. An entry is unambiguously identified by a distinguished name (DN). A DN is the concatenation of selected attributes from each entry, called the relative distinguished name (RDN), in the tree along a path leading from the root down to the named entry. Users of the X.500 directory may (subject to access control) interrogate and modify the entries and attributes in the DIB.

this, the cluster job submission portal was augmented with ICAT experiment metadata facilities. As job submission is handled by the portal, it is practical to automatically capture execution environment information directly from within the submission scripts. Initially the user is required to provide some information about the job they are submitting including:

- Source of the data (ICAT, web, etc.);
- Code version (obtained from Subversion);
- Experiment description;
- Which study this is a part of; and
- The names of the relevant output files.

Using the information provided, the portal can automatically create a new investigation within the appropriate study. Upon completion of the job, the output data is stamped with a checksum and injected into the institutional repository. This provides a permanent record of the particular experiment.

In some cases there is significant experiment parameter tuning required before any meaningful output is produced (for instance when fixing bugs). In this case it is undesirable to capture every experiment automatically, rather after the fact the practitioner may choose to capture an experiment. This is done by providing the same information as previously, except this investigation will be given a flag to indicate that it was not an automatic capture.

Furthermore, after an investigation has been captured, the practitioner may wish to annotate this run with notes about motivation for certain design or parameter decisions.

5.4 Command Line Assistant

A command line assistant is provided to assist with jobs run in batch mode, such as cluster environments. This can be used to call the actual application, and will produce an XML document in the analysis schema. This will need to be completed by the user. All other information is supplied automatically by the system, such as environment, input files and parameters. The resulting document is then submitted to the ICAT service along with the output details of the execution.

6 Case Study and Discussion

This section outlines a case study using the proposed DPMS and discusses its effectiveness in managing in silico provenance information.

6.1 Task

Consider the life cycle of a typical data mining exercise. It involves the following steps:

- *Data collection or retrieval*
- *Data conversion and filtering*
- *Processing*

Any of these steps may be performed many times, leading to a tree of descendant investigations originating from the initial dataset. For an institution which has many concurrent and historic lines of investigation, this information is critical when revisiting previous work, or for discovering colleagues who are deriving work from common internal, public or reference datasets.

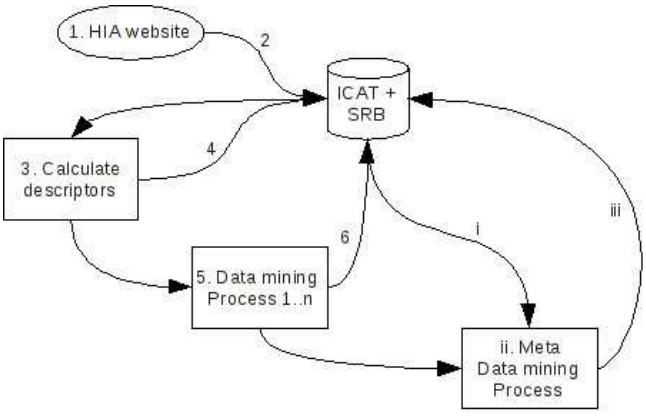

Figure 7: HIA Case Study Workflow

This reference implementation was tested using a PhD student computational chemistry study lines. The study involves taking a public dataset of *human intestinal absorption* (HIA) for pharmacological compounds, and constructing a predictive or explanatory model using characteristics of these compounds. The dataset consists of 3D molecular models of the compounds under investigation, and as their HIA values. The derivation of the molecular characteristics can be performed in many ways, such as enumerating common physical structures, calculating general topological properties, or computing approximations of electrochemical properties. In general, this approach generates a number of numerical or categorical descriptors which are representative of the function of the individual compounds. These descriptors can be used directly in statistical or data mining models.

6.2 Workflow

The study proceeds as outlined in the following workflow (see Figure 7):

1. Download reference data set
2. Register dataset with ICAT
3. Calculate descriptors
4. Store transform with ICAT
5. Perform analysis
6. Store results in ICAT
7. Repeat steps 5 and 6 refining the model parameters

In steps 1 and 2, the outcome is to retrieve this dataset from its public repository and place it in ICAT as a special "retrieval" investigation. This involves uploading the dataset into data storage area (i.e., SRB), which provides a local copy of the data and ensures that subsequent investigations are using the same dataset. This offers protection against changes to datasets in external repositories (both public and private), going unnoticed and corrupting investigations.

Next, steps 3 and 4 transform these input data models into a set of molecular descriptors which can be used within the data mining methods. This transformation is done using the E-DRAGON (Tetko et al. 2005) web portal [12]. It produces a number of output files, which are processed into a single dataset file.

[12] E-DRAGON is the electronic remote version of the well known software DRAGON, which is an application for the calculation of molecular descriptors developed by the QSAR Research Group. These descriptors can be used to evaluate molecular structure-activity or structure-property relationships, as well as for similarity analysis and high throughput screening of molecule databases.

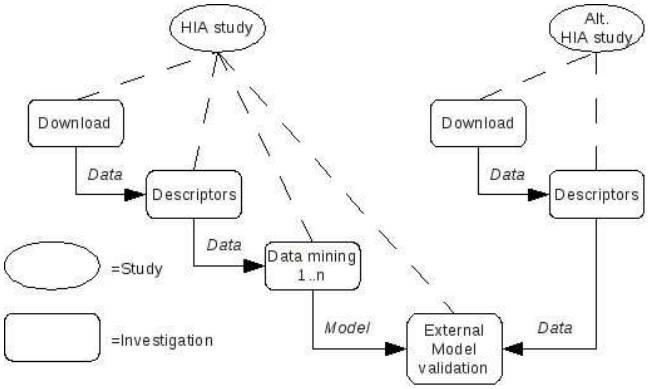

Figure 8: Relationships Between Investigations

This step is registered in ICAT as a transformation investigation on the initial download investigation, having the input dataset being the output dataset from the previous step.

Once the data preparation stage is complete, the actual investigation can proceed. Steps 5 and 6 of the workflow are particularly exploratory, requiring a number of independent investigations to explore different approaches for generating a robust model. This includes executing three different types of model, and using different pre-processing steps. Each of these data mining processes are stored in ICAT.

Conclusions as to the appropriate type of model and pre-processing to use to obtain satisfactory results cannot be drawn from the results of the data mining steps which have just been completed. However, questions still remain as to whether this model is general enough to be run against another HIA dataset and produce reliable results. In this case there is alternative HIA investigation stored in the ICAT, that uses a unique dataset. A new investigation is started, reusing the model produced in the best performing investigation from the previous phase, and using the dataset from the alternative HIA study. This is illustrated as steps I, ii and iii in Figure 7.

6.3 Post Experiment Analysis

Should the performance of the model not be as high as desired, then the practitioner could check the other investigations run against the alternative HIA dataset. This may reveal that the dataset was found to be unreliable, or otherwise not representative of HIA datasets. Being able to visualise these relationships between investigations can be useful.

Figure 8 illustrates the longer term value of recording experimental provenance metadata with a system like ICAT. The schema and its implementation allows for the straightforward retrieval and comparison of results. ICAT also serves as an experiment activity record keeping system which can be utilised as an audit trail of time stamped records. This can be used in a variety of ways including fulfilling supervision and intellectual property requirements.

7 Conclusions

There are numerous inconsistencies with capturing data provenance in the Computer and Information sciences fields. This paper presented a DPMS based on the utility of the ICAT metadata storage service as a viable schema for representing in silico experiments. The system provides a portal interface to integrate ICAT with job execution. We have added a data repository which can handle arbitrary data size, complexity and type. When used in conjunction with a source code versioning system such as Subversion, end users and other parties can determine which version of the code was executed for the experimentation, and whether there have been any modifications to the code since the experimentation occurred.

The proposed system can be practically used to compare, validate and aid in the repetition of historic experiments. It allows for verification of data from external repositories/sources through the use of SRB. SRB is powerful in that it brings together data from multiple disparate sources over a distributed environment. This provides superior facilities for reporting/auditing of experiments and their results.

ICAT provides a uniform format for instrument, sensor and computation data. It also establishes a unified security procedure for verifying and obtaining datasets. This is based on the underlying security provided by SRB.

While the proposed system is based on extensions to ICAT proposed by the ARCHER project (i.e., MCAText), it works independently of the ARCHER system. Users are free to use any existing platform they desire to conduct experiments. The proposed system augments the existing platform with pre and post processes to manage and capture the desired provenance information. This approach is more flexible and versatile than previous data provenance capturing approaches as users do not have to migrate to entirely new platforms in order to harness data provenance recording features.

At present the experiment model does not apply to all forms of investigation (e.g., continuous streams). We aim to address this in future work. Other future work involves exploring the integration of other domain specific metadata requirements. Furthermore, there is scope for expansion of the reporting functionality based on actual requirements. Additionally, we will extend the work through a summary/meta-investigation of the use of a popular dataset. Finally, it is desirable to expand the proposed system to be a service integrated with other computational products.

Acknowledgment

The authors would like to thank the anonymous reviewers for their comments during the peer review process.

References

Altintas, I., Barney, O. & Jaeger-Frank, E. (2006), Provenance Collection Support in the Kepler Scientific Workflow System, *in* 'International Provenance and Annotation Workshop (IPAW)', pp. 118-132.

Androulakis, S., Buckle, A.M., Atkinson, I., Groenewegen, D., Nicholas, N., Treloar, A. & Beitz, A. (2009), ARCHER e-Research Tools for Research Data Management, *in* 'International Journal of Digital Curation', **4**(1).

Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R.H., Konwinski, A., Lee, G., Patterson, D.A., Rabkin, A., Stoica, I. & Zaharia, M. (2009), Above the Clouds: A Berkeley View of Cloud Computing, 'Technical Report. UCB/EECS-2009-28', EECS Department, University of California, Berkeley.

Baru, C., Moore, R., Rajasekar, A. & Wan, M. (1998), The SDSC Storage Resource Broker, *in* 'Cascon '98: proceedings of the 1998 conference of the centre for advanced studies on collaborative

research', IBM Press, Toronto, Ontario, Canada, pp. 5.

Buneman, P., Khanna, S. & Tan, W. (2000), Data provenance: Some basic issues, *in* '20th Conference on Foundations of Software Technology and Theoretical Computer Science'.

Buyya, R., Yeo, C.S. & Venugopal, S. (2008), Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities, *in* '10th IEEE International Conference on High Performance Computing and Communications (HPCC)'.

Cui, Y. & Widom, J. (2000), Practical Lineage Tracing in Data Warehouses, *in* '16th International Conference on Data Engineering', pp. 367–378.

Flannery, D., Matthews, B., Griffin, T., Bicarregui, J., Gleaves, M., Lerusse, L., Downing, R., Ashton, A., Sufi, S., Drinkwater, G. & Kleese, K. (2009), ICAT: Integrating Data Infrastructure for Facilities Based Science, *in* '5th IEEE International Conference on e-Science', pp. 201–207.

Foster, I. & Carl, K. (1999), *The Grid: Blueprint for a New Computing Infrastructure*, 'Morgan Kaufmann Publishers', ISBN 1–55860–475–8.

Foster, I., Vckler, J., Wilde, M. & Zhao, Y. (2002), Chimera: A Virtual Data System for Representing, Querying, and Automating Data Derivation, *in* '14th International Conference on Scientific and Statistical Database Management (SSDBM)'.

Fran, B., Hey, A.J.G. & Fox, G.C. (2003), *Grid Computing: Making The Global Infrastructure a Reality*, Wiley. ISBN 0-470-85319-0.

Frew, J. & Bose, R. (2001), Earth System Science Workbench: A Data Management Infrastructure for Earth Science Products, *in* '13th International Conference on Scientific and Statistical Database Management', pp. 180–189.

Ludscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., Lee, E.A., Tao, J. & Zhao, Y. (2005), Scientific Workflow Management and the KEPLER System, *in* 'Concurrency and Computation: Practice & Experience', Wiley Interscience. DOI: 10.1002/cpe.994.

Myers, J., Pancerella, C., Lansing, C., Schuchardt, K. & Didier, B. (2003), Multi-Scale Science, Supporting Emerging Practice with Semantically Derived Provenance, *in* 'ISWC workshop on Semantic Web Technologies for Searching and Retrieving Scientific Data'.

Oinn, T., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, M., Carver, T., Glover, K., Pocock, M.R., Wipat, A. & Li, P. (2004), Taverna: a tool for the composition and enactment of bioinformatics workflows, *in* 'Bioinformatics', **20**(17), Oxford University Press, London, UK, pp. 3045–3054.

Pancerella, C., Hewson, J., Koegler, W., Leahy, D., Lee, M., Rahn, L., Yang, C., Myers, J.D., Didier, B., McCoy, R., Schuchardt, K., Stephan, E., Windus, T., Amin, K., Bittner, S., Lansing, C., Minkoff, M., Nijsure, S., Laszewski, G.v., Pinzon, R., Ruscic, B., Al Wagner, Wang, B., Pitz, W., Ho, Y.L., Montoya, D., Xu, L., Allison, T.C., Green, W.H. & Frenklach, M. (2003), Metadata in the collaboratory for multi-scale chemical science, *in* 'Dublin Core Conference'.

Simmhan, Y.L., Plale, B. & Gannon, D. (2005), A survey of data provenance in e-science, *in* 'Special Interest Group on Management of Data Record (SIGMOD Record)', **34**(3), pp. 31–36.

Stevens, R.D., Robinson, A.J. & Goble, C.A. (2003), myGrid: Personalized Bioinformatics on the Information Grid, *in* 'Bioinformatics', **19**(1), Oxford University Press, London, UK, pp. 302-304.

Tetko, I.V., Gasteiger, J., Todeschini, R., Mauri, A., Livingstone, D., Ertl, P., Palyulin, V.A., Radchenko, E.V., Zefirov, N.S. & Makarenko, A.S. (2005), Virtual computational chemistry laboratory - design and description, *in* 'Journal of Computer Aided Mol. Des.' Vol 19, pp. 453–463.

Trevathan, J., Atkinson, I., Read, W., Johnstone, R., Bajema, N. & McGeachin, J. (2010), Establishing Low Cost Aquatic Monitoring Networks for Developing Countries, *in* 'the International Conference on Wireless Communications and Information Technology in Developing Countries', pp. 37–48.

Widom, J. (2005), Trio: A System for Integrated Management of Data, Accuracy, and Lineage, *in* 'Conference on Inovative Data System Research (CIDR)', pp. 262–276.

Yu, J. & Buyya, R. (2005), A Taxonomy of Workflow Management Systems for Grid Computing, 'Technical Report', GRIDS-TR-2005-1, Grid Computing and Distributed Systems Laboratory, University of Melbourne.