

# Supercomputing Center Management using AIRS

Robert A. Ballance, Jared Galbraith, and Roy Heimbach  
High-Performance Computing, Education and Research Center  
The University of New Mexico  
Albuquerque, New Mexico

No Institute Given

## 1 Introduction

Running a large University supercomputing center teaches many lessons, including the need to centralize data collection and analysis, to automate system administration functions, and to enable users to manage their own projects. Albuquerque Integrated Reporting System (AIRS) evolved in response to these pressures.

### 1.1 The setting

The High-Performance Computing, Education, & Research Center (HPCERC)<sup>1</sup> at the University of New Mexico, is a leading academic site for high-performance computing and scientific programming. When Los Lobos, the Center's Linux-based 512-processor supercluster, came on-line in June, 2000, HPCERC ranked among the largest five academic supercomputing centers in the U.S. The Center also operates a 128 processor production Linux supercluster on behalf of the National Computational Science Alliance (NCSA). Locally, the Center maintains three 32-processor research superclusters from IBM and VA Linux Systems, Silicon Graphics visualization servers, a novel visualization cluster for parallel graphics, a visualization laboratory, and a cluster of workstations, all in support of a broad range of scientific processing. The Center owns and operates two Access Grid studios [1, 2] for use in Internet-based teleconferencing research. All of our production and research systems run either Linux, BSD, AIX, or IRIX, with Linux the dominant OS. Staff members choose from a menu of Linux, BSD, Mac, or Windows as their primary workstation OS.

Federally-funded research scientists from across the U.S. use HPCERC resources as part of their research programs. Within UNM, HPCERC assists over 20 associated faculty and their students, from the Colleges of Arts and Sciences, Engineering, Fine Arts, and the School of Medicine in their

---

<sup>1</sup> Formerly known as Albuquerque High-Performance Computing Center (AHPCC).

research and teaching programs. Nationally, HPCERC partners with other Alliance member institutions in developing, deploying, and promoting high-performance computing technologies.

## 1.2 The Problem

Imagine this, then:

- Multiple clusters, each with their own accounting systems;
- Distinct and separate naming domains (`alliance.unm.edu` and `ahpcc.unm.edu`) within a single facility;
- Required daily usage reporting via electronic data transfer to our sponsors (NCSA);
- Dozens of users and projects, many administered remotely;
- Active and historical data scattered in multiple locations, including flat files, desktop databases, on paper.

## 1.3 The Solution: Take a deep breath of ... AIRS

What's a manager to do? Reach for the open source and get to work. Two years later, the result is AIRS, an open source solution for managing today's supercomputing center. The name was originally just the "Integrated Reporting System" — chosen for its acronym and its ability to insinuate itself into virtually every aspect of Center management.

Core features of the system, including daily reporting of usage information, have been online since early 2001. Other features are in various stages of deployment. We are working to simplify installation and deployment at other sites. Distribution tarballs will be available at [airshq.hpcerc.unm.edu/](http://airshq.hpcerc.unm.edu/).

This article provides a brief overview of AIRS: its capabilities, design, and evolution.

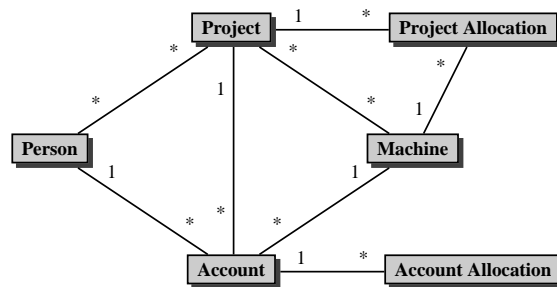
## 2 Design Pressures

The development of AIRS began several months after our first large Linux cluster (RoadRunner) came online as a production computing environment. The RoadRunner cluster was funded jointly by NCSA and UNM. As a production machine, we needed to be able to account for the usage of the machine. For example, daily usage had to be reported to NCSA. Not all usage mind you, but just the usage for those projects with NCSA allocations. However, as soon as we could gather information for our sponsors, we immediately began to use it ourselves. It is a comfort to receive mail each day summarizing the usage for

the day before. It was a small leap to add daily, weekly, biweekly, monthly, and quarterly reports.

In addition, membership in the National Computational Science Alliance required that we add a remote account management interface to enable other Alliance member sites to manage local accounts. This feature is separable from the rest of the system.

AIRS usage accounting is organized around the concepts of PERSONS, PROJECTS, ACCOUNTS, ALLOCATIONS and MACHINES as shown in Figure 1. To interpret the figure, note for example that each PROJECT can have zero or more (\*)PROJECT ALLOCATIONS, but each PROJECT ALLOCATION relates to a single (1) PROJECT. Figure 1 shows the “domain-level” relationships supported by AIRS. The actual SQL tables are quite different, of course!



**Fig. 1.** Key domain-level relationships

PERSONS can have multiple accounts affiliated with multiple PROJECTS. An ACCOUNT is specific to a MACHINE. Each PROJECT has one or more PROJECT ALLOCATIONS of CPU-hours specific to given MACHINES. Usage is debited against the PROJECT ALLOCATION. Once an allocation’s hours are used, a new PROJECT ALLOCATION can be granted.

To simplify user account management on the machines, the ACCOUNT table is further factored to include a LOGIN record. It is the LOGIN table that handles actual machine-level (NIS, LDAP) login, group, and initial password information. The initial passwords are maintained (encrypted) within the database so that logins can be reset to their default password if necessary.

The system is designed to accommodate the distribution of PROJECT ALLOCATIONS into individual ACCOUNT ALLOCATIONS. However, as currently

implemented, all of the ACCOUNTS for a given project simply draw on the total available allocation for the PROJECT.

Usage data is gathered from several sources. The primary source, for each cluster, is the data gathered by the scheduling system. In clusters like ours, a scheduling system manages the individual compute nodes, handles requests for nodes, and actually launches the jobs. Thus, it is the scheduler that knows job-related accounting information. All of the available schedulers provide accounting data, either as files or as a connection to an SQL database. The job data is quite different from Unix process accounting data.

In principle, we can add real-time connections between the scheduling systems and the accounting database. For example, the scheduling system could query AIRS to see if an account has sufficient remaining hours to run a job. Alternatively, the scheduling system could be modified to add job accounting information directly into AIRS. Neither connection is implemented in Version 1.0.

Creating a flexible design that would link PERSONS to ACCOUNTS, ACCOUNTS to PROJECTS, and ACCOUNTS to MACHINES was the first step. From there, AIRS has grown to accommodate a number of other features. For example:

- A web interface provides the ability to create, manage, and review the information.
- A workflow model uses the web interface to route approvals for project, account, and allocation requests to the proper members of the Center staff. When all approvals are in place, it takes only a few clicks to actually create an account in the systems domain, send the email, and print paper letters (that include sensitive information) for mailing.  
The approval mechanism regularly reminds participants when their inputs are required to further the process.
- Each PROJECT is managed by one or more Principal Investigators (PIs). We wanted to provide regular usage reporting to each PI concerning the usage on their project. PIs can now set up to receive usage information for their projects by email. Individual users can also set up reporting for their own accounts. Reports can be delivered either by electronic mail or via the Web.
- To simplify the maintenance of the system, we wanted the PIs to be able to add or remove users from their projects. A Web interface for project management is provided.
- To meet NCSA requirements, AIRS provides a transaction-based interface to NCSA that allows the primary site to create, modify, and delete PERSONS, PROJECTS, ACCOUNTS, and PROJECT ALLOCATIONS.

- To reduce clutter, more and more system information is migrating into the database. For example, since all the machines are present, it became natural to add inventory, vendor, and serial number information.
- To support Globus-based grid computing [8], AIRS maintains relationships between local logins and the [DN??]'s used to identify users within the Computational Grid [7].

Today, the AIRS database manages over 70 tables that represent over 500 data columns (including references). The accounting data includes over 140,000 records for individual job runs. The web interface is substantially complete, including the workflow capabilities needed for approving projects and accounts. PI-management of projects is also substantially complete. The actual database design has been stable for many months, except in areas (like the trouble ticketing system) where there is active development. Recently, we've started to add trouble tickets and help desk capabilities. We are also in the process of adding time tracking to AIRS so that we can associate consulting time to PROJECTS, and changes to MACHINES. Future work is discussed in Section ??.

## Glossary

- Albuquerque High-Performance Computing Center (AHPCC)** , p. 1.
- Albuquerque Integrated Reporting System (AIRS)** A centralized management system for supercomputing centers, p. 1.
- High-Performance Computing, Education, & Research Center (HPCERC)**  
, p. 1.
- National Computational Science Alliance (NCSA)** , p. 1.
- Principal Investigator (PI)** , p. 4.

## References

1. Access grid home page. [www.accessgrid.org](http://www.accessgrid.org).
2. Robert A. Ballance, Thomas P. Caudell, John Greenfield, and Rick Stevens. The National Computational Science Alliance Access Grid: An Internet-based collaboration tool augmented by high-performance computing. In *DoD High-Performance Computing User's Conference*, Albuquerque, NM, June 2000.
3. Tom Christiansen, Nathan Torkington, and Larry Wall. *Perl Cookbook*. O'Reilly & Associates, Inc., Sebastapol, CA, 1998.
4. Damian Conway. *Object Oriented Perl*. Manning Publications Co., Greenwich, CT, 2000.
5. Alligator Descartes and Tim Bunce. *Programming the Perl DBI*. O'Reilly & Associates, Inc., Sebastapol, CA, 2000.
6. Paul DuBois and Michael Widenius. *MySQL*. New Riders Publishing, Indianapolis, 1999.
7. Ian Foster and Carl Kesselman, editors. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan-Kauffman, 1998.
8. Globus project home page. [www.globus.org](http://www.globus.org).
9. Joseph N. Hall and Randal L. Schwartz. *Effective Perl Programming: Writing Better Programs with Perl*. Addison Wesley Longman, Reading, MA, USA, 1998.
10. Larry Wall, Tom Christiansen, and Jon Orwant. *Programming Perl*. O'Reilly & Associates, Inc., Sebastapol, CA, third edition, 2000.