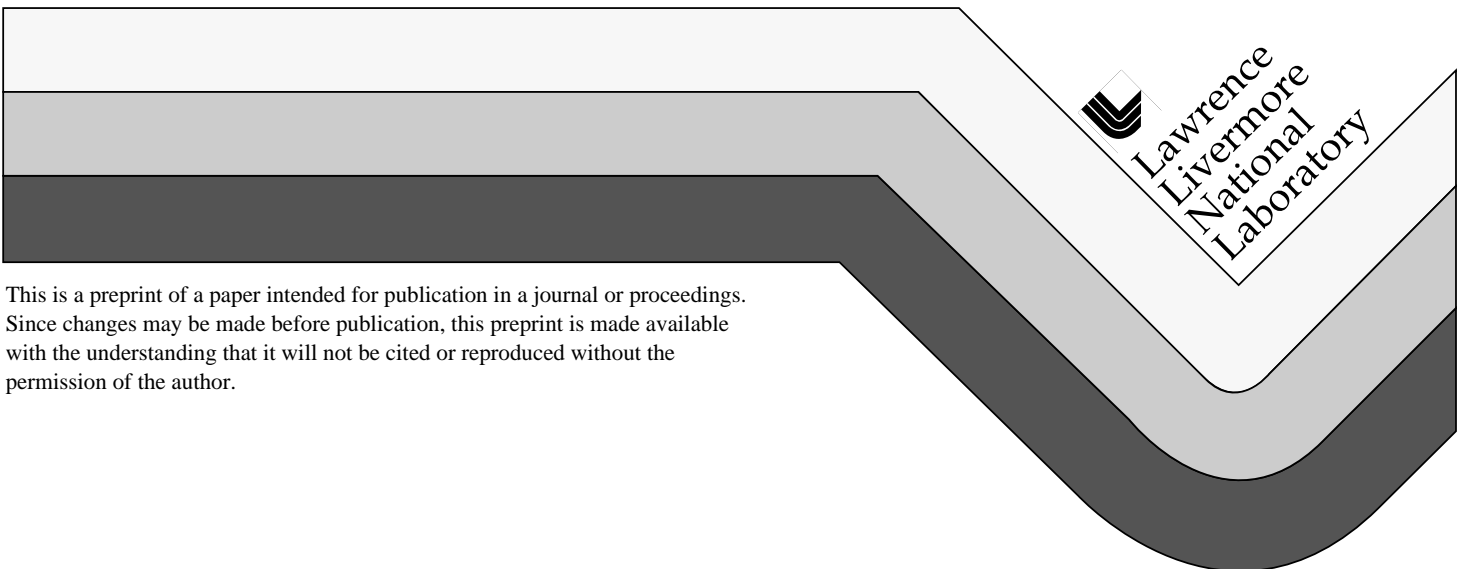


# SimTracker-Using the Web to Track Computer Simulation Results

J. Long  
P. Spencer  
R. Springmeyer

This paper was prepared for submittal to the  
*1999 International Conference on Web-Based Modeling and Simulation*  
*San Francisco, CA*  
*January 17-20, 1999*

August 26, 1998



This is a preprint of a paper intended for publication in a journal or proceedings. Since changes may be made before publication, this preprint is made available with the understanding that it will not be cited or reproduced without the permission of the author.

#### DISCLAIMER

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

# **SimTracker - Using the Web to Track Computer Simulation Results**

**J. Long, P. Spencer, R. Springmeyer  
Lawrence Livermore National Laboratory**

## **ABSTRACT**

Large-scale computer simulations, a hallmark of computing at Lawrence Livermore National Laboratory (LLNL), often take days to run and can produce massive amounts of output. The typical environment of many LLNL scientists includes multiple hardware platforms, a large collection of eclectic software applications, data stored on many devices in many formats, and little standard metadata, which is accessible documentation about the data. The exploration of simulation results typically proceeds as a laborious process requiring knowledge of this complex environment and many application programs.

We have addressed this problem by developing a web-based approach for exploring simulation results via the automatic generation of metadata summaries which provide convenient access to the data sets and associated analysis tools. In this paper we will describe the SimTracker tool for automatically generating metadata that serves as a quick overview and index to the archived results of simulations.

The SimTracker application consists of two parts - a generation component and a viewing component. The generation component captures and generates calculation metadata from a simulation. These metadata include graphical snapshots from various stages of the run, pointers to the input and output files from the simulation, and assorted annotations describing the run. SimTracker generation can be done either during a simulation or afterwards. When integrated with a code system, SimTracker does its work on the fly, allowing the user to monitor a calculation while it is running.

The viewing component of SimTracker provides a web-based mechanism for both quick perusing and careful analysis of simulation results. HTML is created on the fly from a series of Perl CGI scripts and metadata extracted from a database. A variety of views are provided, ranging from a high-level table of contents showing all of one's simulations, to an in-depth results page from which numeric values can be extracted and analysis tools can easily be launched. Annotations can be associated with a calculation at any time, allowing an end-user to customize the summary pages with titles, abstracts, and pointers to related information, for example.

In this paper, we will present an overview of the design, implementation, and operational aspects of the SimTracker application. We will also discuss how it is being deployed in the environment of the Accelerated Strategic Computing Initiative [1]. SimTracker was designed as an extensible application that we are now adapting to use with several simulation codes.

**Keywords:** Computer simulation, electronic notebook, metadata, simulation monitoring

## INTRODUCTION

SimTracker is applicable to a wide range of computing and work environments, but this paper focuses on the environment of our target users: physicists at the Lawrence Livermore, Los Alamos, and Sandia national laboratories. The LLNL computing environment includes workstation clusters, symmetric multiprocessor (SMP) clusters, massively parallel machines, high speed parallel storage architectures, disk and tape archives, local area networks, and internet wide area networks (WANs), all connected via very high speed data networks and switches.

Physicist end users perform extremely complex and large-scale simulations to model physical phenomena. They use a suite of computer applications, called a simulation code system, that can take from hours to weeks to run and that can generate massive amounts of data. Typical components of a code system include a generation code that creates the initial digital representation of the simulation based on user-supplied grid and material information; a physics code that performs the actual simulation and which generates the majority of the data; and a handful of utility codes which perform tasks ranging from data translation to scientific visualization and analysis.

The nature of the phenomena being studied often makes it necessary to use multiple code systems, each with specialized modeling capabilities and grid characteristics. Physicists compare their many sets of results with other computational results and with experimental data, in an effort to understand how well their simulations predict physical events. A user typically runs dozens or even hundreds of variations. Managing the resulting output has been handled on an ad hoc basis by each user. Most use hardcopy notebooks to keep track of their computational plans and results. Their data is scattered across multiple platforms, uses a variety of formats, is difficult for them to keep organized, and is typically inaccessible to their colleagues. Managing on-line calculation components, from input decks to results files, is currently a very manual process. Files are moved from one system to another with FTP, analysis tools are launched manually, and the metadata that does get created is typically hand-written. With the advent of the massive computing capability provided by the national laboratories' current accelerated program, managing calculation results in this fashion becomes untenable.

Our approach to this problem is to provide users with a familiar, web-based, paradigm for accessing their simulation components, from input parameters to results. Metadata is generated automatically where possible, and mechanisms for easily adding personal annotations are supported. All metadata is stored in a database suitable for subsequent searching. The mechanics of transferring data from one host to another and of launching analysis tools are hidden from the user. Most importantly, with SimTracker the user can easily find both recent and legacy results, and then simply view and analyze all pertinent datafiles associated with the run. Furthermore, other scientists can use SimTracker to view the results of colleagues.

## BACKGROUND AND RELATED WORK

Techniques for monitoring and documentation of simulation runs or experimental data collections have been developed for several applications areas, such as chemistry and biology. Ecce' is an integrated environment for molecular modeling and simulation activities, developed at PNL [2]. LBL has with other partners developed a collaborative aimed at facilitating remote collaboration and sharing of spectro-microscopy data [3]. Electronic notebooks, such as the Virtual Notebook System [4] can replace traditional

hand-written lab notebooks and add new functionality, such as searching, hyperlinks, and WYSIWYG editing. SimTracker shares some of the features of electronic notebooks, such as electronically collecting related information together, and associating metadata with data. It differs in that it has been tailored to run within the framework of large physics code systems and to produce updated summaries as the code runs. It meets a specific goal of tracking a simulation and it also fits into a larger system of metadata tools, in which a database is used to store and access metadata created by SimTracker and other tools. Our goal was a flexible architecture that can be adapted to many code systems and can work within that framework of tools.

The SimTracker work grew out of the Intelligent Archive project [5,7] at LLNL, and is now being sponsored by the Scientific Data Management area in ASCI. The metadata formats used are heavily influenced by the efforts of an archiving project which has defined a set of metadata fields for the exchange of information and data within a subset of the Department of Energy complex [6]. The remainder of this paper describes in detail our design and implementation of the SimTracker tool for solving this problem.

## DESIGN

The SimTracker application consists of two distinct parts - a generation component and a viewing component. (See Figure 1.) The generation component runs on the large computing platforms along with the simulation codes. It is responsible for tracking all files associated with the code run, for copying these files to the mass storage system, and for generating meaningful metadata such as graphical snapshots from the code output. The viewing component is completely separate and is responsible for creating the web-based interface to the results.

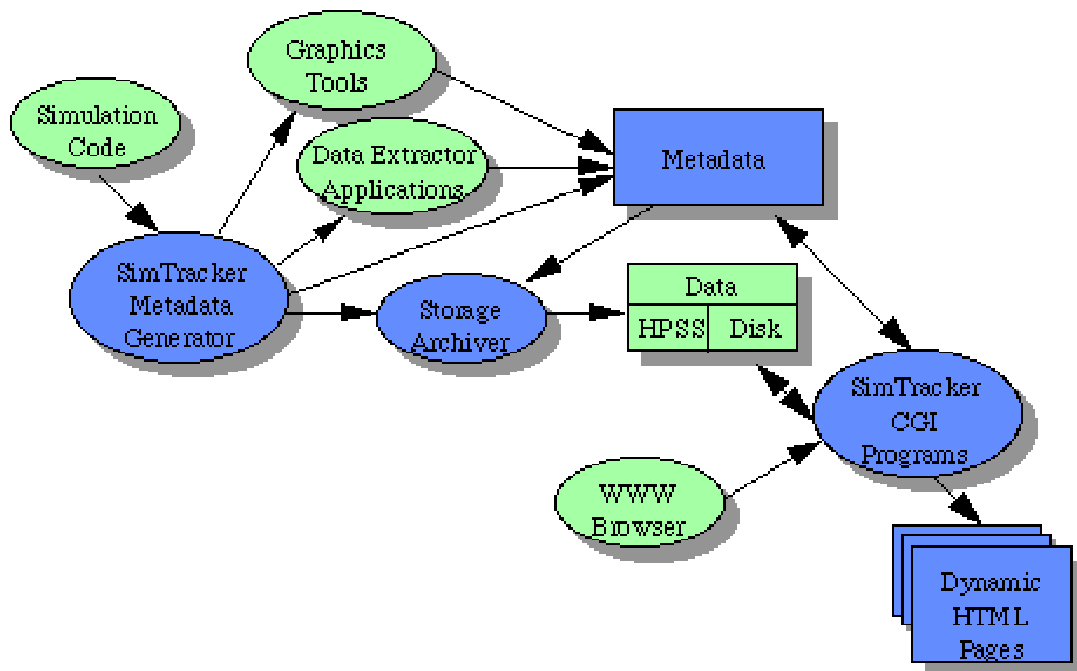






Figure 1: SimTracker Architecture Diagram

An important design consideration for SimTracker was to support the creation of calculation summaries both as the simulation is running, and afterwards. This was accomplished by splitting the generation component into three pieces - “begin”, “cycle”, and “end”, to reflect the times during which files can be collected and operations performed. When integrated with a code system, SimTracker allows a user to track calculations as they are running, by viewing updated summaries including graphical snapshots.

A key requirement for SimTracker was to provide web-based viewing of calculation summaries. In an effort to gather user feedback, our quick and simple first prototype created static HTML documents. Armed with a full set of user requirements, we refined the design and migrated to a CGI-based dynamic HTML solution. This gives us a great deal of flexibility in customizing and extending the capabilities of SimTracker.

The viewing component of SimTracker consists of a collection of CGI programs which create HTML calculation summaries on the fly, giving users a quick glance at their results and a way to initiate further analysis. The contents of the web pages are assembled from information in a metadata database populated by the generation component. The layout of the pages is controlled by customizable template files. The view provided to the user is hierarchical (See Figures 2,3,4). The top level, or table of contents, shows all calculations at a glance. The next level down is a results page consisting of information about a specific calculation. The final level down, the cycle page, consists of information about a particular state in a calculation.

**Calculation Index**  
Jeffery W. Long (07-07-1998 14:36)

Data	Project	ID	Date	Title	Comment	Run Stat
	Hydro Experiment 1410B	mmm	07/07/98 13:51	Jet Formation Calculation	Calculation of the jet formation problem in preparation for...	Cycle 1349
	Container Drop Test	dip	07/03/98 14:04	COYOTE Transportation Container Drop Test	CORBA coupled COYOTE/ALEGRA simulation of a tranporta...	Done
	Tecolote	shl	07/01/98 14:10	Tecolote Spherical Shells	Spherical shells test problem run with Tecolote. ...	Done
	Compaction Study	cth	06/26/98 14:13	CTH Run	CTH calculation of particle compaction. ...	Done

Compare House Keeping

**SimTracker**  
INTELLIGENT ARCHIVE

Figure 2: Table of contents page showing entries for all simulations.

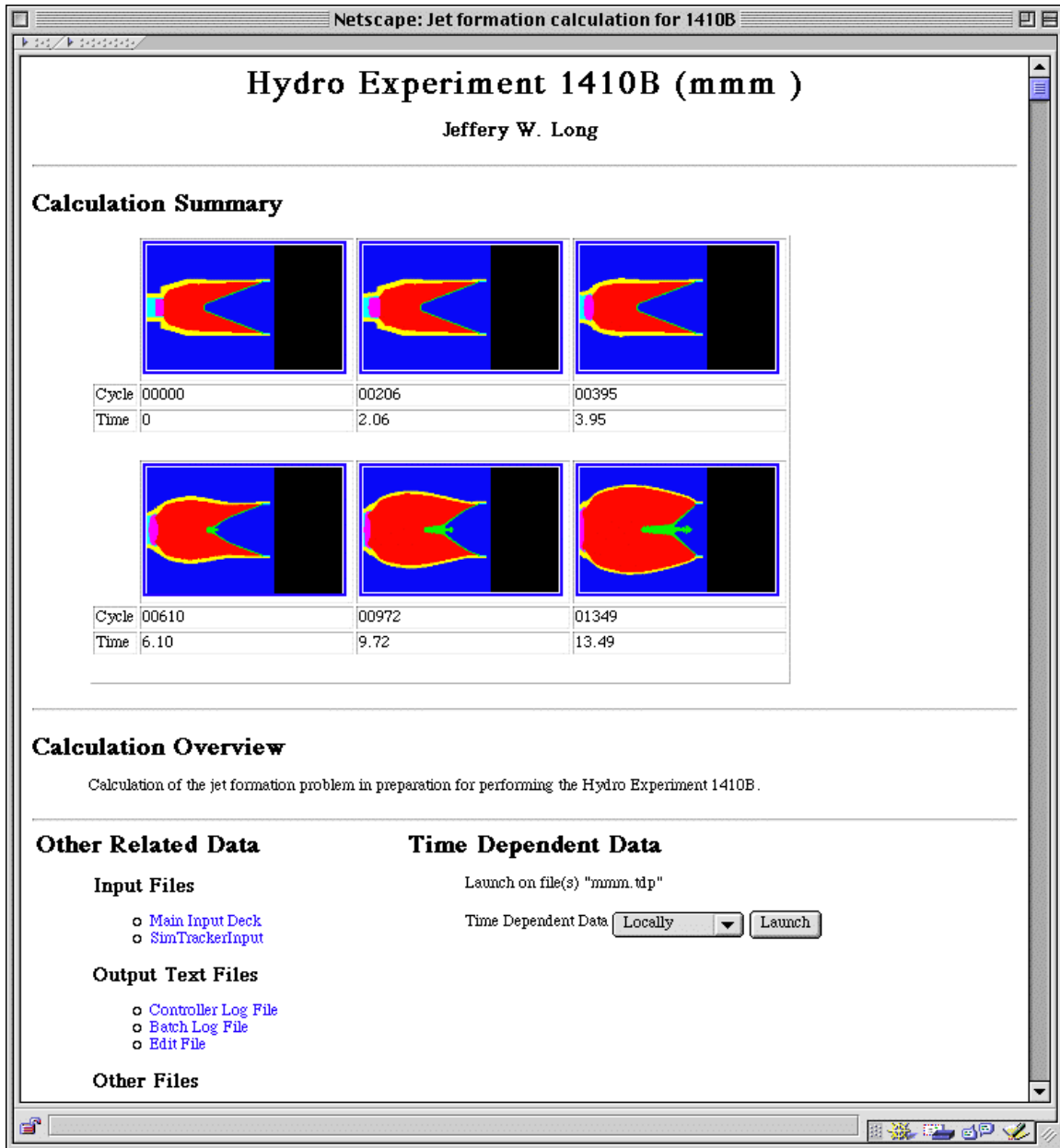


Figure 3: Result page showing summary of one simulation

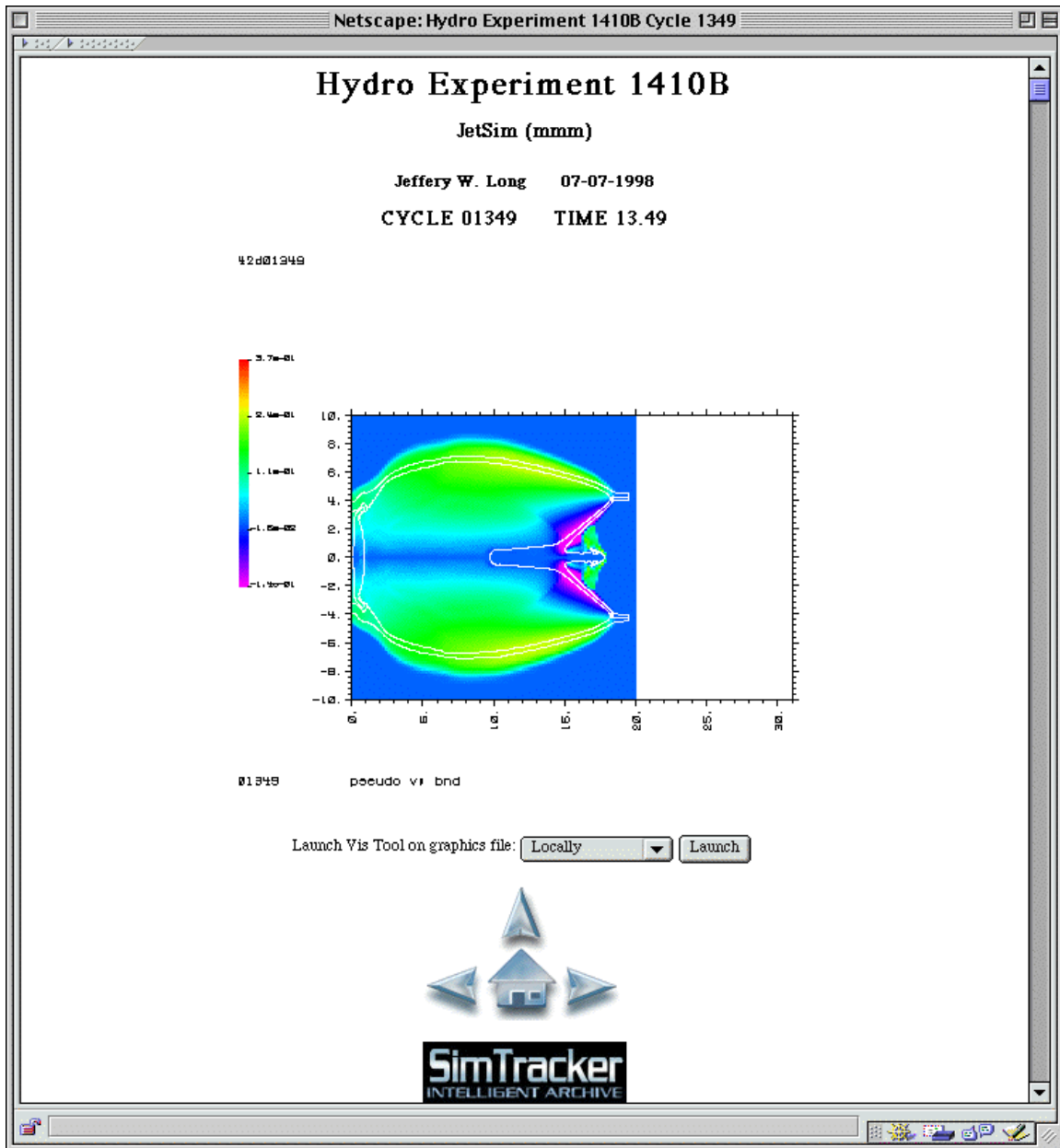


Figure 4: Cycle page showing details from a specific cycle, or state.

## IMPLEMENTATION

The generation component is implemented as a set of Perl scripts. An early prototype was implemented in Expect, but difficulties operating under the local batch job submission system required an alternative approach. Perl has lent itself well to this project's needs -- many quick experiments and prototyping ventures, with an underlying evolutionary software design model.

The files to be tracked during the calculation are defined in a file definition template. This document consists of a set of triplets: a unique identifier associated with a file or set of files, a sentinel defining when to collect this set of files -- at the beginning of the run, at the end of the run, during the course of the run, or any combination of these, and a regular expression used to match the actual file names created by the calculation. These



expressions include special variables which will match a job identifier associated with the run, the current cycle number, and so on. Implementing the file definition template was a major step towards generalization of SimTracker. By defining the files to track in a general, external fashion, SimTracker is much easier to deploy with new code systems.

Each of the three generation components consists of general code which is independent of the code system. Any custom or other code system-specific development is done in separate modules pulled in when appropriate. We have found approximately 80% of SimTracker coding to be general across the code systems with which we've worked. The remaining code is put into code system-specific modules which handle idiosyncrasies such as collecting files that cannot be specified via the file definition template. The code system-specific modules also will invoke the appropriate metadata generation tools provided by the code system. These tools include visualization applications which are used for creating thumbnail images.

Additional operations performed by the generation components include running data translation and conversion utilities otherwise done manually by the user, compressing data files, creating metadata describing the calculation and where it is stored, running post-processors to create visualizations, and copying all tracked result data and metadata to archival storage.

The CGI-based viewing components are also implemented in Perl, due to the language's strong support for string processing and rapid prototyping. Like the generation component, template files are also used by the viewing scripts. They control the layout of the dynamic web pages created for the users. These templates are HTML documents with additional SimTracker-defined tags embedded throughout. These tags allow one to specify information such as MIME types for files to be downloaded, execute lines for remote launching of analysis tools, which files to provide links to in the web pages, and labels to use for individual links.

## OPERATION

The begin script performs a number of important set-up steps. It first processes the file definition template to generate a set of file identifiers and associated regular expressions. It then sets up initial temporary directories and copies files from the calculation which do not change during the course of a run. The user's input command file is an example of an unchanging file which can be collected at the outset.

The cycle script is invoked periodically during the calculation. Each time it is invoked it will generate thumbnail images and larger user-defined plots, and it will collect all files requested on the cycle interval. The number and frequency of these invocations is controlled by the code system itself. Preferably, the code system will provide a mechanism for the user to control these parameters.

The end script is invoked at the conclusion of the calculation. Files requested to be copied at this time, typically log files, are processed. This is also the time to run pertinent post-processing applications such as time dependent data extractors.

The user typically will begin a viewing session by launching the CGI script which renders the table of contents from metadata stored in the SimTracker database. Because there are no static HTML documents on disk, all references in the pages are to other CGI scripts. The final column in the table of contents shows the run status of the calculation. This field can contain strings such as "Done" for completed jobs or "Cycle *n*" if the job

is still underway. In the latter case, the thumbnail image shown in the table of contents is periodically updated to show the current state of the calculation. Once the job has completed, the thumbnail reverts to the initial problem state, as requested by the majority of our users.

Other fields in the table of contents include job identification, title, date of run, and comment. Other than the date, all fields are editable by the user. Initial values can be supplied at the beginning of the simulation, but default values are supplied if not present. The web interface allows the user to easily update these values at any time.

All of the CGI scripts have a built-in search path to use when looking for files. The metadata for a calculation defines the identifier used for the job, as well as on-line and off-line locations where they were stored. Small files such as thumbnails are kept initially in the user's home directory. All files from a calculation are initially stored in a scratch partition on the machine where the calculation was run. Likewise, all files are copied to archival storage. If when trying to retrieve a file, SimTracker will look first in the home directory, then in the scratch partition, and if neither of those are successful, the pertinent data is automatically moved from storage.

## FUTURES

SimTracker presently has been deployed with two different code systems. In the coming weeks and months we expect to deploy with two more. And this number will continue to grow in the coming year. The feedback received to this point has been extensive and very positive. This feedback has been crucial to establishing the direction for new work.

One request from a number of users is the ability to incorporate additional information with calculation summaries. In particular, users would like to add analysis data, such as plots, and related experimental data. This ability would be a step towards creating more of a notebook environment for the user. This is an area we plan to investigate aggressively in the coming year. We have also begun assembling a mechanism for comparing simulations. At present this is rudimentary, but we hope to devise a more sophisticated comparison capability. Finally, we are currently using a simple DBM file for storing metadata. We expect in the next year to migrate to a commercial DBMS.

This work was performed under the auspices of the U.S. Dept. of Energy at LLNL under contract no. W-7405-Eng-48.

## REFERENCES

- [1] Accelerated Strategic Computing Initiative: Overview, LANL Technical Report, LALP-97-175. November 1997. Available at [http://www.llnl.gov/asci/sc97fliers/lanl/ASCI\\_Overview.pdf](http://www.llnl.gov/asci/sc97fliers/lanl/ASCI_Overview.pdf).
- [2]. Extensible Computational Chemistry Environment. Available at <http://www.emsl.pnl.gov:2080/docs/ecce/ecce.html>
- [3]. D.A. Agarwal, S.R. Sachs, W.E. Johnston, "The Reality of Collaboratories", Computer Physics Communications vol. 110, issue 1-3 (May 1998), pages 134-141.
- [4]. Burger, A. M., "The Virtual Notebook System", Proceedings of Hypertext'91, pp. 395-401.

[5] Intelligent Archive: Integrated Data Access and Organization for Scientists, LLNL Technical Report, UCRL-TB-118571, November 1995. Available at <http://www.llnl.gov/ia/>.

[6] Bruce Lownsbery and Helen Newton, The Key to Enduring Access: Multi-organizational Collaboration on the Development of Metadata for Use in Archiving Nuclear Weapons Data, First IEEE Metadata Conference, April 16-18, 1996, NOAA Auditorium, Silver Spring, Maryland.

[7] R. Springmeyer, N. Werner, J. Long, "Mining Scientific Data Archives through Metadata Generation", First IEEE Metadata Conference, April 16-18, 1996, NOAA Auditorium, Silver Spring, Maryland.