CTWatch QUARTERLY

ISSN 1555-9874 AVAILABLE ON-LINE AT http://www.ctwatch.org/quarterly/ VOLUME 3 NUMBER 4 **NOVEMBER 2007**

Software Enabling Technologies for Petascale Science

GUEST EDITOR **FRED JOHNSON**

CyberInfrastructure Partnership

Cyberinfrastructure Technology Watch
http://www.ctwatch.org/

CTWatch QUARTERLY

ISSN 1555-9874 VOLUME 3 NUMBER 4 NOVEMBER 2007

Software Enabling Technologies for Petascale Science

GUEST EDITOR **FRED JOHNSON**

Introduction

Fred Johnson
Acting Director, Computational Science Research & Partnerships (SciDAC) Division
Office of Advanced Scientific Computing Research
DOE Office of Science

The critical importance of *enabling software technology* for leading edge research is being thrown into sharp relief by the remarkable escalation in the application complexity, quantities of data that scientists must now grapple with, and the scale of the computing platforms that they must use to do it. The effects of this ongoing complexity and data tsunami as well as the drive toward petascale computing are reverberating throughout every level of the software environment on which today's vanguard applications depend – through the algorithms, the libraries, the system components, and the diverse collection of tools and methodologies for software development, performance optimization, data management, and data visualization. It is increasingly clear that our ability today to adapt and scale up the elements of this common software foundation will largely determine our ability tomorrow to attack the questions emerging at the frontiers of science.

Nowhere is this connection between scalable software technology and breakthrough science more evident than in the articles of this issue of *CTWatch Quarterly*. Each one offers an informative and stimulating discussion of some of the major work being carried out by one of the *Centers for Enabling Technologies (CET)* of the Department of Energy's wide ranging and influential SciDAC program. The joint mission of the CETs is to assure that the scientific computing software infrastructure addresses the needs of SciDAC applications, data sets and parallel computing platforms, and to help prepare the scientific community for an environment where distributed, interdisciplinary collaboration is the norm. Each CET is a multidisciplinary team that works closely with one or more of SciDAC's major application teams. Each one focuses its attention on the mathematical and computing problems confronting some major aspect of software functionality, such as distributed data management, application development, performance tuning, or scientific visualization. Making necessary progress in any of these areas requires the collective effort from the national (and international) research community, yet as these articles show, working in the context of SciDAC research has enabled these CETs to make leadership contributions.

The articles here reflect the rich diversity of components, layers and perspectives encompassed by SciDAC's software ecosystem. They are grouped together according to the aspect of the problem of scalability they address. One group of articles focuses on the software innovations that will be necessary to cope with multiple order of magnitude increases in the number of processors and processor cores on petascale systems and beyond; another set focuses on the data management challenges spawned by the exponential growth in the size of tomorrow's routine data sets; and finally, CETs dedicated to scientific visualization address the need to understand increasingly large and complex data sets generated either experimentally or computationally. The articles in this issue of *CTWatch Quarterly* follow these groupings.

We begin with a discussion (Gibson et al.) of the future requirements for fault tolerant computing from the leaders of the *Petascale Data Storage Institute (PDSI)*. Given the surprising consequences that scaling up often introduces, it seems to strike an appropriate note – sobriety based on experience. The PDSI team has been collecting and analyzing data on failure rates from contemporary HPC systems in an effort to understand the impact that scaling up to systems with millions of hardware elements will have on successful application execution in general, and on the requirements for next generation storage systems, in particular. The results of their timely analysis are thought provoking. They show generally that as systems scale up, conventional approaches to fault tolerance based on familiar check-point and restart may break down along various fronts because the size and frequency of the checkpoints that must be taken on massive systems makes the process unsustainable. Their analysis makes it clear that systems research in this area is destined to become more and more critical.

Three of the CETs focus on issues of software development and maintenance that are raised by the extreme demands of next generation applications and the requirements of the HPC systems on which they must run. The scope of the *Center for Technology for Advanced Scientific Computing Software (TASCS)*, presented in Parker et al., is the most general. For the TASCS group, the increasing scale and complexity of SciDAC applications and systems software is itself a critical problem. They argue that a far higher degree of *modularity* is required in the software that describes the multi-physics, multi-scale simulations that are now being developed. The more stove-piped these applications are, the less smoothly and intelligently they will be able to adapt and innovate to meet the conditions that we know are coming – more parallelism, more data intensity, shorter mean time to failure, and so on. The core techniques, tools, components and best practices of the Common Component Architecture (CCA) that they survey in their article are designed to help solve this aspect of the scalability problem for the broad SciDAC community.

The other two code-oriented CETs – the *Performance Engineering Research Institute (PERI)* and the *Center for Scalable Application Development Software (CScADS)* – focus on application performance and programmer productivity in the context of systems designed with thousands or millions of multicore and/or heterogeneous processors. They share the common goal of providing a tool set for achieving high performance that is as automated and easy to use as possible, allowing researchers to keep their attention focused on the domain science questions at hand. Both have made concerted efforts, through sponsored workshops and direct contact, to engage with and leverage the experience of the SciDAC developer community, with initial emphasis in the areas of Fusion Energy and Combustion. Yet their work emphasizes different, but complementary aspects of the problem. The PERI group (Bailey et al.) builds on a foundation of performance modeling, endeavoring to understand, through systematic empirical testing and analysis, the way real world applications behave on real world systems. The knowledge gained thereby is then used to help guide the application design and development process through a variety of techniques, the more automated the better. By contrast, the CScADS group (Mellor-Crummey et al.) is exploring programming models that make the process of developing well tuned, highly parallel software as easy and efficient as possible by innovatively combining high level languages, scripting languages, compilers and other software tools. As these efforts converge, their collective results hold tremendous promise for the HPC developer community.

The CETs dedicated to scientific visualization have to confront the problem of petascale science from a uniquely important point of view, namely, where the bits meet the mind and the bandwidth is inherently limited. Their task is to find ways to enable scientists to fruitfully apply their observational capabilities, constrained as they are by nature, to some of the world's largest and most complex datasets, using some of the world's most massive and sophisticated computational platforms.

As described in the Bethel, Johnson et al. article, the *Visualization and Analytics Center for Enabling Technologies (VACET)* group is developing solutions to this problem that combine "query-driven" strategies, which pre-filter the data to be visualized for relevance and interest, with "context and focus" user interface designs, which enable scientists to control their field of attention while navigating complex data spaces. The success of this approach obviously depends on finding fast and efficient ways to index and search targeted data sets; VACET is collaborating closely with other centers in researching this problem. The work of the *Institute for Ultrascale Visualization (Ultraviz Institute)*, described in Kwan-Liu Ma's article, also (by necessity) puts the question of interface design at the center of its research agenda, especially for cases requiring the exploration of time-varying multivariate volume data. The Institute's investigation of "in situ" visualization attempts to address problems at the other end of the visualization pipeline. To overcome the severe problems of data logistics involved in managing the rendering of multi-terabyte data sets in networked environments, *in situ* visualization performs the necessary calculations while data still resides on the supercomputer that was used to generate it.

Similar problems of petascale data logistics are central to the mission of the three CETs that focus on large scale data management for distributed environments. As the leaders of the *Center for Enabling Distributed Petascale Science (CEDPS)* make clear in their article (Schopf et al.), such questions of "data placement" are central to the end-to-end effectiveness of SciDAC's highly distributed collaboration environments. The authors describe their development of a policy-driven data placement service, which builds on their experience working with several leading application communities, including HEP, Fusion Energy, Combustion, and Earth Systems. Complementary efforts on automated scientific workflow, using well known Kepler middleware, are also underway at the *Scientific Data Management (SDM) Center.* But in order to help investigators manage and analyze the data deluge they confront, the SDM Center research portfolio extends farther down the storage middleware stack. In the Shoshani et al. article, they describe the ensemble of software tools and middleware that they are developing to help scientists to explore their data through automatic feature extraction and highly scalable indexing of massive data sets, and to optimize their use of storage resources through low level parallel I/0 libraries and in situ processing on the storage nodes ("active storage").

The third CET focused on data management – the *Earth System Grid Center for Enabling Technologies (ESG-CET)* – revolves, as the name suggests, around a single major application community, viz. the climate research community. This community has been at the forefront of the data grid movement for many years, aggressively developing and deploying data grid technology to show how high impact data sharing can be implemented on a global scale, even while the volume of data continues to escalate. Their discussion (Williams et al.) of the past successes, the current implementation, and the future plans for the Earth System Grid describes a model that several other application communities would do well to emulate as we enter the era of petascale data.

Reflecting on the range and diversity of the work on software cyberinfrastructure presented in this issue of *CTWatch Quarterly*, it's hard to avoid the conclusion that the relentless movement toward petascale science, in which the DOE SciDAC program has played such a leading role, has generated a software ecosystem whose continued vitality seems more and more essential to success on the new frontiers of research. But we cannot be complacent. The push beyond petascale is just around the corner and, as before, the effort to scale up even further is certain to bring up uniquely difficult problems that we have not yet anticipated. We must hope, therefore, that the next generation of enabling software technology researchers contains the same kind of energetic, dedicated and creative pioneers that have led the current one.

Failure Tolerance in Petascale Computers

Garth Gibson
Carnegie Mellon University

Bianca Schroeder
Carnegie Mellon University

Joan Digney
Carnegie Mellon University

Introduction

Three of the most difficult and growing problems in future high-performance computing (HPC) installations will be avoiding, coping and recovering from failures. The coming PetaFLOPS clusters will require the simultaneous use and control of hundreds of thousands or even millions of processing, storage, and networking elements. With this large number of elements involved, element failure will be frequent, making it increasingly difficult for applications to make forward progress. The success of petascale computing will depend on the ability to provide reliability and availability at scale.

While researchers and practitioners have spent decades investigating approaches for avoiding, coping and recovering from abstract models of computer failures, the progress in this area has been hindered by the lack of publicly available, detailed failure data from real large-scale systems.

We have collected and analyzed a number of large data sets on failures in high-performance computing (HPC) systems. Using these data sets and large scale trends and assumptions commonly applied to future computing systems design, we project onto the potential machines of the next decade our expectations for failure rates, mean time to application interruption, and the consequential application utilization of the full machine, based on checkpoint/restart fault tolerance and the balanced system design method of matching storage bandwidth and memory size to aggregate computing power.[1]

Not surprisingly, if the growth in aggregate computing power continues to outstrip the growth in per-chip computing power, more and more of the computer's resources may be spent on conventional fault recovery methods. For example, we envision applications being denied as much as half of the system's resources in five years.[2] The alternatives that might compensate for this unacceptable trend include application-level checkpoint compression, new special checkpoint devices or system level process-pairs fault-tolerance for supercomputing applications.

Our interest in large-scale cluster failure stems from our role in a larger effort, the DOE SciDAC-II Petascale Data Storage Institute (PDSI), chartered to anticipate and explore the challenges of storage systems for petascale computing.[3] In as much as checkpoint/restart is a driving application for petascale data storage systems, understanding node failure and application failure tolerance is an important function for the PDSI. To increase the benefit of our data collection efforts, and to inspire others to do the same, we are working with the USENIX Association to make publicly available these and other datasets in a Computer Failure Data Repository (CFDR).[4] Systems researchers and developers need to have ready access to raw data describing how computer failures have occurred on existing large-scale machines.

[1] Grider, G. "HPC I/O and File System Issues and Perspectives," In *Presentation at ISW4, LA-UR-06-0473*, Slides available at http://www.dtc.umn.edu/disc/isw/presentations/isw4 6.pdf, 2006.

[2] Schroeder, B., Gibson, G. "Understanding Failures in Petascale Computers," In SciDAC 2007: *Journal of Physics: Conference Series* 78 (2007) 012022.

[3] Scientific Discovery through Advanced Computing (SciDAC), The Petascale Data Storage Institute (PDSI). http://www.pdsi-scidac.org/, 2006.

[4] The Computer Failure Data Repository (CFDR) - http://cfdr.usenix.org/.

Data Sources

The primary data set we are studying was collected between 1995 and 2005 at Los Alamos National Laboratory (LANL, www.lanl.gov) and covers 22 high-performance computing systems, including a total of 4,750 machines and 24,101 processors.[5] Figure 1 shows pictures of two LANL systems. The data contain an entry for any failure that occurred during the nine year time period that resulted in an application interruption or a node outage. It covers all aspects of system failures: software failures, hardware failures, failures due to operator error, network failures, and failures due to environmental problems (e.g., power outages). For each failure, the data notes start time and end time, the system and node affected, as well as categorized root cause information. To the best of our knowledge, this is the largest failure data set studied to date, both in terms of the time-period it spans and the number of systems and processors it covers. It is also the first to be publicly available to researchers.[6]

Figure 1. Example high-performance computer clusters at Los Alamos National Laboratory, Blue Mountain (above) and ASC Q.

Understanding Outages in LANL Computers

The first question most ask is "What causes a node outage?" Figure 2 provides a root cause breakdown of failures from the LANL data into human, environment, network, software, hardware, and unknown, with the relative frequency of the high-level root cause categories on the left. Hardware is the single largest source of malfunction, with more than 50% of all failures assigned to this category. Software is the second largest contributor, with around 20% of all failures. The trends are similar if we look at Figure 2(b), which shows the fraction of total repair time attributed to each of the different root cause categories.

[5] Schroeder, B., Gibson, G. "A large-scale study of failures in high-performance computing systems," In *Proc. of the 2006 International Conference on Dependable Systems and Networks (DSN'06)*, 2006.

[6] The LANL raw data and more information are available at: http://www.lanl.gov/projects/computerscience/data/.

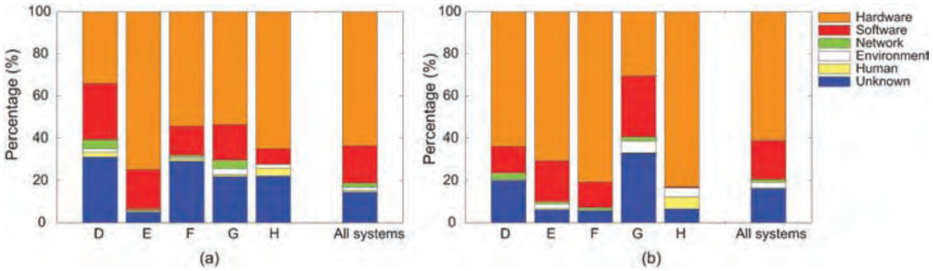


Figure 2. (a) The breakdown of failures by root cause. (b) The breakdown of total repair time spent on a system due to each root cause. Each bar shows the breakdown for the systems of one particular hardware platform, labeled D, E, F, G, and H, and the right-most bar shows aggregate statistics across all LANL systems.

It is important to note that the number of failures with undetermined root cause is significant. Since the fraction of hardware failures is larger than the fraction of undetermined failures, and the fraction of software failures is close to that of undetermined failures, we can still conclude that hardware and software are among the largest contributors to failures. However, we cannot conclude that any of the other failure sources (Human, Environment, Network) is actually insignificant.

A second question is "How frequently do node outages occur?" or "How long can an application be expected to run before it will be interrupted by a node failure?" Figure 3(a) shows the average number of node failures observed per year for each of the LANL systems according to the year that each system was introduced into use. The figure indicates that the failure rates vary widely across systems, from less than 20 failures per year per system to more than 1100 failures per year. Note that a failure rate of 1100 per year means that an application running on all the nodes of the system will be interrupted and forced into recovery more than two times per day. Since many of the applications running on these systems require a large number of nodes and weeks of computation to complete, failure and recovery are frequent events during an application's execution.

One might wonder what causes the large differences in failure rates across the different systems. The main reason for these differences is that the systems vary widely in size. Figure 3(b) shows the average number of failures per year for each system normalized by the number of processors in the system. The normalized failure rates show significantly less variability across the different types of systems, which leads us to two interesting suggestions. *First, the failure rate of a system grows in proportion to the number of processor chips in the system. Second, there is little indication that systems and their hardware get more reliable over time as technology changes.*

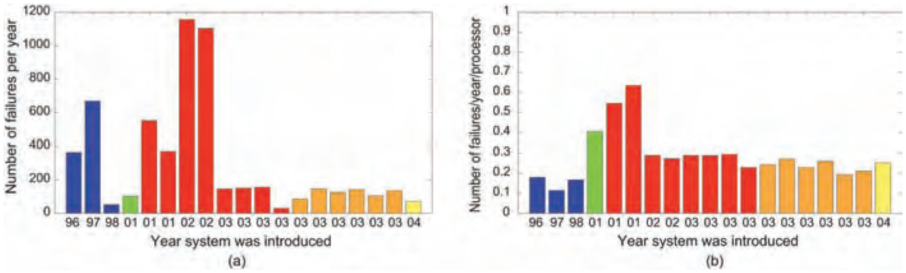


Figure 3. (a) Average number of failures for each LANL system per year. (b) Average number of failures for each system per year normalized by number of processors in the system. Systems with the same hardware type have the same color.

Lower Mean Time To Interrupt (MTTI) in Petascale Computers

What does our data analysis, examined in the light of recent technology trends, predict for the reliability and availability of future HPC systems?

Our essential prediction is that the number of processor chips will grow with time, increasing failure rates and fault tolerance overheads.

First, we expect petascale computers will be conceived and constructed according to long standing trends (aggregate compute performance doubling every year) shown on the top500.org list of the largest documented computers.[7] Second, we expect little or no increase in clock speed, but an increase in the number of processor cores per processor chip, commonly referred to as a socket in the new multi-core processor era, at a fast rate, estimated as doubling every two years.[8] Our data also predicts that failure rates will grow in proportion to the number of sockets in the system and that there is no indication that the failure rate per socket will decrease over time with technology changes. Therefore, as the number of sockets in future systems increases to achieve top500.org performance trends, we expect the system wide failure rate will increase.

In an attempt to quantify what one might expect to see in future systems, we examined the LANL data and found that an optimistic estimate for the failure rate per year per socket is 0.1. Our data does not predict how failure rates will change with increasing numbers of cores per processor chip core, but it is reasonable to predict that many failure prone mechanisms operate at the chip level, so we make the (possibly highly optimistic) assumption that failure rates will increase only with the number of chip sockets, and not with the number of cores per chip.

As a baseline for our projections, we modeled the Jaguar system at Oak Ridge National Laboratory (ORNL). After it is expanded to a Petaflop system in 2008, Jaguar is expected to have around 11,000 processor sockets (dual-core Opterons), 45 TB of main memory and a storage bandwidth of 55 GB/s.[9] Predictions for system expansion are bracketed with three projected rates of growth, with numbers of cores doubling every 18, 24 and 30 months.

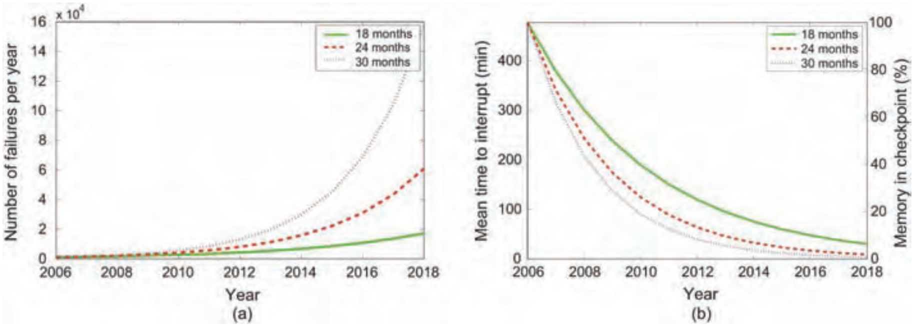


Figure 4. (a) The expected growth in failure rate and (b) decrease in MTTI, assuming that the number of cores per socket grows by a factor of two every 18, 24 and 30 months, respectively, and the number of sockets increases so that aggregate performance conforms to top500.org.

Figure 4 plots the expected increase in failure rate and corresponding decrease in mean time to interrupt (MTTI), based on the above assumptions. Even if we assume a zero increase in failure rate with more cores per socket (a stretch), the failure rates across the biggest machines in the top 500 lists of the future can be expected to grow dramatically.

[7] Top 500 supercomputing sites - http://www.top500.org/, 2007.

[8] Asanovic, K., Bodik, R., Catanzaro, B. C., Gebis, J. J., Husbands, P., Keutzer, K., Patterson, D. A., Plishker, W. L., Shalf, J., Williams, S. W., Yelick, K. A. "The landscape of parallel computing research: A view from Berkeley," *Technical Report UCB/EECS-2006-183*, EECS Department, University of California, Berkeley, Dec 2006.

[9] Roth, P. C. "The Path to Petascale at Oak Ridge National Laboratory," In *Petascale Data Storage Workshop Supercomputing '06*, 2006.

Decreasing Effectiveness of Checkpoint-Restart Fault Tolerance

Observing this sort of dramatic increase in failure rates brings up the question of how the utility of future systems will be affected. Fault tolerance in HPC systems is typically implemented with checkpoint restart programming. Here, the application periodically stops useful work to write a checkpoint to disk. In case of a node failure, the application is restarted from the most recent checkpoint and recomputes the lost results.

The time to write a checkpoint depends on the total amount of memory in the system, the fraction of memory the application needs to checkpoint to be able to recover, and the I/O bandwidth. To be conservative, we assume that demanding applications may utilize and checkpoint their entire memory. For a system like Jaguar, with 45TB of memory and 55 GB/s of storage bandwidth, that means one system-wide checkpoint will take on the order of 13 minutes. In a balanced system model, where bandwidth and memory both grow in proportion to compute power, the time to write a checkpoint will stay constant over time. However, with failures becoming more frequent, restarting will be more frequent and application work will be recomputed more frequently. Reducing the time between checkpoints reduces the amount of work recomputed on a restart but it also increases the fraction of each checkpoint interval spent taking a checkpoint.

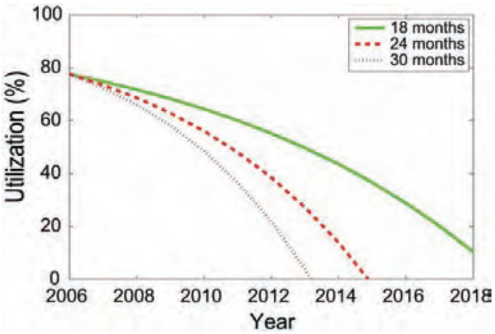


Figure 5. Effective application utilization drops because mean time to interrupt is dropping and more time will be lost to taking checkpoints and restarting from checkpoints. The three models are the same as in Figure 4.

Based on the models of Figure 4 and on an optimal selection of the period between checkpoints,[10] Figure 5 shows a prediction that the effective resource utilization by an application will drastically decrease over time. For example, in the case where the number of cores per chip doubles every 30 months, the utilization drops to zero by 2013, meaning the system is spending 100% of its time writing checkpoints or recovering lost work, a situation that is clearly unacceptable. In the next section we consider possible ways to stave off this projected drop in resources utilization.

Better Fault Tolerance for Petascale Computers

As LANL's data suggests that failure rate grows proportionally to the number of sockets, keeping the number of sockets constant should stave off an increase in the failure rate. To do this, however, means either failing to achieve the top500.org aggregate performance trends, or increasing the performance of each processor chip faster than currently projected.[8] Chip designers consider it unlikely that we will see a return to the era of rapid decreases in processor cycle time because of the power consumption implications. The remaining alternative, increasing the number of cores per chip faster, would probably not be effective, even if it was possible, because memory bandwidth

[10] Young, J. W. "A first order approximation to the optimum checkpoint interval," *Commun. ACM*, 17(9):530–531, 1974.

per chip will not keep up. Therefore, we think the number of processor chip sockets will continue to increase to keep performance on the top500.org trends.

Socket Reliability: The increase in failure rates could also be prevented if individual processor chip sockets were made more reliable each year in the future, i.e., if the per socket MTTI would increase proportionally to the number of sockets per system over time. Unfortunately, LANL's data does not indicate that hardware has become more reliable over time, suggesting that as long as the number of sockets is rising, the system-wide MTTI will drop.

Partitioning: The number of interrupts an application sees depends on the number of processor chips it is using in parallel. One way to stave off the drop in MTTI per application would be to run it only on a constant-sized sub-partition of the machine, rather than on all nodes of a machine. Unfortunately, while this solution works for small applications that do not need performance to increase faster than the speed each chip increases, it is not appealing for the most demanding "hero" applications, for which the largest new computers are often justified.

Faster Checkpointing: The basic premise of the checkpoint restart approach to fault tolerance, often called the balanced system design, is that even if MTTI is not decreasing, storage bandwidth increases in proportion to total performance.[1] Though achieving balance (i.e., doubling of storage bandwidth every year) is a difficult challenge for storage systems, one way to cope with increasing failure rates is to effect further increases in storage bandwidth. For example, assuming that the number of sockets and hence the failure rate grows by 40% per year, the effective application utilization would stay the same if checkpoints were taken in 30% less time each year. Projections show that this sort of increase in bandwidth is orders of magnitude higher than the commonly expected increase in bandwidth per disk drive (generally about 20% per year). Therefore, an increase in bandwidth would have to come from a rapid growth in the total number of drives, well over 100% per year, increasing the cost of the storage system much faster than any other part of petascale computers. This might be possible, but it is not very desirable.

Another option is to decrease the amount of memory being checkpointed, either by not growing total memory as fast, or by better compression of application data leading to only a smaller fraction of memory being written in each checkpoint. Growing total memory at a slower than balanced rate will help reduce total system cost, which is perhaps independently likely, but may not be acceptable for the most demanding applications. Of the two, compression seems to be the more appealing approach, and is entirely under the control of application programmers.

Achieving higher checkpoint speedups purely by compression will require significantly better compression ratios each year. As early as in the year 2010, an application will have to construct its checkpoints with a size at most 50% of the total memory. Once the 50% mark is crossed, other options, such as diskless checkpointing where the checkpoint is written to the volatile memory of another node rather than disk, [11] [12] or hybrid approaches[13] become viable. We recommend that any application capable of compressing its checkpoint size should pursue this path; considering the increasing number of cycles that will go into checkpointing, the compute time needed for compression may be time well spent.

A third approach to taking checkpoints faster is to introduce special devices between storage and memory that will accept a checkpoint at speeds that scale with memory, then relay the checkpoint to storage after the application has resumed computing. Although such an intermediate memory could be very expensive as it is as large as

[11] Plank, J. S., Li, K. "Faster checkpointing with N + 1 parity," In *Proc. 24th International Symposium on Fault Tolerant Computing*, 1994.

[12] Plank, J. S., Li, K., Puening, M. A. "Diskless checkpointing," *IEEE Trans. Parallel Distrib. Syst.*, 9(10):972–986, 1998.

[13] Vaidya, N. H. "A case for two-level distributed recovery schemes," In *Proceedings of the 1995 ACM SIGMETRICS conference*, 1995.

memory, it might be a good application for cheaper but write-limited technologies such as flash memory, because checkpoints are written infrequently.

Non-Checkpoint-based Fault Tolerance: Process-pairs duplication and checking of all computations is a traditional method for tolerating detectable faults that hasn't been applied to HPC systems.[14] [15] [16] Basically, every operation is done twice in different nodes so the later failure of a node does not destroy the operation's results. Process pairs would eliminate both the cost associated with writing checkpoints, because they are not needed, as well as lost work in the case of failure. However, using process pairs is expensive in that it requires giving up 50% of the hardware to compute each operation twice in different nodes and it introduces further overheads to keep process pairs in synch. However, if no other method works to keep utilization above 50%, this sacrifice might become appropriate, and it bounds the decrease in effectiveness to about 50%, perhaps without requiring special hardware.

Conclusions

The most demanding applications, often the same applications that justify the largest computers, will see ever-increasing failure rates if the trends seen at top500.org continue. Using the standard checkpoint restart fault tolerance strategy, the efficacy of petascale machines running demanding applications will fall off. Relying on computer vendors to counter this trend is not recommended by historical data, and relying on disk storage bandwidth to counter it is likely to be expensive at best. We recommend that these applications consider spending an increasing number of cycles compressing checkpoints. We also recommend experimentation with process pairs fault tolerance for supercomputing. And if technologies such as flash memory are appropriate, we recommend experimenting with special devices devoted to checkpointing.

The Computer Failure Data Repository

The work described in this article is part of our broader research agenda with the goal of analyzing and making publicly available the failure data from a large variety of real production systems. To date, large-scale studies of failures in real production systems are scarce, probably a result of the reluctance of the owners of such systems to release failure data. Thus, we have built a public *Computer Failure Data Repository* (CFDR), hosted by the USENIX association[4] with the goal of accelerating research on system reliability by filling the nearly empty collection of public data with detailed failure data from a variety of large production systems. We encourage all petascale computing organizations to collect and publish failure data for their systems in the repository.

Acknowledgments

We would like to thank Jamez Nunez and Gary Grider from the High Performance Computing Division at Los Alamos National Lab for collecting and providing us with data and helping us to interpret the data. We thank the members and companies of the PDL Consortium (including APC, Cisco, Google, EMC, Hewlett-Packard, Hitachi, IBM, Intel, LSI, Microsoft, Network Appliance, Oracle, Panasas, Seagate, and Symantec) for their interest and support. This material is based upon work supported by the Department of Energy under Award Number DE-FC02-06ER25767 [3] and on research sponsored in part by the Army Research Office, under agreement number DAAD19–02–1–0389.

[14] Bressoud, T. C., Schneider, F. B., "Hypervisor-based fault tolerance," *ACM Trans. Comput. Syst.*, 14(1):80–107, 1996.

[15] Chapin, J., Rosenblum, M., Devine, S., Lahiri, T., Teodosiu, D., Gupta, A. "Hive: fault containment for shared-memory multiprocessors," In *SOSP '95: Proceedings of the fifteenth ACM symposium on Operating systems principles*, 1995.

[16] McEvoy, D. "The architecture of tandem's nonstop system," In *ACM 81: Proceedings of the ACM '81 conference*, page 245, New York, NY, USA, 1981. ACM Press.

Enabling Advanced Scientific Computing Software

Steven Parker
University of Utah

Rob Armstrong
Sandia National Laboratory

David Bernholdt
Oakridge National Laboratory

Tamara Dahlgren
Lawrence Livermore National Laboratory

Tom Epperly
Lawrence Livermore National Laboratory

Joseph Kenny
Sandia National Laboratory

Manoj Krishnan
Pacific Northwest National Laboratory

Gary Kumfert
Lawrence Livermore National Laboratory

Jay Larson
Argonne National Laboratory

Lois Curfman McInnes
Argonne National Laboratory

Jarek Nieplocha
Pacific Northwest National Laboratory

Jaideep Ray
Sandia National Laboratory

Sveta Shasharina
Tech-X Corporation

Overview

The SciDAC Center for Technology for Advanced Scientific Computing Software (TASCS) focuses on developing tools, components and best practices for developing high quality, reusable high-performance computing software. TASCS fosters the Common Component Architecture (CCA) through a community forum that involves a wide range of participants. The CCA environment aims to bring component-based software development techniques and tools, which are commonplace in the computing industry, to high performance computing. To do so, several challenges are being addressed including parallelism, performance, and efficient handling of large datasets. The CCA has produced a specification that allows components to be deployed and reused in a highly extensible yet efficient parallel environment. The primary advantage of this component-based approach is the separate development of simulation algorithms, models, and infrastructure. This allows the pieces of a complex simulation to evolve independently, thereby helping a system grow intelligently as technologies mature. The CCA tools have been used to improve productivity and increase capabilities for HPC software in meshing, solvers, and computational chemistry, among other applications.

TASCS supports a range of core technologies for using components in high-performance simulation software, including the Caffeine framework, the Babel interoperability tool, and the Bocca development environment for HPC components. In addition, the CCA helps provide access to tools for performance analysis, for coupling parallel simulations, for mixing distributed and parallel computing, and for ensuring software quality in complex parallel simulations. These tools can help tame the complexity of utilizing parallel computation, especially for sophisticated applications that integrate multiple software packages, physical simulation regimes or solution techniques. We will discuss some of these tools and show how they have been used to solve HPC programming challenges.

Component-based Software Engineering

The component-based software engineering (CBSE)[1] methodology has been developed to facilitate the understanding, development, and evolution of large-scale software systems. By emphasizing strong encapsulation of code with well defined interfaces between modules, a component approach provides a way of decomposing software into units that are conceptually manageable, and that interact in specific and easily understood ways.

These characteristics also facilitate the design and evolution of large, complex software systems by distinguishing between the functional specification of a component (fixed or slowly changing) and its implementation (possibly more rapidly changing, or even having multiple implementations). With thoughtful design of interfaces, component approaches can promote software reuse and interoperability. The encapsulation of components makes them useful in collaborative software development situations, where individuals or small groups take responsibility for the implementation of components conforming to interface specifications agreed to by the collaboration as a whole. These characteristics match very well with the way that the modern computational science

[1] Szyperski, C. *Component Software: Beyond Object-Oriented Programming*, ACM Press, New York, 1999.

community approaches simulation software, which makes the component approach an ideal match for high-performance scientific computing. Furthermore, simulation coupling is of rapidly growing importance in scientific computing, and maps directly to the philosophy of software components.

Although the idea of CBSE has a long history, component architectures have only recently become practical for use in high-performance scientific computing. Development of the Common Component Architecture,[2] a general component environment, began with the establishment of the CCA Forum[3] in 1998. The Cactus framework,[4] originally developed primarily to support numerical relativity simulations, began appearing in the scientific literature in roughly 1999. Another domain-specific framework effort, the Earth System Modeling Framework (ESMF),[5] [6] began in 2001.

Scientific computing poses both technical and sociological challenges to the deployment and adoption of new technologies, such as components and frameworks. The scientific CBSE community is still in the formative stages of understanding how these concepts can be used most effectively in the context of advanced computational science applications. At the same time, the field is evolving: computing power grows, the tools and software environments evolve, and applications move to take advantage of new capabilities not just to solve larger problems faster, but also at higher levels of physical fidelity. If CBSE is to become a routine part of computational science, we need to anticipate emerging trends and how they will impact the concepts and tools of CBSE. These emerging trends in high-end scientific computing pose both challenges and opportunities for component-based software development, and provide incredible opportunities to dramatically enhance the reliability, maintainability, and scope of HPC applications.

The CCA Component Model

Formally, the Common Component Architecture is a specification of an HPC-friendly component model. This specification provides a focus for an extensive research and development effort. The research effort emphasizes understanding how best to utilize and implement component-based software engineering practices in the high-performance scientific computing arena. The development effort creates practical reference implementations and helps scientific software developers use them to create CCA-compliant components and applications.

The CCA specification is expressed as a set of abstract interfaces[7] written in the Scientific Interface Definition Language (SIDL). SIDL is used by the Babel language interoperability tool (discussed further below), which implicitly defines bindings to the various languages that Babel supports (currently Fortran 77, Fortran 90, C, C++, Python, and Java).

The primary players in a CCA application are *Components* that encapsulate a particular piece of software, *Ports* that define the interfaces between components, and *Framework* that glue the aforementioned components together and allow them to communicate through the ports that are defined. Figure 1 illustrates how several such components combine together to form a single application. This conceptual model should be familiar to anyone that has used component-based systems before, except that the components explicitly support parallelism and the ports facilitate fine-grained communication of large quantities of data without the copying that is inherent in many such systems.

[2] Allan, B. A., Armstrong, R., Bernholdt, D. E., Bertrand, F., Chiu, K., Dahlgren, T. L., Damevski, K., Elwasif, W. R., Epperly, T. G. W., Govindaraju, M., Katz, D. S., Kohl, J. A., Krishnan, M., Kumfert, G., Larson, J. W., Lefantzi, S., Lewis, M. J., Malony, A. D., McInnes, L. C., Nieplocha, J., Norris, B., Parker, S. G., Ray, J., Shende, S., Windus, T. L., Zhou, S. "A Component Architecture for High-Performance Scientific Computing," *Intl. J. High-Perf. Computing Appl.*, 2006, pp. 163-202.

[3] CCA Forum - http://cca-forum.org/

[4] Cactus - http://www.cactuscode.org/

[5] Earth System Modeling Framework - http://www.esmf.ucar.edu/

[6] Collins, N., Theurich, G., DeLuca, C., Suarez, M., Trayanov, A., Balaji, V., Li, P., Yang, W., Hill, C., da Silva, A. "Design and Implementation of Components in the Earth System Modeling Framework,". *Intl. J. High-Perf. Computing Appl.*, 2005, pp. 341-350.

[7] CCA Specification - https://www.cca-forum.org/wiki/tiki-index.php?page=CCA+Specification

The core of the CCA specification is the *Services* interface. This is the primary means by which components interact with the framework, allowing the component to inform the framework of component capabilities and interfaces, and to request access to other services the framework may provide, such as information about connections between itself and other components, or the ability to instantiate and otherwise manipulate other components. The Services interface allows a component to declare two different types of ports, those that it will *provide* and those that it will *use*. These ports can also be thought of as callee and caller. These ports make it possible for a CCA framework to effectively mediate connections between components, and allows components to be assembled by another entity (a user through a GUI, a script, or even another component). The CCA component model espouses a minimalist approach, requiring only that components implement a single method/function (called setServices) that establishes contact between the component and the framework.

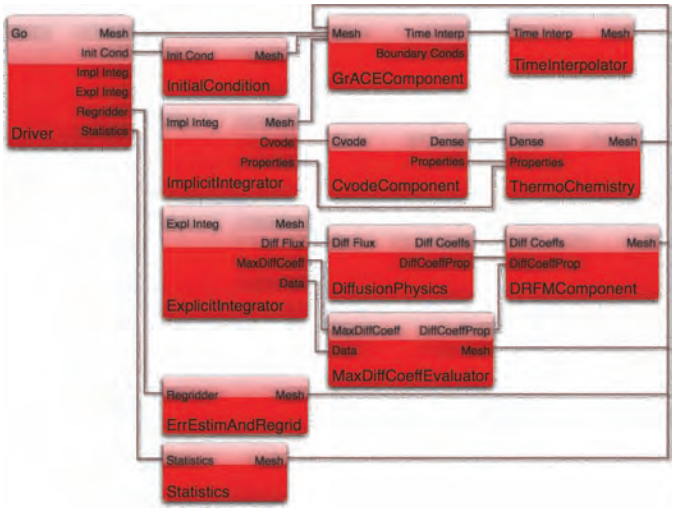


Figure 1. A CCA Component wiring diagram showing the interconnection of components in a reaction-diffusion combustion simulation.

CCA ports are simply a babel-described interface, and are also identified by a type and name. An optional set of properties associated with each port can describe additional functionality, such as the minimum/maximum number of connections. Components may provide multiple ports and even multiple instances of the same port. In addition to defining the port mechanism, the CCA specification also defines a number of specific ports that are useful in multiple applications, such as the GoPort (for starting an application), parameter ports for communicating basic configuration information to the application, and ports for communicating events to other components. The CCA reuses this port mechanism to export services provided by the framework that allow a component to assemble and manage other components, monitor available components, and watch for application events. Using this mechanism, graphical user interfaces become simply a component that is instantiated in the system and are not tied to the underlying framework. These services also allow dynamic behavior of the application itself, such as swapping components, and provide a mechanism for a hierarchy of components that are assembled at multiple levels of abstraction.

Software Tools

Beyond the core specification, a number of software tools have been developed that assist users in developing HPC applications around component technology. These tools, developed both through the TASCS Center, and through CCA collaborators, provide interoperability between programming languages, assistance with packing and deployment, and tools for performance analysis. There also exists a handful of different CCA-compliant frameworks that target different operating environments. A few of these tools are described here and additional information can be found at the CCA Forum home page.

Babel (pronounced babble)[8] addresses the language interoperability problem using a Scientific Interface Definition Language (SIDL) that provides the ability to interact between programming languages and platforms, while addressing the unique needs of parallel scientific computing. Given a SIDL description that describes the calling interface (but not the implementation) of a particular software library, Babel generates glue code that allows software implemented in one supported language to be called from any other supported language. SIDL supports complex numbers and dynamic multi-dimensional arrays as well as parallel communication directives that are required for parallel distributed components. SIDL also provides other common features that are generally useful for software engineering, such as enumerated types, symbol versioning, and name space management, and employs an object-oriented inheritance model similar to Java. Babel provides a code splicing capability that preserves old edits during the regeneration of implementation files after modifications to the SIDL source.

Babel recently added a remote method invocation that provides a consistent mechanism to communicate between objects regardless of where they are located. This model provides a simpler and more consistent object-oriented programming model than CORBA or COM, and provides an API for third-party plug-ins to customize the underlying communication model. A simple TCP/IP protocol is provided that outperforms both CORBA and Web Services. Babel RMI fills a niche in "short-haul" distributed computing - within a machine room, or even in a single machine with concurrent MPI runs.

Ccaffeine[9] is the main CCA framework implementation for HPC parallel computing and it supports the component analogs of both the single program/multiple data (SPMD) and multiple program/multiple data (MPMD) parallel programming models. We refer to these as single or multiple *component*/multiple data (SCMD or MCMD) models. Figure 2 depicts the SCMD case; each process is loaded with the same set of components wired together in the same way. Interactions among components within a given process (vertical direction) take place through the normal CCA means - through Ports. Interactions within a parallel component (called a parallel cohort) take place via the parallel programming model that the component uses (typically MPI). "Diagonal" interactions - between component A on one process and component B on another process - are not prohibited by the CCA, but are currently not supported in Ccaffeine.

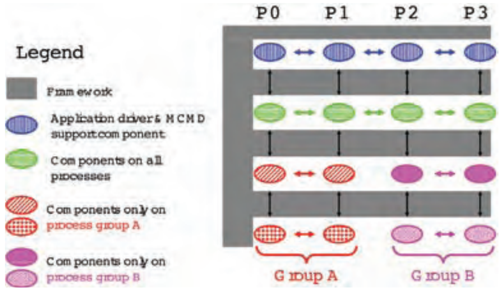

P0
P1
P2
P3
Legend
Framework
Application driver & MCMD support component
Components on all processes
Components only on process group A
Components only on process group B
Group A
Group B

Figure 2. A schematic representation of the CCA parallel programming environment in the single component/multiple data (SCMD) paradigm.

[8] Babel - http://www.llnl.gov/CASC/components/babel.html

[9] CCaffeine - http://www.cca-forum.org/ccafe/

Bocca is a system for creating, managing, and deploying CCA-based components. Bocca can create components, define ports and interfaces, and manage the build system for the resulting component. Bocca is a new addition to the CCA tool suite, but promises to dramatically simplify the process of creating a component from scratch and subsequently maintaining it.

Performance Monitoring and Tuning. TAU is a robust and portable measurement interface and system for software performance evaluation. Using SIDL to describe TAU's measurement API, Babel has enabled access to TAU across all supported languages. CCA/Babel has also enabled incorporation of dynamic selection of measurement options into the TAU performance evaluation tools. Users can choose from a variety of measurement options interactively at runtime, without re-compilation of applications. Proxy components are automatically generated to mirror a component's interface, allowing dynamic interposition of proxies between callers and callees, via hooks into the intermediate Babel communication layer. Such inter-component interaction measurements can correlate performance to application parameters, used for constructing more sophisticated performance models.

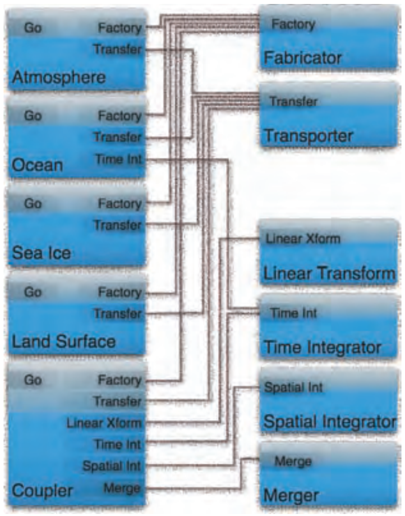


Figure 3. A Component wiring diagram showing how explicit coupling components can manage the interfaces in a multi-physics climate simulation.

Components for Parallel Coupling. Multiphysics and multiscale models face a formidable obstacle: the parallel coupling problem. Parallel coupling involves the description, transfer, and transformation of distributed data. We are developing a set of CCA components (the Parallel Coupling Infrastructure, or PCI Toolkit) that leverage successful parallel coupling technology - the Model Coupling Toolkit - to simplify the process of remapping data between disparate discretizations and processor mappings.

Additional tools. In addition to these tools and software frameworks, the TASCS Center maintains additional component-based software for developing HPC applications on CCA technology. A graphical user interface allows interactive construction and monitoring of HPC applications. CCA-lite is a slimmed down version of the CCA specification designed for statically-linked components that are written in C, C++ and/or Fortran. Additional frameworks, such as the SCIJump distributed/parallel framework from Utah and the LegionCCA Grid-based framework from SUNY Binghamton, also provide alternative deployment vehicles for CCA components.

Applications

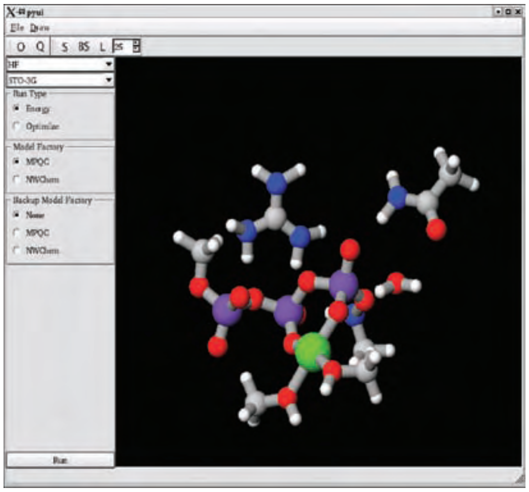
CCA is applicable to a broad range of parallel applications. We highlight a few of these endeavors.

Combustion Modeling. One of the most sophisticated implementations of the CCA paradigm to date is in combustion modeling. The endeavor, which started in 2001 at the Computational Facility for Reacting Flow Science (CFRFS) project,[10] seeks to create a facility for the high fidelity simulation of flames involving realistic physical models, nonlinear PDEs, and a spectrum of time and length scales. Given the complexity of the problem and the multiplicity of physical and mathematical models required for the task, a component based approach was a natural fit. CCA was chosen primarily for its high performance and simplicity. General purpose components, implementing a particular numerical or physical functionality, are reused in various code assemblies.

[10] Najm, H. (PI), Computational Facility for Reacting Flow Science (CFRFS) - http://cfrfs.ca.sandia.gov/.

Fusion. The standardization of fusion codes has become of paramount importance with the beginning of the fusion integrated modeling. In 2006, two SciDAC projects, SWIM (Center for Simulation of RF Wave Interactions with Magnetohydrodynamics)[11] and CPES (the Center for Plasma Edge Simulation),[12] were funded. In 2007, another integrating SciDAC project FACETS (the Framework Application for Core-Edge Transport Simulations)[13] started. Each of these three projects addresses a different area of integration, and some of them will possibly unite in the upcoming Fusion Simulation Project sometime in 2008. All projects are looking at components technologies to assist them in defining and composing participating codes. FACETS uses the CCA language interoperability tool, Babel, to wrap F90 modules for the use in its C++ framework.

Quantum Chemistry. In response to the strong need for a community software base in the quantum chemistry community, members of the Quantum Chemistry Scientific Application Partnership (QCSAP) have leveraged the software engineering practices and tools developed by the CCA Forum to create a community-based architecture for chemistry simulation. By designing flexible interfaces by which quantum chemistry capabilities can be shared, the scaling of human effort is drastically improved, allowing the exploration of new algorithms and hardware in a fraction of the time required by traditional approaches. A screenshot of such an application is shown in Figure 4. For the quantum chemistry packages adopting this new design paradigm [GAMESS (Ames Laboratory), NWChem (Pacific Northwest National Laboratory) and MPQC (Sandia National Laboratories) thus far], many advantages have already been realized, including the abilities to leverage more efficient optimization components written by domain experts, transparently share low-level integral evaluation routines, more efficiently utilize high performance computers, and automatically tune applications for specific molecular systems and hardware environments.



HF

STO-3G

Run Type
● Energy
○ Optimize

Model Factory
● MPQC
○ NWChem

Backup Model Factory
● None
○ MPQC
○ NWChem

Figure 4. Screenshot of a graphical component front-end developed within the QCSAP.

The Future

Some of the ongoing work includes advances in component capabilities for massively parallel and heterogeneous architectures, runtime enforcement of behavioral semantics, additional expressibility for complex interactions between components, and parallel coupling. We provide brief highlights below; see [3, 14] for additional details.

Emerging HPC Environments. Scientists developing petascale computational science capabilities continue to face major challenges in effectively using emerging high-performance computing (HPC) architectures, which are characterized by large processor counts and increasing use of heterogeneous, specialized environments. We are thus developing new tools for CCA users to simplify and accelerate the development of true petascale applications on diverse hardware platforms. Our goals are that CCA users will be able to flexibly and dynamically express higher levels of parallelism,[15] transparently exploit specialized coprocessing resources, and support intelligent application-level responses to the hardware failures that are inevitable on systems of this scale. For example, we are working with a bioinformatics/proteomics application team to analyze data generated by mass spectrometers at PNNL.[16]

Software Quality and Verification. To help make the vision of interchangeable components a reality for scientific software, we are developing capabilities for the composition- and execution-time verification of interface semantics.[17] [18] Component

[11] SWIM - http://cswim.org/

[12] CPES - http://www.cims.nyu.edu/cpes/

[13] FACETS - https://www.facetsproject.org/facets

[14] McInnes, L., Dahlgren, T., Nieplocha, J., Bernholdt, D., Allan, B., Armstrong, R., Chavarria, D., Elwasif, W., Gorton, I., Kenny, J., Krishan, M., Malony, A., Norris, B., Ray, J., Shende, S. "Research Initiatives for Plug-and-Play Scientific Computing," *Journal of Physics*: Conference Series 78 (2007), (available via http://www.iop.org/EJ/abstract/1742-6596/78/1/012046).

[15] Krishnan, M., Alexeev, Y., Windus, T., Nieplocha, J. "Multilevel Parallelism in Computational Chemistry using Common Component Architecture and Global Arrays," *Proceedings of SuperComputing*, 2005.

[16] High-Performance Mass Spectrometry Facility, Pacific Northwest National Laboratory - http://www.emsl.pnl.gov/capabs/hpmsf.shtml

[17] Dahlgren, T., Devanbu, P. "Improving Scientific Software Component Quality Through Assertions," *Proc. 2nd Int. Workshop on Software Engineering for High Performance Computing System Applications*, 2005, pp. 73-77, (available via http://csdl.ics.hawaii.edu/se-hpcs/papers/sehpcs-proceedings.pdf).

[18] Dahlgren, T. "Performance-Driven Interface Contract Enforcement for Scientific Components," *Proc. 10th Int. Symp. on Component-Based Software Engineering*, LNCS 4608, Springer-Verlag, 2007, pp. 157-172.

interfaces, expressed separately from implementations, can be extended with semantic information to provide concise specifications that are both human-readable and interpreted by software. Unlike traditional verification techniques based either on post-execution comparisons with prior or analytical results or on algorithm-based fault tolerance techniques, this approach enables error detection closer to the point of failure. The result is improved testing, debugging, and runtime monitoring of software quality, thereby providing software developers with a powerful tool for catching errors early and ensuring correct software usage.

Computational Quality of Service. As computational science progresses toward ever more realistic multi-physics applications, no single research group can effectively select or tune all components of a given application, and no solution strategy can seamlessly span the entire spectrum efficiently. Common component interfaces enable easy access to suites of independently developed algorithms and implementations. The challenge then becomes how, during runtime, to make the best choices for reliability, accuracy, and performance. As motivated by simulations in combustion,[10] quantum chemistry,[19] fusion,[13] and accelerators,[20] TASCS researchers are addressing this challenge by developing tools for Computational Quality of Service (CQoS), or the automatic selection and configuration of components to suit a particular computational purpose and environment.[21] The two main facets of CQoS tools are (1) measurement and analysis infrastructure and (2) control infrastructure for dynamic component replacement and domain-specific decision making.[22]

Parallel Data Redistribution and Parallel Remote Method Invocation. Parallel components raise questions about the semantics of method invocations and the mechanics of parallel data redistribution involving these components. Method invocations between parallel components are an opportunity to automate the data redistribution and translation semantics of the interaction between those components. The so-called MxN problem (where M processors associated with one component coordinate with N processors associated with another) arises often when multiple simulation components are joined in a single application. This allows an application to utilize a combination of task-based parallelism and domain decomposition to achieve integration, regardless of the scaling characteristics and resource constraints of the individual components. Support for this capability is being integrated into the Babel compiler.

Conclusion

The Common Component Architecture is a solid foundation for developing modular, maintainable high-performance simulations. Through support of the TASCS Center and collaborators, the surrounding ecosystem continues to flourish, and provides new functionality for taming the complexity of multi-physics, multi-scale, scalable applications.

Acknowledgment

This work was supported by the U.S. Department of Energy's Scientific Discovery through Advanced Computing (SciDAC) program, through the Office of Advanced Scientific Computing Research, Office of Science. The CCA Forum is a community involving participants for numerous DOE national laboratories, Universities, Companies, and other organizations.

[19] Gordon, M. (PI), Chemistry Framework using the CCA - http://www.scidac.gov/matchem/better.html

[20] Spentzouris, P. (PI), SciDAC Community Petascale Project for Accelerator Science and Simulation (COMPASS) - https://compass.fnal.gov

[21] Norris, B., Ray, J., Armstrong, R., McInnes, L.,Bernholdt, D., Elwasif, W., Malony, A., Shende, S. "Computational Quality of Service for Scientific Components," *Proc. Int. Symp. on Component-Based Software Engineering*, 2004, Edinburgh, Scotland (available via ftp://info.mcs.anl.gov/pub/tech_reports/reports/P1131.pdf).

[22] McInnes, L., Ray, J., Armstrong, R., Dahlgren, T., Malony, A., Norris, B., Shende, S., Kenny, J., and Steensland, J. "Computational Quality of Service for Scientific CCA Applications: Composition, Substitution, and Reconfiguration," Argonne National Laboratory preprint ANL/MCS-P1326-0206, 2006, (available via ftp://info.mcs.anl.gov/pub/tech_reports/reports/P1326.pdf).

Performance Engineering: Understanding and Improving the Performance of Large-Scale Codes

David H. Bailey
Lawrence Berkeley National Laboratory

Robert Lucas
University of Southern California

Paul Hovland
Boyana Norris
Argonne National Laboratory

Kathy Yelick
Dan Gunter
Lawrence Berkeley National Laboratory

Bronis de Supinski
Dan Quinlan
Lawrence Livermore National Laboratory

Pat Worley
Jeff Vetter
Phil Roth
Oak Ridge National Laboratory

John Mellor-Crummey
Rice University

Allan Snavely
University of California, San Diego

Jeff Hollingsworth
University of Maryland

Dan Reed
Rob Fowler
Ying Zhang
University of North Carolina

Mary Hall
Jacque Chame
University of Southern California

Jack Dongarra
Shirley Moore
University of Tennessee, Knoxville

1. Introduction

Understanding and enhancing the performance of large-scale scientific programs is a crucial component of the high-performance computing world. This is due not only to the increasing processor count, architectural complexity and application complexity that we face, but also due to the sheer cost of these systems. A quick calculation shows that if one can increase by just 30% the performance of two of the major SciDAC[1] applications codes (which together use, say, 10% of the NERSC and ORNL high-end systems over three years), this represents a savings of some $6 million.

Within just five years, systems with one million processors are expected, which poses a challenge not only to application developers but also to those engaged in performance tuning. Earlier research and development by us and others in the performance research area focused on the memory wall – the rising disparity between processor speed and memory latency. Now the emerging multi-core commodity microprocessor designs, with many processors on a single chip and large shared caches, create even greater penalties for off-chip memory accesses and further increase optimization complexity. With the release of systems such as the Cray X1, custom vector processing systems have re-emerged in U.S. markets. Other emerging designs include single-instruction multiple-data (SIMD) extensions, field-programmable gate arrays (FPGAs), graphics processors and the Sony-Toshiba-IBM Cell processor. Understanding the performance implications for such diverse architectures is a daunting task.

In concert with the growing scale and complexity of systems is the growing scale and complexity of the scientific applications themselves. Applications are increasingly multilingual, with source code and libraries created using a blend of Fortran 77, Fortran-90, C, C++, Java, and even interpreted languages such as Python. Large applications typically have rather complex build processes, involving code preprocessors, macros and make files. Effective performance analysis methodologies must deal seamlessly with such structures. Applications can be large, often exceeding one million lines of code. Optimizations may be required at many locations in the code, and seeming local changes can affect global data structures. Applications are often componentized and performance can depend significantly on the context in which the components are used. Finally, applications increasingly involve advanced features such as adaptive mesh refinement, data intensive operations and multi-scale, multi-physics and multi-method computations.

The PERI project emphasizes three aspects of performance tuning for high-end systems and the complex SciDAC applications that run on them: (1) performance modeling of applications and systems; (2) automatic performance tuning; and (3) application engagement and tuning. The next section discusses the modeling activities we are undertaking both to understand the performance of applications better and to be able to determine what are reasonable bounds on expected performance. Section 3 presents the PERI vision for how we are creating an automatic performance tuning capability, which ideally will alleviate scientific programmers of this burden. Automating performance tuning is a long-term research project, and the SciDAC program has scientific objectives that cannot await its outcome. Thus, as Section 4 discusses, we are engaging with DOE computational scientists to address today's most

[1] DOE SciDAC Program – http://www.scidac.gov/

pressing performance problems. Finally, Section 5 summarizes the current state of the PERI SciDAC-2 project.

2. Performance Modeling and Prediction

The goal of performance modeling is to understand the performance of an application on a computer system via measurement and analysis. This information can be used for a variety of tasks: evaluating architectural tradeoffs early in the system design cycle, validating performance of a new system installation, guiding algorithm choice when developing a new application, improving optimization of applications on specific platforms, and guiding the application of techniques for automated tuning and optimization.[2] Modeling is now an integral part of many high-end system procurements,[3] thus making performance research useful beyond the confines of performance tuning. For performance engineering, modeling analyses (when coupled with empirical data) can inform us when tuning is needed, and just as importantly, when we are done. Naturally, if they are to support automatic performance tuning, then the models themselves must be automatically generated.

Traditional performance modeling and prediction have been done via some combination of three methods: (1) analytical modeling; (2) statistical modeling derived from measurement; and (3) simulation. In the earlier SciDAC-1 Performance Evaluation Research Center (PERC), researchers developed a semi-automatic yet accurate methodology based on application signatures, machine profiles and convolutions. These methodologies allow us to predict performance to within reasonable tolerances for an important set of applications on traditional clusters of SMPs for specific inputs and processor counts.

PERI is extending these techniques not only to account for the effects of emerging architectures, but also to model scaling of input and processor counts. It has been shown that modeling the response of a system's memory hierarchy to an application's workload is crucial for accurately predicting its performance on today's systems with deep their memory hierarchies. The current state-of-the-art works well for weak scaling (i.e., increasing the processor count proportionally with input). PERI is developing advanced schemes for modeling application performance, such as by using neural networks.[4] We are also exploring variations of existing techniques and parameterized statistical models built from empirical observations to predict application scaling. We are also pursuing methods for automated extrapolation of scaling models, as a function of increasing processor count, while holding the input constant.[5] One of our goals is to provide the ability to reliably forecast the performance of a code on a machine size that has not yet been built.

Within PERI, we are also extending our framework to model communication performance as a function of the type, size, and frequency of application messages, and the characteristics of the interconnect. Several parallel communication models have been developed that predict performance of message-passing operations based on system parameters.[6][7][8] Assessing the parameters for these models within local area networks is relatively straightforward and the methods to approximate them have already been established and are well understood.[8][9] Our models, which are similar to PlogP, capture the effects of network bandwidth and latency; however, a more robust model must also account for noise, contention and concurrency limits. We are developing performance models directly from observed characteristics of applications on existing architectures. Predictions from such models can serve as the basis to optimize collective MPI operations,[10] and permit us to predict network performance in a very general way. This work

[2] Bailey, D. H., Snavely, A. "Performance Modeling: Understanding the Present and Predicting the Future," EuroPar 2005 , September 2005, Lisbon.

[3] Hoisie, A., Lubeck, O., Wasserman, H. "Performance and Scalability Analysis of Teraflop-Scale Parallel Architectures Using Multidimensional Wavefront Applications," *International Journal of High Performance Computing Applications*, Vol. 14 (2000), no. 4, pg 330-346.

[4] Ipek, E., de Supinski, B. R., Schulz, M., McKee, S. A. "An Approach to Performance Prediction for Parallel Applications," Euro-Par 2005, Lisbon, Portugal, Sept. 2005.

[5] Weinberg, J., MCracken, M. O., Snavely, A., Strohmaier, E. "Quantifying Locality In The Memory Access Patterns of HPC Applications," *Proceedings of SC2005*, Seattle, WA, Nov. 2005.

[6] Culler, D., Karp, R., Patterson, D., Sahay, A., Schauser, K. E., Santos, S., Subramonian, R., von Eicken, T. "LogP: Towards a Realistic Model of Parallel Computation," *Proceedings of the Fourth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, ACM Press, 1993, pg. 1-12.

[7] Alexandrov, A., Ionescu, M. F., Schauser, K. E., Scheiman, C. "LogGP: Incorporating long messages into the LogP model," In *Proceedings of the 7th annual ACM Symposium on Parallel Algorithms and Architectures*, ACM Press, 1995, pg. 95-105.

[8] Kielmann, T., Bal, H. E., Verstoep, K. "Fast Measurement of LogP Parameters for Message Passing Platforms," in Jose D.P. Rolim, editor, *IPDPS Workshops*, volume 1800 of Lecture Notes in Computer Science, pp. 1176-1183, Cancun, Mexico, May 2000. Springer-Verlag.

[9] Culler, D., Lui, L. T., Martin, R. P., Yoshikawa, C. "Assessing Fast Network Interfaces," *IEEE Micro*, Vol. 16 (1996), pg. 35-43.

[10] Pjesivac-Grbovic, J., Angskun, T., Bosilca, G., Fagg, G., Gabriel, E., Dongarra, J. "Performance Analysis of MPI Collective Operations," *4th International Workshop on Performance Modeling, Evaluation, and Optmization of Parallel and Distributed Systems* (PMEO-PDS 2005) Denver, CO, Apr. 2005.

will require us to develop a new open-source network simulator to analyze communication performance.

Finally, we will reduce the time needed to develop models, since automated tuning requires on-the-fly model modification. For example, a compiler, or application, may propose a code change in response to a performance observation, and need an immediate forecast of the performance impact of the change. Dynamic tracing, the foundation of current modeling methods, requires running existing codes and can be quite time consuming. Static analysis of binary executables can make trace acquisition much faster by limiting it to only those features that are not known before execution. User annotations[11] can broaden the reach of modeling by specifying at a high level the expected characteristics of code fragments. Application phase modeling can reduce the amount of data required to form models. We are exploring less expensive techniques to identify dynamic phases through statistical sampling and time-series cluster analysis. For on-the-fly observation, we are using DynInst to attach to a running application, slow it down momentarily to measure something, then detach.[12] In PERI, we will advance automated, rapid, machine-independent model formation to push the efficacy of performance modeling down into lower levels of the application and architecture lifecycle.

3. Automatic Performance Tuning

In discussions with application scientists, it is clear that users want to focus on their science and not be burdened with optimizing their code's performance. Thus, the ideal performance tool analyzes and optimizes performance without human intervention, a long-term vision that we term automatic performance tuning. This vision encompasses tools that analyze a scientific application, both as source code and during execution, generate a space of tuning options, and search for a near-optimal performance solution. There are numerous daunting challenges to realizing the vision, including enhancement of automatic code manipulation tools, automatic run-time parameter selection, automatic communication optimization, and intelligent heuristics to control the combinatorial explosion of tuning possibilities. On the other hand, we are encouraged by recent successful results such as ATLAS, which has automatically tuned components of the LAPACK linear algebra library.[13] We are also studying techniques used in the highly successful FFTW library[14] and several other related projects.[15] [16] [17] The PERI strategy for automatic performance tuning is presented in greater detail in this section of this paper.

Figure 1 illustrates the automated performance tuning process and integration we are pursuing in PERI. We are attempting to integrate performance measurement and modeling techniques with code transformations to create an automated tuning process for optimizing complex codes on large-scale architectures. The result will be an integrated compile-time and run-time optimization methodology that can reduce dependence on human experts and automate key aspects of code optimization. The color and shape code in Figure 1 indicates the processes associated with the automation of empirical tuning on either libraries or whole applications. Blue rectangles indicate specific tools or parts of tools to support automated empirical tuning. Yellow ovals indicate activities that are part of a code that is using automatic tuning at runtime. Green hexagons indicate information may be supplied to guide the optimization selection during empirical tuning. The large green hexagon lists the type of information that may be used.

As shown in Figure 1, the main input to the automatic tuning process is the application source code. In addition, there may also be external code (e.g., libraries),

[11] Alam, S. R., Vetter, J. S. "A Framework to Develop Symbolic Performance Models of Parallel Applications," *Proc. 5th International Workshop on Performance Modeling, Evaluation, and Optimization of Parallel and Distributed Systems* (PMEO-PDS 2006), 2006.

[12] Buck, B. R., Hollingsworth, J. K. "An API for Runtime Code Patching," *Journal of High Performance Computing Applications*, Vol. 14 (2000) no. 4.

[13] Whaley, C., Petitet, A., Dongarra, J. "Automated Empirical Optimizations of Software and the ATLAS Project," *Parallel Computing*, Vol. 27 (2001), no. 1, pg. 3-25.

[14] Frigo, M., Johnson, S. "FFTW: An Adaptive Software Architecture for the FFT," *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, Seattle, Washington, May 1998.

[15] Bilmes, J., Asanovic, K., Chin, C. W., Demmel, J. "Optimizing Matrix Multiply using PHi-PAC: a Portable, High-Performance, ANSI C Coding Methodology," *Proceedings of the Inter-national Conference on Supercomputing*, Vienna, Austria, ACM SIGARCH, July 1997.

[16] Vuduc, R., Demmel, J., Yelick, K. "OSKI: A Library of Automatically Tuned sparse Matrix Kernels," *Proceedings of SciDAC 2005*, Journal of Physics: Conference Series, June 2005.

[17] Chen, C., Chame, J., Hall, M. "Combining Models and Guided Empirical Search to Optimize for Multiple Levels of the Memory Hierarchy," *Proceedings of the Conference on Code Generation and Optimization*, March, 2005.

ancillary information such as performance models or annotations, sample input data, and historical data from previous executions and analyses. With these inputs, we anticipate that the automatic tuning process involves the following steps:

- **Triage**. This step involves performance measurement, analysis and modeling to determine whether an application has opportunities for optimization.
- **Semantic analysis**. This step involves analysis of program semantics to support safe transformation of the source code, including traditional compiler analyses to determine data and control dependencies. Here we can also exploit semantic information provided by the user.
- **Transformation**. Transformations include traditional optimizations such as loop optimizations and in-lining, as well as more aggressive data structure reorganizations and domain-specific optimizations. Tiling transformations may be parameterized to allow for input size and machine characteristic tuning. Unlike traditional compiler transformations, we allow user input.
- **Code generation**. The code generation phase produces a set of possible implementations to be considered. Code generation may either come from general transformations to source code in an application or from a domain-specific tool that produces a set of implementations for a given computation, as is the case with the ATLAS BLAS generator.
- **Offline search**. This phase evaluates the generated code to select the "best" version. Offline search entails running the generated code and searching for the best-performing implementation. The search process may be constrained by guidance from a performance model or user input. By viewing these constraints as guidance, we allow the extremes of pure search-based, model-based, or user-directed, as well as arbitrary combinations.
- **Application assembly**. At this point, the components of optimized code are integrated to produce an executable code, including possible instrumentation and support for dynamic tuning.

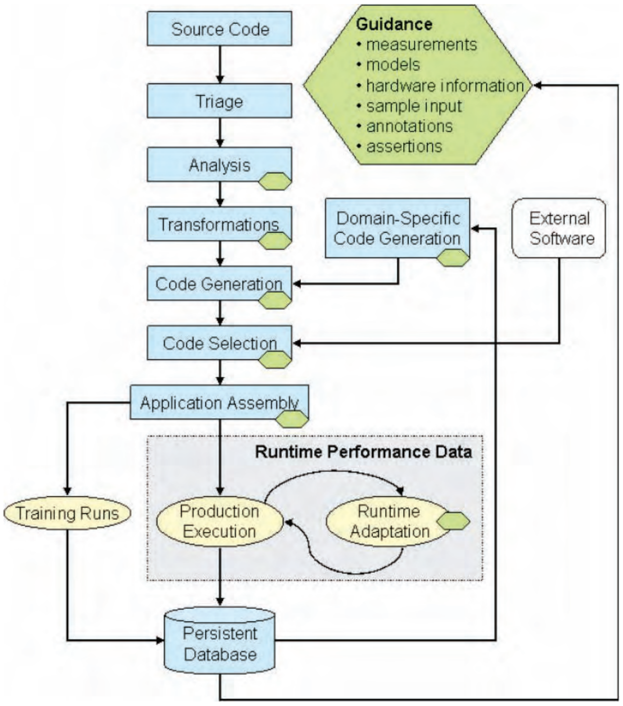


Figure 1. The PERI automatic tuning workflow

- **Training runs**. Training runs involve a separate execution step designed mainly to produce performance data for feedback into the optimization process. This step may be used prior to a large run to have the code well-tuned for a particular input set.
- **Online adaptation**. Finally, optimizations may occur during production runs, especially for problems or machines whose optimal configuration changes during the execution.

Automatic tuning of a particular application need not involve all of these steps. Furthermore, there will likely not be a single automatic tuning tool, but rather a suite of interacting tools that are themselves research projects.

A key part of the automatic tuning process is the maintenance of a persistent store of performance information from both training and production runs. Of particular concern are changes in the behavior of production codes over time. Such changes can be symptomatic of changes in the hardware, of the versions and configuration of system software, of changes to the application, or of changes to problems being solved. Regardless of the source, such changes require analysis and remediation. The problem of maintaining persistent performance data is recognized across the HPC community. PERI therefore formed a Performance Database Working Group, which involves PERI researchers as well as colleagues at the University of Oregon, Portland State University, and Texas A&M University. The group has developed technology for storing performance data collected by a number of performance measurement and analysis tools, including TAU, PerfTrack, Prophesy, and SvPablo. The PERI Database system provides web interfaces that link to the performance data and analysis tools in each tool's home database.

4. Application Engagement

The key long-term research objective of PERI is to automate as much of the performance tuning process as possible. Ideally, in five years we will produce a prototype of the kind of system that will free scientific programmers from the burden of tuning their codes, especially when simply porting from one system to another. While this may offer today's scientific programmers hope for a brighter future, it does little to help with the immediate problems they face as they try ready their codes for Petascale. PERI has therefore created a third activity that we are calling application engagement, wherein PERI researchers will bring their tools and skills to bear in order both to help DOE meet its performance objectives and to ground our own research in practical experience. This section discusses the current status of our application engagement activities.

PERI has a two-pronged application engagement strategy. Our first strategy is establishing long term liaison relationships with many of the application teams. PERI liaisons who work with application teams without significant, immediate performance optimization needs provide these application teams with advice on how to collect performance data and track performance evolution, and ensure that PERI becomes aware of any changes in these needs. For application teams with immediate performance needs, the PERI liaison works actively with the team to help them meet their needs, utilizing other PERI personnel as needed. The status of a PERI liaison activity, passive or active, changes over time as the performance needs of the application teams change. As of June 2007, PERI is working actively with six application teams and passively with ten others. The nature of each interaction is specific to each application team.

The other primary PERI application engagement strategy is tiger teams. A tiger team works directly with application teams with immediate, high-profile performance requirements. Our tiger teams, consisting of several PERI researchers, strive to improve application performance by applying the full range of PERI capabilities, including not only performance modeling and automated tuning research but also in-depth familiarity with today's state-of-the-art performance analysis tools. Tiger team assignments are of a relatively short duration, lasting between 6 and 12 months. As of June 2007, PERI tiger teams are working with two application codes that will be part of the 2007 JOULE report: S3D[18] and GTC_s.[19] We have already identified significant opportunities for performance improvements for both applications. Current work is focused on providing these improvements through automated tools that support the continuing code evolution required by the JOULE criteria.

5. Summary

The Performance Engineering Research Institute was created to focus on the increasingly difficult problem of achieving high scientific throughput on large-scale computing systems. These performance challenges arise not only from the scale and complexity of leadership class computers, but also from the increasing sophistication of today's scientific software. Experience has shown that scientists want to focus their programming efforts on discovery and do not want to be burdened by the need to constantly refine their codes to maximize performance. Performance tools that they can use themselves are not embraced, but rather viewed as a necessary evil.

To alleviate scientists from the burden of performance tuning, PERI has embarked on a research program addressing three different aspects of performance tuning: performance modeling of applications and systems; automatic performance tuning; and application engagement and tuning. Our application engagement activities are intended to help scientists address today's performance related problems. We hope that our automatic performance tuning research will lead to technology that, in the future, will significantly reduce this burden. Performance modeling in-forms both of these activities.

While PERI is a new project, as are all SciDAC-2 efforts, it builds on five years of SciDAC-1 research and decades of prior art. We believe that PERI is off to a good start, and that its investigators have already made contributions to SciDAC-2 and to DOE's 2007 Joule codes. We confidently look forward to an era of Petascale computing in which scientific codes migrate amongst a variety of leadership class computing systems without their developers being overly burdened by the need to continually refine them so as to achieve acceptable levels of throughput.

[18] Echekki, T., Chen, J. H. "DNS of Autoignition in Nonhomogeneous Hydrogen-Air Mixtures," *Combust. Flame*, Vol.134 (2003), pg. 169-191.

[19] Lin, Z., Hahm, T. S., Lee, W. W., Tang, W. M., White, R. B. "Turbulent Transport Reduction by Zonal Flows: Massively Parallel Simulations," *Science*, 281 (1998), pg. 1835-1837.

Creating Software Tools and Libraries for Leadership Computing

John Mellor-Crummey
Rice University

Peter Beckman
Argonne National Laboratory

Keith Cooper
Rice University

Jack Dongarra
University of Tennessee, Knoxville

William Gropp
Argonne National Laboratory

Ewing Lusk
Argonne National Laboratory

Barton Miller
University of Wisconsin, Madison

Katherine Yelick
University of California, Berkeley

1. Center for Scalable Application Development Software

The Department of Energy's (DOE) Office of Science is deploying leadership computing facilities, including a Blue Gene/P system at Argonne National Laboratory and a Cray XT system at Oak Ridge National Laboratory, with the aim of catalyzing scientific discovery. These emerging systems composed of tens of thousands of processor cores are beginning to provide immense computational power for scientific simulation and modeling. However, harnessing the capabilities of such large-scale microprocessor-based, parallel systems is daunting for application developers. A grand challenge for computer science is to develop software technology that simplifies using such systems.

To help address this challenge, in January 2007 the Center for Scalable Application Development Software (CScADS)[1] was established as a partnership between Rice University, Argonne National Laboratory, University of California – Berkeley, University of Tennessee – Knoxville, and University of Wisconsin – Madison. As part of the DOE's Scientific Discovery through Advanced Computing (SciDAC) program, CScADS is pursuing an integrated set of activities that aim to increase the productivity of DOE computational scientists by catalyzing the development of software tools and libraries for leadership computing platforms. These activities include workshops to engage the research community in the challenges of leadership-class computing, research and development of open-source software, and work with computational scientists to help them develop codes for leadership computing platforms.

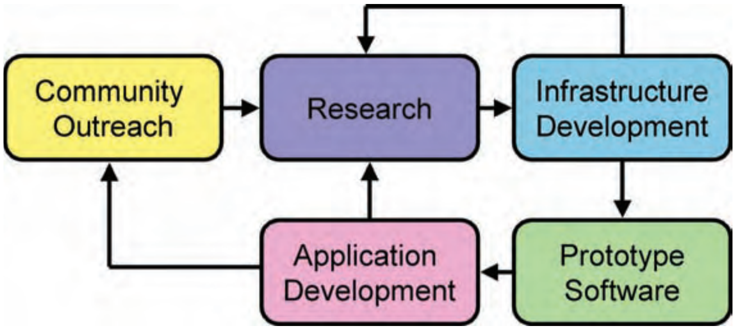


Figure 1. Relationship between CScADS activities.

Figure 1 illustrates the relationships between the Center's activities. The flow of ideas originates from two sources: workshops for community outreach and vision-building, and direct involvement with application development. These activities focus research efforts on important problems. In turn, research drives the infrastructure development by identifying capabilities that are needed to support the long-range vision. Infrastructure feeds back into the research program, but also supports prototyping of software tools that support further application development. Finally, experiences by developers using prototype compilers, tools and libraries will spur the next cycle of research and development.

[1] CScADS - http://cscads.rice.edu/

First, we briefly describe each of the Center's activities in a bit more detail. Then, we describe the themes of CScADS research. Finally, we conclude with a brief discussion of ongoing work.

1.1 Community Outreach and Vision-Building

Achieving petascale performance with applications will require a close collaboration between scientists developing computational models and computer science teams developing enabling technologies. To engage the community in the challenges and to foster interdisciplinary collaborations, we have established the *CScADS Summer Workshops* – an annual series of workshops that will focus on topics related to scalable software for the DOE's leadership-class systems. In July 2007, we held our first series of four workshops in Snowbird, Utah.

- *Automatic Tuning for Petascale Systems*. This workshop brought together researchers to discuss some of the code generation challenges for multicore processors that are the building blocks for emerging petascale systems, to identify some of the opportunities afforded by the use of automatic tuning techniques, and to explore opportunities for collaboration among tuning researchers.
- *Performance Tools for Petascale Computing*. This workshop brought together researchers to discuss the challenges of measurement, attribution, analysis, and presentation of application performance for leadership computing platforms. An important workshop goal was to explore opportunities for research teams to break their software into components to foster community collaboration and accelerate development of effective performance tools for petascale systems.
- *Petascale Architectures and Performance*. This workshop brought together computer scientists and SciDAC application scientists who aim to develop codes for the leadership computing platforms. The goal of this workshop was to introduce the computer scientists to the SciDAC applications and to familiarize the application scientists with the emerging leadership computing platforms, software libraries and tools for parallel computing, and effective strategies for parallel programming.
- *Libraries and Algorithms for Petascale Applications*. This workshop brought together computer scientists working on algorithms and libraries with members of the SciDAC application teams. The principal workshop goal was to identify challenges for library and algorithm developers from the needs of the SciDAC applications and to foster collaboration between the communities. Workshop topics included the use of multicore processors and the use of automatic tuning in libraries.

The latter two workshops included "hands-on" sessions in which computer scientists and application scientists collaboratively explored application challenges on leadership computing platforms.

1.2 Research and Development

Several national reports, such as the 2000 report *Information Technology Research: Investing in our Future* by the President's Information Technology Advisory Committee, have pointed out that open-source software represents an opportunity to address the shortage of software support for programming high-end systems. The power of this approach has been amply demonstrated by the success of Linux in fostering the development of operating systems for high-performance clusters.

The CScADS research program focuses on strategies for improving the productivity of application developers for developing high-performance codes for leadership-class machines. Rather than attack a narrow range of problems within this space, CScADS will explore a broad spectrum of issues because we believe that there is a high degree of synergy to be exploited.

Research on software support for high-end systems cannot be performed in a vacuum. Direct interaction between application developers and enabling technologies teams can clarify the problems that need to be addressed, yield insight into strategies for overcoming performance bottlenecks, identify how those strategies might be automated, and produce a vision for new tools and programming systems.

1.3 Open-Source Software Infrastructure

To facilitate the research, both within CScADS and in the community at large, we are developing the *CScADS Open Software Suite*. This suite will include an open-source software infrastructure to support compiler/programming-language research, development, and evaluation based on the Open64 compiler as well as Rice's D System compiler infrastructure. Other components will include software infrastructure for performance tools, including support for binary analysis, instrumentation, data collection, and measurement interpretation that will draw from Rice's HPCToolkit[2] and Wisconsin's Paradyn[3] and Dyninst tools, and a range of libraries that help harness the power of leadership-class platforms composed of multicore processors.

2. CScADS Research Themes

In CScADS, we have begun a broad program of research on software to support scalability in three dimensions: productivity, homogeneous scalability, and platform heterogeneity. We briefly outline the themes of this work in each of these areas.

2.1 Rapid Construction of High-Performance Applications

An application specification is high level if (1) it is written in a programming system that supports rapid prototyping; (2) aside from algorithm choice, it does not include any hardware-specific programming strategies (e.g., loop tiling); and (3) it is possible to generate code for the entire spectrum of different computing platforms from a single source version. The goal of CScADS productivity research is to explore how we can transform such high-level specifications into high-performance implementations for leadership-class systems.

For higher productivity, we believe that developers should construct high-performance applications by using scripting languages to integrate domain-specific component libraries. At Rice we have been exploring a strategy, called *telescoping languages*, to generate high-performance compilers for scientific scripting languages. The fundamental idea is to preprocess a library of components to produce a compiler that understands and optimizes component invocations as if they were language primitives. As part of this effort, we have been exploring analysis and optimization based on inference about generalized types. A goal of CScADS research is to explore how we can adapt these ideas to optimize programs based on the Common Component Architecture (CCA).

2.2 Scaling to Homogeneous Parallel Systems

Achieving high performance on a modern microprocessor, though challenging, is not by itself enough for SciDAC applications; in addition, applications must be able to scale to the thousands or even hundreds of thousands of processors that make up a

[2] HPCToolkit - http://www.hipersoft.rice.edu/hpctoolkit/

[3] Paradyn - http://www.paradyn.org/

petascale computing platform. Two general classes of software systems are needed to make this feasible: (1) tools that analyze scalable performance and help the developer overcome bottlenecks, and (2) compiler support that can take higher-level languages and map them efficiently to large numbers of processors.

2.2.1 Tools for Scalable Parallel Performance Analysis and Improvement

Effectively harnessing leadership-class systems for capability computing is a grand challenge for computer science. Running codes that are poorly tuned on such systems would waste these precious resources. To help users tune codes for leadership-class systems, we are conducting research on performance tools that addresses the following challenges:

Analyzing integrated measurements. Understanding application performance requires capturing detailed information about parallel application behavior, including the interplay of computation, data movement, synchronization, and I/O. We are focusing on analysis techniques that help understand the interplay of these activities.

Taming the complexity of scale. Analysis and presentation techniques must support top-down analysis to cope with the complexity of large codes running on thousands of processors. To understand executions on thousands of processors, it is not practical to inspect them individually. We are exploring statistical techniques for classifying behaviors into equivalence classes and differential performance analysis techniques for identifying scalability bottlenecks.

Coping with dynamic parallelism. The arrival of multicore processors will give rise to more dynamic threading models on processor nodes. Strategies to analyze the effectiveness of dynamic parallelism will be important in understanding performance on emerging processors.

This work on performance tools extends and complements activities in the Performance Engineering Research Institute (PERI). The CScADS tools research and development will build upon work at Rice on HPCToolkit and work at Wisconsin on Dyninst as well as other tools for analysis and instrumentation of application binaries. An outcome of this effort will be shared interoperable components that will accelerate development of better tools for analyzing the performance of applications running on leadership class systems.

2.2.2 Compiler Technology for Parallel Languages

The principal stumbling block to using parallel computers productively is that parallel programming models in wide use today place most of the burden of managing parallelism and optimizing parallel performance on application developers. We face a looming productivity crisis if we continue programming parallel systems at such a low level of abstraction, as these parallel systems increase in scale and architectural complexity. As a component of CScADS research, we are exploring a range of compiler technologies for parallel systems ranging from technologies with near-term impact to technologies for higher level programming models that we expect to pay off further in the future. This work is being done in conjunction with the DOE-funded Center for Programming Models for Scalable Parallel Computing. Technologies that we are exploring include:

Partitioned global address space (PGAS) languages. Communication optimization will be critical to the performance of PGAS languages on large-scale systems. As part of CScADS, we are enhancing the Open64 compiler infrastructure to support compile-time communication analysis and optimization of Co-Array Fortran and UPC.

Global array languages. High-level languages that support data-parallel programming using a global view offer a dramatically simpler alternative for programming parallel systems. Programming in such languages is simpler; one simply reads and writes shared variables without worrying about synchronization and data movement. An application programmer merely specifies how to partition the data and leaves the details of partitioning the computation and choreographing communication to a parallelizing compiler. Having an HPF program achieve over 10 TFLOPS on Japan's Earth Simulator has rekindled interest in high-level programming models within the US. Research challenges include improving the expressiveness, performance, and portability of high-level programming models.

Parallel scripting languages. Matlab and other scripting languages boost developer productivity both by providing a rich set of library primitives and by abstracting away mundane details of programming. Ongoing work at Rice is exploring compiler technology for Matlab. Work at Tennessee involves parallel implementations of scripting languages such as Matlab, Python, and Mathematica. As a part of this project, we are exploring compiler and run-time techniques that will enable such high-level programming systems to scale to much larger computation configurations while retaining support for most languages features.

2.2.3 Support for Multicore Platforms

Multicore chips will force at least two dimensions of parallelism into scalable architectures: (1) on-chip, shared-memory parallelism and (2) cross-chip distributed-memory parallelism. Many architects predict that with processor speed improvements slowing, the number of cores per chip is likely to double every two years. In addition, many of the contemplated architectures will incorporate multi-threading on each of the cores, adding a third dimension of parallelism. Based on this increased complexity, we see three principal challenges in dealing with scalable parallel systems constructed from multicore chips.

- *Decomposing available parallelism and mapping it well to available resources*. For a given loop nest, we will need to find instruction-level parallelism to exploit short-vector operations, multi-threaded parallelism to map across multiple cores, and outer-loop parallelism to exploit an entire scalable system.
- *Keeping multiple cores busy requires that more data be transferred from off-chip memory*. In the near term, given the limitations on sockets, the aggregate off-chip bandwidth will not scale linearly with the number of cores. For this reason, it will be critical to transform applications to achieve high levels of cache reuse.
- *Choreographing parallelism and data movement*. Rather than having cores compute independently, coordinating their computation with synchronization can improve reuse.

We are pursuing three approaches to cope with the challenges of multicore computing. First, Tennessee is exploring the design of algorithms and component libraries for systems employing multicore chips. This work seeks to achieve the highest possible performance, produce useful libraries, and drive the research on compilation strategies and automatic tuning for multicore chips. Second, Rice is exploring compiler transformations to exploit multicore processors effectively by carefully partitioning and scheduling computations to enhance inter-core data reuse. Third, Argonne is exploring the interaction of multi-threaded application programs with systems software such as node operating systems and communication libraries such as MPI.

2.3 Portability and Support for Heterogeneous Platforms

The third dimension of scalability is mapping an application to different sequential and parallel computing platforms. Over the lifetime of an application, the effort spent in porting and retuning for new platforms can often exceed the original implementation effort. In support of portability, we are initially focusing on obtaining the highest possible performance on leadership-class machines. In addition, we will explore compilation and optimization of applications to permit them to run efficiently on computer systems that incorporate different kinds of computational elements, such as vector/SIMD and scalar processors.

2.3.1 Automatic Tuning to New Platforms

The success of libraries such as ATLAS[4] and FFTW[5] has increased interest in automatic tuning of components and applications. The goal of research in this area is to develop compiler and run-time technology to identify which loop nests in a program are critical for high performance and then restructure them appropriately to achieve the highest performance on a target platform.

The search space for alternative implementations of loop nests is too big to explore exhaustively. We have been exploring several strategies to reduce the cost of searching for the best loop structure. By leveraging capabilities of Rice's HPCToolkit, we can pinpoint sources of inefficiency at the loop level, which can guide exploration of transformation parameters. Also, we have been employing model guidance along with search to dramatically reduce the size of the search needed for good performance.[6]

As part of CScADS, the Rice and Tennessee groups are continuing their efforts based on LoopTool, HPCToolkit, and ATLAS 2, with a focus on pre-tuning component libraries for various platforms. This work will provide variants of arbitrary component libraries optimized for different platforms and different application contexts, much as Atlas does today. A second group at Rice is extending adaptive code optimization strategies to tune components. This work will explore adaptive transformations and aggressive interprocedural optimization.

2.3.2 Compiling to Heterogeneous Computing Platforms

Emerging high-end computing architectures are beginning to have heterogeneous computational components within a single system. Exploiting these features (or even coping with them) will be a challenge. We believe that new techniques must be incorporated into compilers and tools to support portable high-performance programming. To date, our work has explored compilation for chips with attached vector units (SSE on Intel chips, Altivec on the IBM G5) and code generation for Cell.[7] We are building upon this work to develop compiler techniques for partitioning and mapping computations onto the resources to which they are best suited. These techniques will be critical for effective use of systems that incorporate both vector and scalar elements in the same machine, such as those outlined in Cray's strategy for "adaptive supercomputing."

3. Recent and Ongoing Work

To date, work in CScADS has included both research and development of a range of technologies necessary to support leadership computing, as well as direct involvement with SciDAC application teams. We briefly summarize a few of these efforts.

[4] Whaley, C., Petitet, A., Dongarra, J. "Automated empirical optimizations of software and the ATLAS project," *Parallel Computing*, Vol. 27 (2001), no. 1, pg. 3-25.

[5] Frigo, M. "A fast Fourier transform compiler," *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation*, Atlanta, Georgia, May 1999.

[6] Qasem, A., Kennedy, K. "A cache-conscious profitability model for empirical tuning of loop fusion," *Proceedings of the 2005 International Workshop on Languages and Compilers for Parallel Computing*, Hawthorne, NY, October 20-22, 2005.

[7] Zhao, Y., Kennedy, K. "Dependence-based code generation for a CELL processor," *Proceedings of the 19th International Workshop on Languages and Compilers for Parallel Computing (LCPC)*, New Orleans, Louisiana, November 2 - 4, 2006.

3.1 Research and Development of Software for Leadership Computing

Rice and Wisconsin have begun collaborative development of a series of performance-tool components that can serve as community infrastructure for performance tools for leadership computing platforms. Initial efforts in this area have been focused on development of multi-platform components for stack unwinding within and across process address spaces that has uses for both debugging and performance analysis, and a library that provides a foundation for sampling-based performance measurement of both statically-linked and dynamically-linked executables.

Berkeley and Tennessee have been collaborating on re-engineering numerical libraries for parallel systems. Initially, this work has been exploring parallel matrix factorization using multi-threading in combination with intelligent scheduling. The new execution model relies on dynamic, dataflow-driven execution and avoids both global synchronization and implicit point-to-point synchronization due to send/receive-style message passing. Experimental results indicate that this strategy can significantly outperform traditional codes by hiding both algorithmic and communication latencies. Future plans call for exploring this programming paradigm for both two-sided linear algebra algorithms and sparse matrix algorithms.

Argonne has been exploring the implementation and performance evaluation of MPI support for multi-threading and remote memory access. Experiments with Argonne's own MPI implementation (MPICH) and various vendor implementations have demonstrated the potential contribution these still little-used parts of MPI can make to parallel program performance and have revealed widely varying attention to efficient implementations.

3.2 Application Engagement

As part of the Center's application engagement efforts, Rice has been working closely with several of the SciDAC S3D and GTC application teams to diagnose application performance bottlenecks on leadership-class platforms using a combination of measurement, analysis, and modeling. S3D is a massively parallel solver for turbulent reacting flows.[8] GTC (Gyrokinetic Toroidal Code) is a three-dimensional particle-in-cell (PIC) code used for studying the impact of fine-scale plasma turbulence on energy and particle confinement in the core of tokamak fusion reactors.[9] Our early experiences with both S3D and GTC demonstrate the value of the CScADS approach of tightly coupling computer science research with application development and tuning. Work with these applications has influenced development of software tools for performance measurement and performance modeling, as well as motivated a study of run-time libraries for adaptive data reordering.

Work with S3D uncovered opportunities for using source-to-source tools to tailor code to improve memory hierarchy utilization. This led to refinement of Rice's LoopTool program transformation tool. Applying LoopTool to S3D yielded improved performance of S3D's most memory intensive loop by nearly a factor of three.[10] Additionally, analysis of experiments with S3D on the hybrid Cray XT3/XT4 system showed that the lower memory bandwidth on the XT3 nodes hurts the weak scaling performance of S3D on the hybrid system. Performance on the hybrid system could be improved by proportionally adjusting the partitioning of computation to account for the higher efficiency of the XT4 nodes.

Work with GTC has focused on exploring opportunities for improving memory hierarchy utilization. One component of this effort has been studying the impact of

[8] Monroe, D. "Energy science with digital combustors," SciDAC Review. Fall 2006. http://www.scidacreview.org/0602/html/combustion.html

[9] Krieger, K. "Simulating star power on earth," SciDAC Review, Spring 2006. http://www.scidacreview.org/0601/html/fusion.html

[10] Mellor-Crummey, J. "Harnessing the power of emerging petascale platforms. SciDAC 2007," *Journal of Physics*: Conference Series 78 (2007) 012048.

data structure layout and code organization on the spatial and temporal locality present in data access patterns. A detailed study of GTC using a performance modeling toolkit developed at Rice[11] identified several opportunities for improving application performance. These included reorganizing the particle data structures to improve spatial reuse in the charge deposition and particle pushing phases of the application, using loop fusion to increase temporal reuse of particle data, and transforming the code to increase instruction-level parallelism and reduce translation look-aside buffer misses. A study of the transformed code on an Itanium2 system showed that our code transformations improved performance by 33%. Code modifications have been provided back to the application team. Ongoing work is exploring on-line adaptive reordering of particle data to improve temporal locality for the cell data structures during the charge deposition and particle pushing phases. Preliminary experiments indicate that this approach offers the potential for substantially improving performance.

An outcome of the CScADS summer workshop *Libraries and Algorithms for Petascale Applications* was a substantial improvement in I/O scaling and performance of the Omega3P simulation tool under development at the Stanford Linear Accelerator Center. Discussions at the workshop led to the use of collective communication patterns to avoid scaling bottlenecks associated with reading input data. Additionally, adjusting the application to use parallel netCDF and MPI-IO reduced the time for writing output data by a factor of 100 when Omega3P was run on thousands of processors on the Cray XT system at Oak Ridge. These improvements dramatically enhanced the scalability of Omega3P.

Acknowledgement

The Center for Scalable Application Development Software is supported by cooperative agreement number DE-FC02-07ER25800 from the Department of Energy's Office of Science.

[11] Marin, G., Mellor-Crummey, J. "Understanding unfulfilled memory reuse potential in scientific applications," Technical Report TR07-6, Department of Computer Science, Rice University, October 2007.

DOE's SciDAC Visualization and Analytics Center for Enabling Technologies – Strategy for Petascale Visual Data Analysis Success

E. Wes Bethel
Lawrence Berkeley National Laboratory

Chris Johnson
University of Utah

Cecilia Aragon
Prabhat
Oliver Rübel
Gunther Weber
Lawrence Berkeley National Laboratory

Valerio Pascucci
Hank Childs
Peer-Timo Bremer
Brad Whitlock
Lawrence Livermore National Laboratory

Sean Ahern
Jeremey Meredith
George Ostrouchov
Oak Ridge National Laboratory

Ken Joy
Bernd Hamann
Christoph Garth
University of California, Davis

Martin Cole
Charles Hansen
Steven Parker
Allen Sanderson
Claudio Silva
Xavier Tricoche
University of Utah

Introduction

Galileo Galilei (15 February 1564 -- 8 January 1642) has been credited with fundamental improvements to early telescope designs that resulted in the first practically usable instrument for observing the heavens. With his "invention," Galileo went on to many notable astronomical discoveries: the satellites of Jupiter, sunspots and the rotation of the sun, and proved the Copernican heliocentric model of the solar system (where the sun, rather than the earth, is the center of the solar system). These discoveries, and their subsequent impact on science and society, would not have been possible without the aid of the telescope – a device that serves to transform the unseeable into the seeable.

Modern scientific visualization, or just visualization for the sake of brevity in this article, plays a similarly significant role in contemporary science. Visualization is the transformation of abstract data, whether it be observed, simulated, or both, into readily comprehensible images. Like the telescope and other modern instruments, visualization has proven to be an indispensable part of the scientific discovery process in virtually all fields of study. It is largely accepted that the term "scientific visualization" was coined in the landmark 1987 report[1] that offered a glimpse into the important role visualization could play in scientific discovery.

Visualization produces a rich and diverse set of output – from the x/y plot to photorealistic renderings of complex multidimensional phenomena. It is most typically "reduced to practice" in the form of software. There is a strong, vibrant, and productive worldwide visualization community that is inclusive of commercial, government and academic interests.

The field of visualization is as diverse as the number of different scientific domains to which it can be applied. Visualization software design and engineering both study and solve what are essentially computer science problems. Much of visualization algorithm conception and design shares space with applied mathematics. Application of visualization concepts (and software) to specific scientific problems to produce insightful and useful images overlaps with cognitive psychology, art, and often the scientific domain itself.

In the present day, the U.S. Department of Energy has a significant investment in many science programs. Some of these programs, carried out under the Scientific Discovery through Advanced Computing (SciDAC) program,[2] aim to study, via simulation, scientific phenomena on the world's largest computer systems. These new scientific simulations, which are being carried out on fractional-petascale sized machines today, generate vast amounts of output data. Managing and gaining insight from such data is widely accepted as one of the bottlenecks in contemporary science.[3] As a result, DOE's SciDAC program includes efforts aimed at addressing data management and knowledge discovery to complement the computational science efforts.

[1] B. McCormick, T. De Fanti et al., "Special Issue on Visualization in Scientific Computing," *Computer Graphics*, 21:6, November 1987.

[2] U.S. Department of Energy, "Scientific Discovery Through Advanced Computing", http://www.scidac.gov/SciDAC.pdf, March 2000.

[3] Mount, R. (ed), The Office of Science Data-Management Challenge – Report from the DOE Office of Science Data-Management Workshops, http://www.slac.stanford.edu/cgi-wrap/getdoc/slac-r-782.pdf, May 2004.

DOE's SciDAC Visualization and Analytics Center for Enabling Technologies – Strategy for Petascale Visual Data Analysis Success

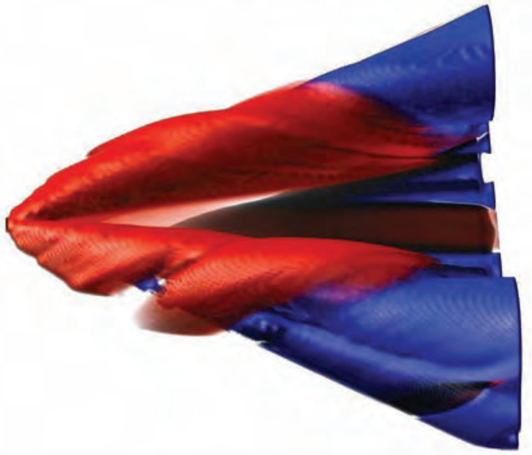


Figure 1. Visualization offers the ability to "see the unseeable."' This image shows visualization of coherent flow structures in a large scale delta wing dataset: Volume rendering of regions of high forward (red) and backward (blue) Finite Time Lyapunov Exponent. Coherent structures appear as surfaces corresponding to the major vortices developing over the wing along the leading edge.[4] Occlusion is a limitation that can be addressed with cropping or clipping. (Image courtesy of X. Tricoche, University of Utah and C. Garth, University of California – Davis)

The focus of this article is on how one group of researchers – the DOE SciDAC Visualization and Analytics Center for Enabling Technologies (VACET) – is tackling the daunting task of enabling knowledge discovery through visualization and analytics on some of the world's largest and most complex datasets and on some of the world's largest computational platforms. As a Center for Enabling Technology, VACET's mission is the creation of usable, production-quality visualization and knowledge discovery software infrastructure that runs on large, parallel computer systems at DOE's Open Computing facilities, and that provides solutions to challenging visual data exploration and knowledge discovery needs of modern science, particularly the DOE science community.

Why Visualization Works So Well

One of the reasons that scientific visualization, and visual data analysis, has proven to be highly effective in knowledge discovery is because it leverages the human cognitive system. Pseudocoloring, a staple visualization technique, performs a mapping of data values to colors in images to take advantage of this very ability. Figure 2 is a good example, where high data values are mapped to a specific color that attracts the eye. Additionally, a very clear 3D structure becomes apparent in this image; it would be virtually impossible to "see" such structure by looking at a large table of numbers. While Figure 2 shows a 3D example, we are all familiar with 2D versions of this technique; the weather report on the evening news often shows pseudo-colored representations of temperature or levels of precipitation overlaid on a map.

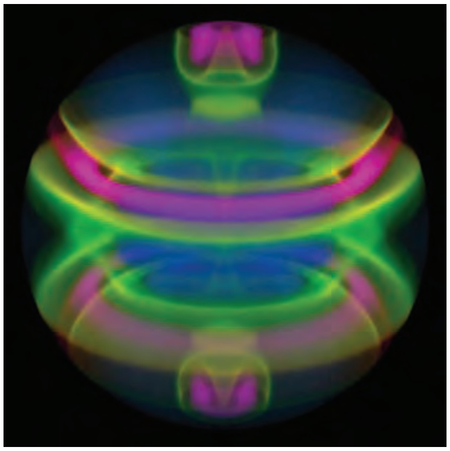


Figure 2. Two types of "features" are immediately visible in this image showing the entropy field of a radiation/hydrodynamic simulation that models the accretion-induced collapse of a star, a phenomena that produces supernovae. One "feature" is the "sandwiching" of high values of entropy between lower values. The other is an overall sense of 3D structure. (Simulation data courtesy of Adam Burrows, University of Arizona, SciDAC Science Application "The Computational Astrophysics Consortium,"' image courtesy of the Visualization Group, Lawrence Berkeley National Laboratory.)

[4] C. Garth, F. Gerhardt, X. Tricoche, H. Hagen. "Efficient Computation and Visualization of Coherent Structures in Fluid Flow Applications." *Transactions on Visualization and Computer Graphics/IEEE Visualization 2007* (accepted for publication).

Surviving the Data Tsunami

Many "tried and true" visualization techniques – like using psuedocoloring to map scalar data values to color – do a great job of leveraging the human cognitive system to accelerate discovery and understanding of complex phenomena. However, we are faced with some difficult challenges when considering the notion of using visualization as a knowledge discovery vehicle on very large datasets. One of many challenges is limited human cognitive bandwidth, which is conveyed in the notional chart shown in Figure 3. This chart conveys that while our ability to generate, collect, store and analyze data grows at a rate tracking the increase in processor speed and storage density, we as humans have fixed cognitive capacity to absorb information. Given that our ability to generate data far exceeds what we can possibly understand, one major challenge for "petascale visual data exploration and analysis" is how to effectively "impedance match" between "limitless data" and a fixed human cognitive capacity.

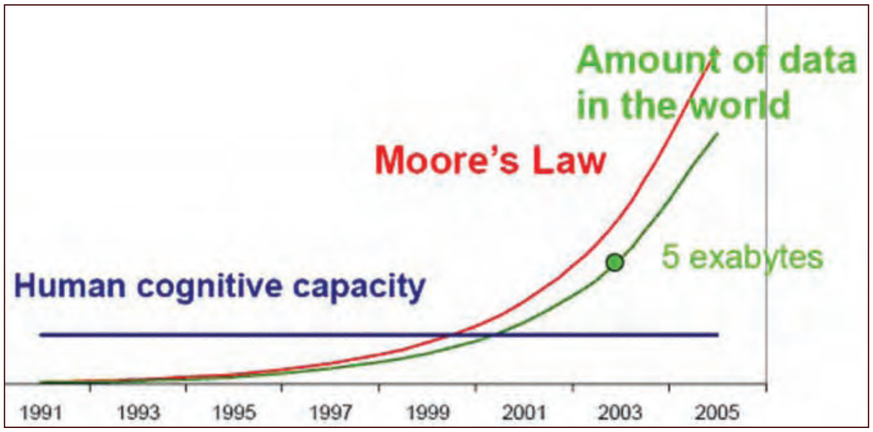


Figure 3. While our ability to generate, collect, store, and analyze data grows at a rate that tracks the increase in processor speed and storage density, our ability as humans to absorb information remains fixed (Illustration adapted from a slide by J. Heer, PARC User Interface Research Group).

> *"What information consumes is rather obvious: it consumes the attention of its recipients. Hence a wealth of information creates a poverty of attention, and a need to allocate that attention efficiently among the overabundance of information sources that might consume it."* – Herb Simon, as quoted by Hal Varian.[5]

In the context of our work – namely, petascale visual data analysis – we are faced with several dilemmas. First, even if we could simply scale up our existing tools and algorithms so they would operate at the petascale rather than the terascale, would the results be useful for knowledge discovery? Second, if the answer to the first question is "no," then how can we help to "allocate attention efficiently among the overabundance of information"?

Let's examine the first question a bit more closely. First, let's assume that we're operating in a gigabyte-sized dataset (10^9 data points), and we're displaying the results in a monitor that has, say, 2 million pixels ($2*10^6$ pixels). For the sake of discussion, let's assume we're going to create and display an isosurface of this dataset. Studies have shown that on the order of about $N^{2/3}$, grid cells in a dataset of size N^3 will contain any given isosurface.[6] In our own work, we have found this estimate to be somewhat low – our results have shown the number to be closer to $N^{0.8}$ for N^3 data. Also, we have found an average of about 2.4 triangles per grid cell will result from the isocontouring algorithm.[7] If we use these two figures as lower and upper bounds, then for our

[5] Varian, H. "The Information Economy," Scientific American, pp 200-201, September 1995.

[6] Bajaj, C., Pascucci, V., Schikore, D. "Fast Isocontouring for Improved Interactivity," In *Proceedings of the 1996 Symposium on Volume Visualization*, pp 39-46, October 1996.

[7] Bowman, I., Shalf, J., Ma, K.-L., Bethel, E. W. "Performance Modeling for 3D Visualization in a Heterogeneous Computing Environment," Technical Report LBNL-56977. Lawrence Berkeley National Laboratory, Berkeley CA, 2004.

gigabyte-sized dataset, we can reasonably expect on the order of between about 2.1 and 40 million triangles for many isocontouring levels. At a display resolution of about 2 million pixels, the result is a depth complexity – the number of objects at each pixel along all depths – of between 1 and 20.

With increasing depth complexity come at least two types of problems. First, more information is "hidden from view." In other words, the nearest object at each pixel hides all the other objects that are further away. Second, if we do use a form of visualization and rendering that supports transparency – so that we can, in principle, see all the objects along all depths at each pixel – we are assuming that a human observer will be capable of distinguishing among the objects in depth. At best, this latter assumption does not always hold true, and at worst, we are virtually guaranteed the viewer will not be able to gain any meaningful information from the visual information overload.

If we scale up our dataset from gigabyte (10^9) to terabyte (10^{12}), then we can expect on the order of between 199 million and 9.5 billion triangles representing a depth complexity ranging between about 80 and 4700, respectively. Regardless of which estimate of the number of triangles we use, we end up drawing the same conclusion: depth complexity and, correspondingly, scene complexity and human workload, grow linearly with the size of the source data. Even if we are able to somehow display all those triangles, we would be placing an incredibly difficult burden on the user. He or she will be facing the impossible task of visually trying to locate "smaller needles in a larger haystack."

The multi-faceted approach we're adopting takes square aim at the fundamental objective: help the scientific researchers more quickly and efficiently do science. In one view, one primary tactical approach that seems promising is to help focus user attention on easily consumable images from the large data collection. We do not have enough space in this brief article to cover all aspects of our team's effort in this regard. Instead, we provide a few details about a couple of especially interesting challenge areas.

Query-Driven Visualization

The term "query-driven visualization" (QDV) refers to the process of limiting visual data analysis processing only to "data of interest."[8] In brief, QDV is about using software machinery combined with flexible and highly useful interfaces to help reduce the amount of information that needs to be analyzed. The basis for the reduction varies from domain to domain, but boils down to "what subset of the large dataset is really of interest for the problem being studied." This notion is closely related to that of "feature detection and analysis," where "features" can be thought of as subsets of the larger population that exhibit some characteristics that are either intrinsic to individuals within the population (e.g., data points where there is high pressure and high velocity) or that are defined as relations between individuals within the population (e.g., the temperature gradient changes sign at a given data point).

For the purposes of our discussion here, we will focus on the first category of features. The second category is also of great interest to our team, where we have developed new technologies for topological data analysis[9] that have proven very useful as the basis for enabling scientific knowledge discovery.

Broadly speaking, QDV consists of three broad conceptual elements. One is how one goes about "specifying interesting." Another is how one displays and analyzes that subset of data. Yet another is the process of storing, indexing, querying and retrieving data subsets from large data archives.

[8] Stockinger, K., Shalf, J., Wu, K., Bethel, E. W. "Query-Driven Visualization of Large Data Sets." In *Proceedings of IEEE Visualization 2005*, pp 167-174, Minneapolis NM, October 2005.

[9] Gyulassy, A., Natarajan, V., Hamann, B., Pascucci, V. "Efficient Computation of Morse-Smale Complexes for Three-Dimensional Scalar Functions," *Transactions on Visualization and Computer Graphics/IEEE Visualization 2007* (accepted for publication).

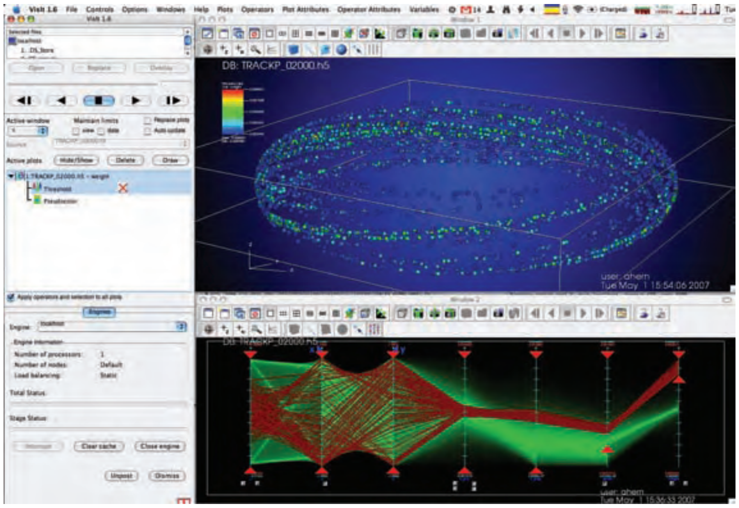
Specifying Queries

In many scientific data analysis applications, "interesting" data can be defined by compound boolean range queries of the form "*(temperature > 1000) AND (0.8 <= density <= 1.0)*". Obviously, one could manually enter such an SQL-like query, but doing so is somewhat clumsy from an interface perspective, but also requires that the user know something about the data characteristics. In many instances, the users are quite familiar with their data, so the expectation of a priori knowledge is not unreasonable. Rather than typing in queries, we propose that a visual interface for specifying queries will result in greater scientific productivity and better serve our mission of enabling data exploration and knowledge discovery.

We have implemented several different types of visual interfaces for specifying queries. The general theme in these implementations is that the visual interface helps the user to formulate queries while at the same time gaining an overall sense of data characteristics. This type of interaction is a variation on a well-known usability design principle called "context and focus," where a given presentation affords the opportunity to see overviews of data (the context) as well as details about specific data of interest (the focus). Numerous works have applied this principle to the effective navigation of complex dataspaces, e.g., application to browsing of hierarchical filesystems.[10]

One example for formulating queries along these lines is an application for exploration of large collections of particle-based datasets produced by the Gyrokinetic Turbulence Code (GTC), which is used to model microturbulence in magnetically confined fusion plasmas.[11] Output from GTC consists of on the order of tens of millions of particles per timestep on present-day computational platforms; this figure is expected to rise at a rate commensurate with growth in computational capacity. From this output, fusion researchers are interested in studying various types of phenomena: formation, evolution and analysis of turbulent structures (eddies, vortices, etc.); and how particle "trapping" and "untrapping" in magnetic fields through microturbulence leads to an erosion of energy efficiency.

We clearly don't want to present an image of the entire dataset at each timestep – the result would be a very cluttered and unintelligible display. Instead, we want to offer the ability for a fusion scientist to focus visual analysis on subsets of data. The result, which is shown below in Figure 4, is an effective context-and-focus interface for rapidly selecting subsets of particles for display.



DB: TRACKP_02000.h5

user: ahern
Tue May 1 15:54:06 2007

Engine: localhost

Engine information

Number of processors: 1
Number of nodes: Default
Load balancing: Static

Total Status:

Stage Status:

DB: TRACKP_02000.h5

user: ahern
Tue May 1 15:36:33 2007

Figure 4. For visual exploration and analysis of GTC data, our implementation provides a visual interface for selecting subsets of particles that meet a set of user-defined criteria. Here, "interesting" is defined as those data points that satisfy a set of multivariate range combinations via the parallel coordinates interface (lower image). The subset satisfying these range conditions then appears in the physical view (top image), where the view may be manipulated, the color transfer changed to draw attention to specific particles based upon other user-defined criteria, or subject to other types of visual or traditional analysis. (Image courtesy S. Ahern, Oak Ridge National Laboratory)

[10] Stasko, J., Zhang, E. "Focus+Context Display and Navigation Techniques for Enhancing Radial, Space-Filling Hierarchy," In *Proceedings of IEEE Information Visualization 2000*, pp57-65, Salt Lake City UT, October 2000.

[11] Lee, W., Ethier, S., Wang, W., Klasky, S. "Gyrokinetic Particle Simulation of Fusion Plasmas: Path to Petascale Computing," *Journal of Physics*: Conference Series 46(2006), pp 73-81, Proceedings of SciDAC 2006. Institute of Physics Publishing, July 2006.

These concepts can be applied to other types of data in other scientific domains, such as exploring the relationships between gene expression levels in cells of a developing organism as shown in Figure 5. These ideas, when combined with multiple linked views where updates in one display are then propagated to other views of the same dataset, offer an extremely powerful framework for rapid exploration of complex data.[12]

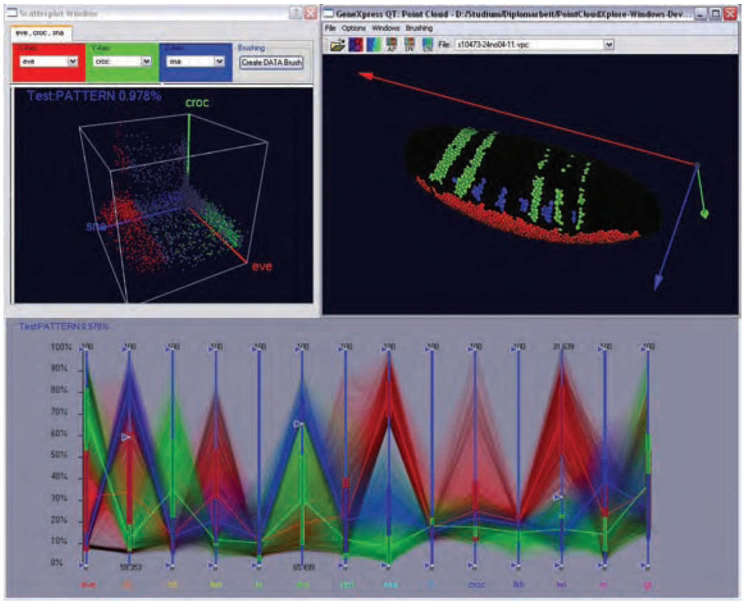


Figure 5. Here, we define three groups of "interesting" data – colored red, green, and blue – with a parallel coordinates interface (bottom pane), and those data points satisfying the multivariate range condition appear in the physical view (upper right). Here, we introduce a third linked view (upper left) that shows a 3D scatterplot – each of the data points from the three "groups of interesting" are colored red, green or blue (according to which "group of interesting" they belong). The three axes of the scatterplot are the expression levels for three specific genes; there are on the order of about 20 different gene expression levels per cell in this dataset – we picked three genes from the group of 20 for the purposes of display. This type of linked view presentation is very helpful in conveying different types of relationships in complex data. (Image courtesy of Oliver Rübel, Lawrence Berkeley National Laboratory)

High Performance Implementation

So far, we've discussed how one might go about specifying queries, or "defining interesting," and have also shown a couple of different ways to present the results that show only "the interesting data." Here, we want to turn our attention to the underlying machinery that makes this kind of approach feasible in high performance implementations suitable for use with very large datasets.

All Computer Science undergraduates are introduced to the idea of binary trees and their use as an indexing data structure. Briefly, if you have a sorted array of data of ***N*** items, you can construct a binary tree that will have ***N-1*** nodes and ***N*** leaves where each interior node partitions the data in deeper nodes and leaves into two groups – "greater than" and "less than or equal to" the value of a key. Once you have constructed this data structure, the search for the data record having the value of some key is performed in $\boldsymbol{log_2 N}$ search steps assuming an optimal, or balanced tree. This basic idea – called tree-based indexing – is widely used in many types of relational and object-oriented database systems. One obvious limitation of this type of approach when considering very large data is that the size of the indexing structure – the tree – is linear with respect to the size of the dataset being indexed. As this size grows larger, we clearly don't want

[12] Rübel, O., Weber, G. H., Keränen, S. V. E., Fowlkes, C. C., Hendriks, C. L., Simirenko, L., Shah, N., Eisen, M., Biggin, M., Hagen, H., Sudar, J., Malik, J., Knowles, D., Hamann, B. "PointCloudXplore: Visual Analysis of 3D Gene Expression Data Using Physical Views and Parallel Coordinates," In *Data Visualization 2006, Proceedings of EuroVis 2006*, pp203-210, Eurographics Association, Aire-la-Ville Switzerland, July 2006.

to incur a commensurately larger storage cost for our search indices. Another problem, which may not be quite as obvious, is that these tree-based approaches require the original data to be sorted. For scientific data, where you typically write the data once then examine it over and over again, this may not be a serious limitation. In some instances, it may simply be impractical to sort the data.

Of greater concern is the so-called "Curse of Dimensionality"[13] The previous paragraph calls out that the storage complexity for a tree-based structure is $O(N)$ when there are N data points. If these data points, or records, have two variables, and we want to create a two-dimensional tree that spans both variables, we end up with a storage complexity of $O(N^2)$. If there are three variables, the storage requirements are of $O(N^3)$. The basic premise is that storage requirements for tree-based indices grow exponentially with respect to the number of variables being indexed. Many modern simulations routinely have on the order of 100 variables that are computed and saved at each time step. It should be obvious that tree-based indexing is simply not practical for large and complex scientific data.

This well-known problem has received a great deal of attention from our colleagues in the field of scientific data management. They have developed a unique technology called "compressed bitmap indices" that have very favorable storage and search complexity.[14] This technology has been applied with great success to index/query problems of some of the world's largest datasets.[15] In a series of collaborative research projects, members of VACET and DOE's Scientific Data Management Center have demonstrated the practicality of combining fast bitmap indexing with high performance visual data analysis, to implement a novel approach to query-driven visualization applied to visual data analysis of problems in combustion modeling[8] and large-scale network traffic analysis.[16]

Adaptive Mesh Refinement Visualization

Adaptive Mesh Refinement (AMR) techniques combine the compact, implicitly specified structure of regular, rectilinear with the adaptivity to changes in scale of unstructured grids. AMR has proven particularly useful for modeling multiscale computational domains that span many orders of magnitude of spatial or temporal scales by focusing solvers on regions where "interesting" physics or chemistry occur. Such domains include applications like astrophysics supernova modeling, where the simulation endeavors to model phenomena that occur at scales ranging from sub-kilometer to interplanetary. AMR avoids the inefficiencies inherent in attempting to model this vast computational domain at a single, fine, homogeneous resolution.

Handling AMR data for visualization is challenging, since coarser information in regions covered by finer patches is superseded and replaced with information from these finer patches. During visualization, it becomes necessary to manage the selection of which resolutions are being used. Furthermore, it is difficult to avoid discontinuities at level boundaries, which, if not properly handled, lead to visible artifacts in visualizations. Due to these difficulties, AMR support as first class data type in production visualization tools has been lacking despite the growing popularity of AMR-based simulations.[17]

[13] Bellman, R. *Adaptive Control Processes: A Guided Tour.* Princeton University Press, 1961.

[14] K. Wu, E. Otoo, and A. Shoshani. "On the Performance of Bitmap Indices for High Cardinality Attributes." In *International Conference on Very Large Data Bases*, Toronto, Canada. September 2004.

[15] Wu, K., Zhang, W.-M., Perevoztchikov, V., Laurent, J., Shoshani, A. "The Grid Collector: Using an Event Catalog To Speed Up User Analysis in a Distributed Environment," *Computing in High Energy and Nuclear Physics (CHEP)*, Interlaken, Switzerland, September 2004.

[16] Stockinger, K., Bethel, E. W., Campbell, S., Dart, E., Wu, K. "Detecting Distributed Scans Using High Performance Query-Driven Visualization," In *Proceedings of SC06 (Supercomputing)*.

[17] Weber, G. H., Beckner, V., Childs, H., Ligocki, T., Miller, M., van Straalen, B., Bethel. E. W., "Visualization Tools for Adaptive Mesh Refinement Data," In *Proceedings of the 4th High End Visualization Workshop* (Tyrol Austria, June 18-22, 2007), pp. 12-25, 2007.

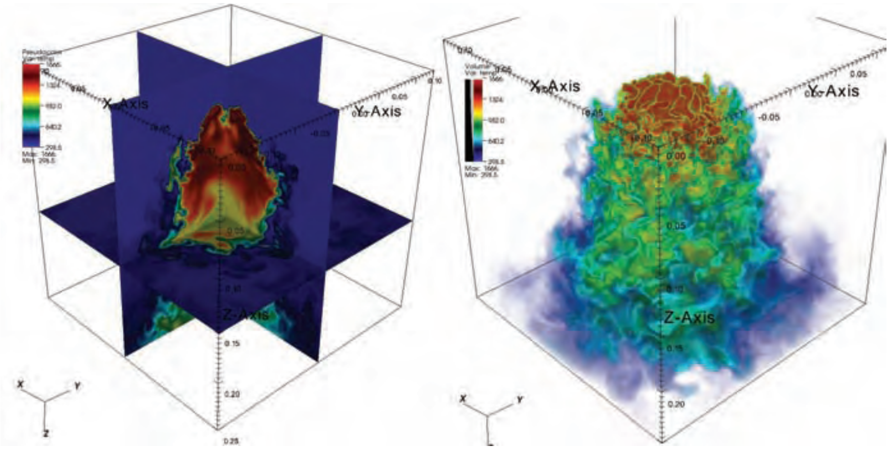
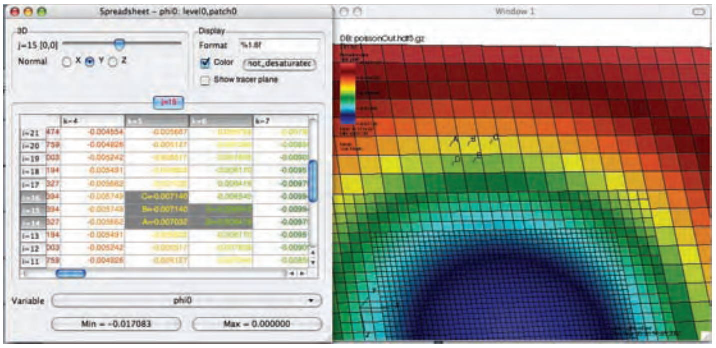


Figure 6. Production-quality visualization of data from an AMR-based simulation of a hydrogen flame. The left panel shows three orthogonal slices colored by temperature (blue is colder, red is hotter). The right panel shows an image produced with volume rendering of the same variable from this dataset. (Simulation data courtesy M. Day and J. Bell, Lawrence Berkeley National Laboratory; images courtesy G. Weber, Lawrence Berkeley National Laboratory).

Through interactions with our computational science stakeholders, VACET is providing production-quality, parallel capable software providing capabilities that fulfills needs in exploratory, analytical and presentation AMR visualization. Our deployment software – VisIt[18] – is an open source visualization tool that accommodates AMR as a first class data type. VisIt handles AMR data as a special case of "ghost data," i.e., data that is used to make computations more efficient, but which is not considered to be part of the simulation result. VisIt tags cells in coarse patches that are available at finer resolution as "ghost" cells, allowing AMR patches to retain their highly efficient native format as rectilinear grids. VisIt offers a rich set of production-quality functions, such as pseudocolor and volume rendering plots (Figure 6), for visualization and analysis of massive scale data sets, making it an ideal candidate to replace specialized AMR visualization tools.


Spreadsheet – phi0: level0,patch0

3D
j=15 [0,0]
Normal ○ X ● Y ○ Z

Display
Format %1.6f
☑ Color not_desaturated
□ Show tracer plane

Variable phi0
Min = -0.017083
Max = 0.000000

Figure 7. Spreadsheet plots are an important tool for debugging AMR codes. They support direct viewing of numerical data in patch cells. VisIt labels selected cells both in Spreadsheet and 3D visualizations allowing users to recognize correspondences quickly and effectively.

Recently, VACET has focused attention on implementing a set of essential debugging features in VisIt so that one of our stakeholders, the DOE SciDAC Applied Partial Differential Equations Center (APDEC), can fully migrate from their project-written and maintained visual data analysis software (ChomboVis) to VisIt. This migration will result in two benefits crucial to APDEC. The first is a cost savings, as APDEC will no

[18] VisIt Visualization Software - http://www.llnl.gov/visit, September 2007.

longer need to expend in-project resources on maintaining visualization software. The second is new AMR visualization capabilities that include the ability to run on parallel machines as well as support for remote and distributed visualization.

We added a new capability in VisIt – AMR spreadsheet plots – that support direct viewing of numerical values on a particular slice of a patch (see Figure 7). This function is essential for debugging and used by AMR code development teams on a daily basis. The new spreadsheet capability is integrated with VisIt's "pick cell" feature, allowing users to "link" them to other plots. Additional new features include the ability to customize the VisIt interface, thereby improving usability so that new users can quickly navigate and employ features familiar to them in their older, retiring software.

While not as visible as the features above, other recent accomplishments include software architecture and engineering work to produce all-important performance improvements. Optimizations in AMR grid processing have produced a ten-fold savings in memory, and support more efficient rendering. Additional performance and memory optimizations improve efficiency for the important use case of rendering patch boundaries. Our new, specialized algorithm is an order of magnitude faster and more memory efficient than the previous implementation.

All of these software enhancements that produce important performance improvements and visualization capabilities crucial to AMR-based computational science projects have been made available to the public through production-quality, parallel-capable, open source visualization software.

Conclusion

This article has but scratched the surface of a number of serious challenges facing modern scientific researchers. At the root of most of these challenges is the fact that we are awash with information, and that gaining understanding from an increasing amount of data is an incredibly challenging task with few, if any, "off-the-shelf" solutions. This article has provided an overview of the value of visualization in scientific knowledge discovery, as well as a couple of examples of current state-of-the-art.

The mission of DOE's SciDAC Visualization and Analytics Center for Enabling Technologies is to gain traction on solutions to this large family of difficult challenges. We use a multi-faceted approach where state-of-the-art technologies from visualization, data analysis, data management, visual interfaces, software architecture and engineering are brought to bear on some of the world's most challenging scientific data understanding problems.

For more information about VACET, please visit our website at www.vacet.org.

Acknowledgment

This work was supported by the Director, Office of Science, Advanced Scientific Computing Research, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231 as part of the DOE Scientific Discovery through Advanced Computing program.

Emerging Visualization Technologies for Ultra-Scale Simulations

Kwan-Liu Ma
University of California at Davis

Supercomputers give scientists the power to model highly complex and detailed physical phenomena and chemical processes, leading to many advances in science and engineering. With the current growth rates of supercomputing speed and capacity, scientists are anticipated to study many problems of unprecedented complexity and fidelity and attempt to study many new problems for the first time. The size and complexity of the data produced by such ultra-scale simulations, however, present tremendous challenges to the subsequent data visualization and analysis tasks, creating a growing gap between scientists' ability to simulate complex physics at high resolution and their ability to extract knowledge from the resulting massive data sets. The Institute for Ultrascale Visualization,[1] [2] funded by the U.S. Department of Energy's SciDAC program,[3] aims to close this gap by developing advanced visualization technologies that enable knowledge discovery at the peta and exa-scale. This article reveals three such enabling technologies that are critical to the future success of scientific supercomputing and discovery.

Parallel Visualization

Parallel visualization can be a useful path to understanding data at the ultra scale, but is not without its own challenges, especially across our diverse scientific user community. The Ultravis Institute has brought together leading experts from visualization, high-performance computing, and science application areas to make parallel visualization technology a commodity for SciDAC scientists and the broader community. One distinct effort is the development of scalable parallel visualization methods for understanding vector field data. Vector field visualization is more difficult to do than scalar field visualization because it generally requires more computing for conveying the directional information and more storage space to store the vector field.

So far, more researchers have worked on the visualization of scalar field data than vector field data, regardless of the fact that vector fields in the same data sets are equally critical to the understanding of the modeled phenomena. 3D vector field visualization particularly requires more attention from the research community because most of the effective 2D vector field visualization methods incur visual clutter when directly applied to depicting 3D vector data. For large data sets, a scalable parallel visualization solution for depicting a vector field is needed even more because the expanded space requirement and additional calculations needed to ensure temporal coherence for visualizing time-varying vector data. Furthermore, it is challenging to simultaneously visualize both scalar and vector fields due to the added complexity of rendering calculations and combined computing requirements. As a result, previous works in vector field visualization primarily focused on 2D, steady flow fields, the associated seed/glyph placement problem, or the topological aspect of the vector fields.

Particle tracing is fundamental to portraying the structure and direction of a vector flow field. When an appropriate set of seed points are used, we can construct paths and surfaces from the traced particles to effectively characterize the flow field. Visualizing a large time-varying vector field on a parallel computer using particle tracing presents some unique challenges. Even though the tracing of each individual particle is independent of other particles, a particle may drift to anywhere in the spatial domain over time, demanding interprocessor communication. Furthermore, as particles move

[1] Institute for Ultrascale Visualization, DOE SciDAC - http://ultravis.org/.

[2] Ma, K. –L., Ross, R., Huang, J., Humphreys, G., Max, N., Moreland, K., Owens, J. D., Shen, H.-W. "Ultra-scale visualization: research and education," *Journal of Physics*, Vol. 78. (also Proceedings of SciDAC 2007 Conference, 24-28 June, 2007, Boston, Massachusetts)

[3] Scientific Discovery through Advanced Computing, Office of Science, Department of Energy - http://www.scidac.gov/.

around, the number of particles each processor must handle varies, leading to uneven workloads. We have developed a scalable, parallel particle tracing algorithm allowing us to visualize large time-varying 3D vector fields at the desired resolution and precision.[4] Figure 1 shows visualization of a velocity field superimposed with volume rendering of a scalar field from a supernova simulation.

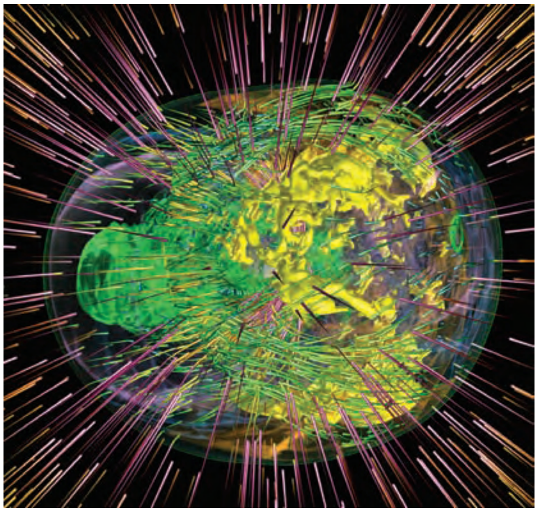


Figure 1. Simultaneous visualization of velocity and angular momentum fields obtained from a supernova simulation.

We take a high-dimensional approach by treating time as the fourth dimension, rather than considering space and time as separate entities. In this way, a 4D volume is used to represent a time-varying 3D vector field. This unified representation enables us to make a time-accurate depiction of the flow field. More importantly, it allows us to construct pathlines by simply tracing streamlines in the 4D space. To support adaptive visualization of the data, we cluster the 4D space in a hierarchical manner. The resulting hierarchy can be used to allow visualization of the data at different levels of abstraction and interactivity. This hierarchy also facilitates data partitioning for efficient parallel pathline construction. We have achieved excellent parallel efficiency using up to 256 processors for the visualization of large flow fields.[4] This new capability enables scientists to see their vector field data in unprecedented detail, at varying abstraction levels, and with higher interactivity, as shown in Figure 2.

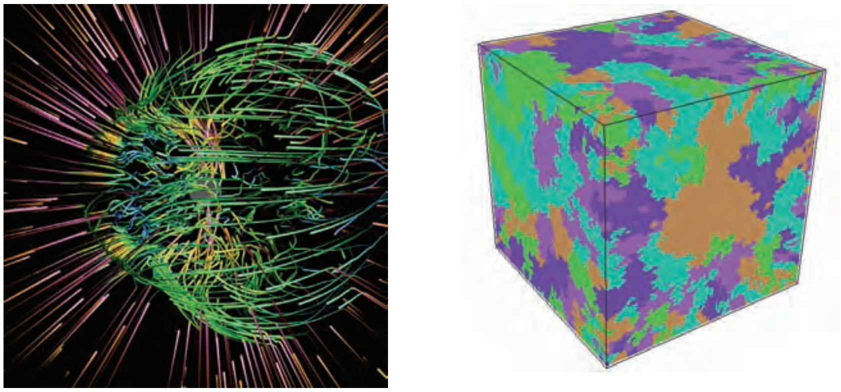


Figure 2. Pathline visualization of velocity field from a supernova simulation and the corresponding vector field partitioning.

[4] Yu, H., Wang, C., Ma, K.-L. "Parallel Hierarchical Visualization of Large 3D Time-Varying Vector Fields," in *Proceedings of the ACM/IEEE Supercomputing 2007 Conference (SC '07)*.

Visualization Interfaces

Over the past 20 years, many novel visualization techniques have been invented but few have been deployed in production systems and tools. Even though some of techniques are made available in a few open-source visualization tools, scientists seem to prefer the more rudimentary tools they have been using. There are several reasons for this. First, scientists are reluctant to switch to a new tool unless the tool can seamlessly fit in their existing computing and analysis environment. Second, although the new technique may produce highly desired visualizations, it will not be widely employed if it requires a tedious process and special hardware to operate. Third and most importantly, for scientists to adopt a new tool, the tool must be very easy and intuitive to use. The past effort in the visualization research community largely focused on improving the performance and quality of visualization calculations. Only over the last few years have the design and deployment of appropriate user interfaces for advanced visualization techniques began to receive more attention.[5] [6]

Interface design has played a major role in several of our visualization projects. One such visualization interface designed for exploring time-varying, multivariate volume data consists of three components, which abstract the complexity of exploring in different spaces of the data and visualization parameters.[7] One important concept realized here is that the interface is also the visualization itself. As shown in Figure 3, the right-most panel displays the time histograms of the data. A time histogram shows how the distribution of data values changes over the whole time sequence and can thus help the user to identify time steps of interest and to specify time-varying features. The middle panel attempts to display the potential correlation between each pair of variables in parallel coordinates for a selected time step. By examining different pairs of variables, the user can often identify features of interest based on the correlations observed. The left-most panel displays the hardware accelerated volume rendering enhanced with the capability to render multiple variables into a single visualization in a user controllable fashion. Such simultaneous visualization of multiple scalar quantities allows users to more closely explore and validate their simulations from the parallel-coordinate space to the 3D physical space. These three components are tightly cross linked to facilitate tri-space data exploration, offering scientists new power to study their time-varying volume data.

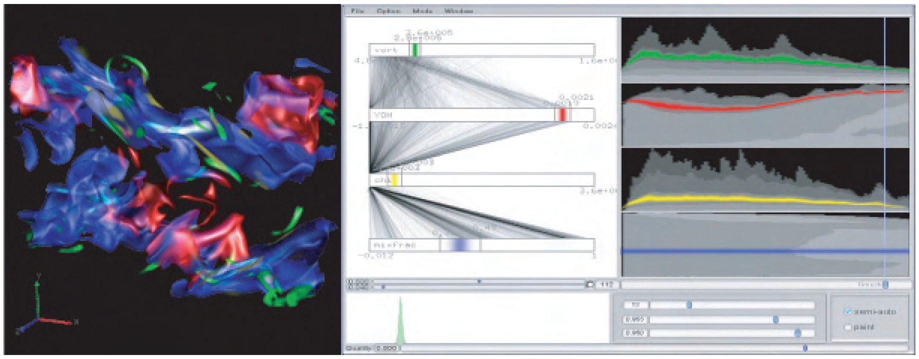


Figure 3. Interface for tri-space visual exploration of time-varying multivariate volume data.[7] From left to right, the spatial view, variable view, and temporal view of the data are given.

The other interface design effectively facilitates visualization of multidimensional particle data output from a gyrokinetic simulation.[8] Depicting the complex phenomena associated with the particle data presents a challenge due to the large quantity of particles, variables, and time steps. By utilizing two modes of interaction–physical space and variable space–our system allows scientists to explore collections of densely packed particles and discover interesting features within the data. While single vari-

[5] Ma, K.-L. "Visualizing Visualizations: User Interfaces for Managing and Exploring Scientific Visualization Data," *IEEE Computer Graphics and Applications*, Vol. 20, Number 5, 2000, pp. 16-19.

[6] K.-L. Ma. "Machine Learning to Boost the Next Generation of Visualization Technology," *IEEE Computer Graphics and Applications*, Volume 27, Number 5, 2007, pp. 6-9.

[7] Akiba, H., Ma, K.-L. "A Tri-Space Visualization Interface for Analyzing Time-Varying Multivariate Volume Data," In *Proceedings of Eurographics/IEEE VGTC Symposium on Visualization*, 2007, pp. 115-122.

[8] Jones, C., Ma, K.-L., Sanderson, A., Myers Jr., L. R. "Visual Interrogation of Gyrokinetic Particle Simulation," *Journal of Physics*, Vol. 78. (also Proceedings of SciDAC 2007 Conference, 24-28 June, 2007, Boston, Massachusetts)

ables can be easily explored through the use of a one dimensional transfer function, we again turn to the information visualization approach of parallel coordinates for interactively selecting particles in multivariate space. In this manner, particles with deeper connections can be separated from the rest of the data and then rendered using sphere glyphs and pathlines, as shown in Figure 4. With this system, scientists at Princeton Plasma Physics Laboratory are able to more easily identify features of interest, such as the location and motion of particles that become trapped in turbulent plasma flow. The combination of scientific and information visualization techniques extend our ability to analyze complex collections of particles.

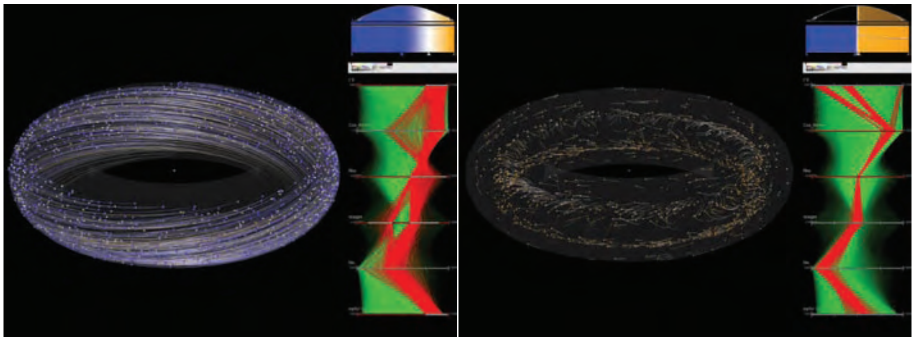


Figure 4. A parallel coordinate interface for multidimensional particle data visualization. The six axes of the parallel coordinates, from top to bottom, are: toroidal coordinate, trap particle condition, parallel velocity, statistical weight, perpendicular velocity, and distance from the center. Left: Visualization of those particles in a layer far from the center, with high parallel velocity and non-zero statistical weight. Right: Visualization of those particles changing direction frequently. This is achieved by restricting the parallel velocity values in a small range.

In addition, we have been studying how to incorporate machine learning into the process of visualization, leading to an intelligent interface for data visualization. Intelligent interfaces are anticipated to replace the current clutter of hardware-specific and algorithm-specific controls with a simple and intuitive interface supported by an invisible layer of complex intelligent algorithms.[6] Only high-level, goal-oriented decisions need to be made by the user, making cutting-edge visualization technology directly accessible to a wide range of application scientists. To make intelligent interfaces widely employed, we need to evaluate the effectiveness of the resulting interface designs using a variety of applications. These studies will pave the way to the creation of next-generation visualization technology. We believe the next generation visualization technology will be built upon further exploitation of human perception to simplify visualization, advanced hardware features to accelerate visualization calculations, and machine learning to reduce the complexity, size, and high-dimensionality of data.

In-Situ Visualization

Due to the size of data output by a large-scale simulation, visualization is almost exclusively done as a post-processing step. Even though it is desirable to monitor and validate some of the simulation stages, the cost of moving the simulation output to a visualization machine could be too high to make interactive visualization feasible. A better approach is not to move the data, or to keep the data that must be moved to a minimum. That is, both simulation and visualization calculations run on the same parallel supercomputer so the data can be shared, as shown in Figure 5. Such in-situ processing can render images directly or extract features, which are much smaller than the full raw data, to store for on-the-fly or later examination. As a result, reducing both the data transfer and storage costs early in the data analysis pipeline can optimize the overall scientific discovery process.

In practice, however, this approach has been sparsely adopted because for two reasons. First, most scientists have been reluctant to use their supercomputer time for visualization calculations. Second, it could take a significant effort to couple a legacy parallel simulation code with an in-situ visualization code. In particular, the domain decomposition optimized for the simulation is often unsuitable for parallel visualization, resulting in the need to replicate data for speeding up the visualization calculations. Hence, the common practice for scientists has been to store only a small fraction of the data or to study the stored data at a coarser resolution, which defeats the original purpose of performing the high-resolution simulations. To enable scientists to study the full extent of the data generated by their simulations and for us to possibly realize the concept of steering simulations at extreme-scale, we should begin investigating the option of in-situ processing and visualization. Many scientists become convinced that simulation-time feature extraction, in particular, is a feasible solution to their large data problem. An important fact is that during the simulation time, all relevant data about the simulated field are readily available for the extraction calculations.

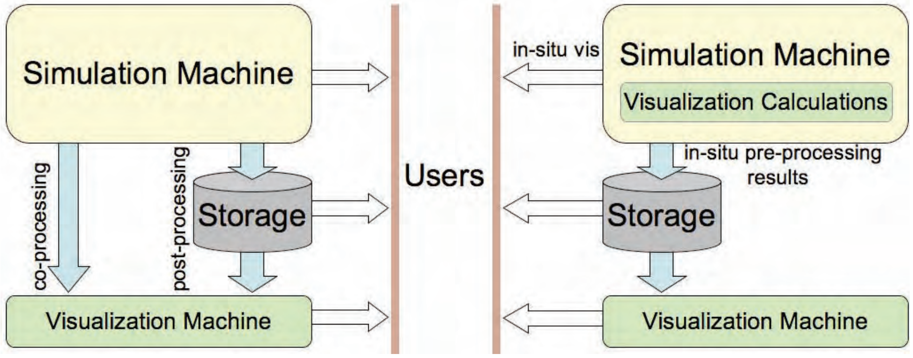


Figure 5. Left: the conventional ways to visualize a large-scale simulation running on a supercomputer. Right: In-situ processing and visualization of large-scale simulations.

In many cases, it is also desirable and feasible to render the data in-situ for monitoring and steering a simulation. Even in the case that runtime monitoring is not practical due to the length of the simulation run or the nature of the calculations, it could still be desirable to generate an animation characterizing selected parts of the simulation. This in-situ visualization capability is especially helpful when a significant amount of the data is to be discarded. Along with restart files, the animations could capture the integrity of the simulation with respect to a particularly important aspect of the modeled phenomenon.

We have been studying in-situ processing and visualization for selected applications to understand the impact of this new approach on ultra-scale simulations, subsequent visualization tasks, and how scientists do their work. Compared with a traditional visualization task that is performed in a post-processing fashion, in-situ visualization brings some unique challenges. First of all, the visualization code must interact directly with the simulation code, which requires both the scientist and the visualization specialist to commit to this integration effort. To optimize memory usage, we have to find a way for the simulation and visualization codes to share the same data structures to avoid replicating data. Second, visualization workload balancing is more difficult to achieve since the visualization has to comply with the simulation architecture and be tightly coupled with it. Unlike parallelizing visualization algorithms for standalone processing where we can partition and distribute data best suited for the visualization calculations, for in-situ visualization, the simulation code dictates data partitioning and distribution. Moving data frequently among processors is not an option for visualization processing. We need to rethink this to possibly balance the visualization workload so the visualization is at least as scalable as the simulation. Finally, visualization calculations must be low cost, with decoupled I/O for delivering the rendering results while the simulation is running. Since the visualization calculations on the supercomputer cannot be hardware accelerated, we must find other ways to simplify the calculations such that adding visualization would take away only a very small fraction of the supercomputer time allocated to the scientist.

We have realized in-situ visualization for a terascale earthquake simulation.[9] This work also won the HPC Analytics Challenges at the SC 2006 Conference[10] because of the scalability and interactive volume visualization we demonstrated. Over a wide-area network, we were able to interactively change view angles, adjust sampling steps, edit the color and opacity transfer function, and zoom in and out for visually monitoring the simulation running on 2048 processors of a supercomputer at the Pittsburgh Supercomputing Center. We were able to achieve high parallel efficiency exactly because we made the visualization calculations, i.e., direct volume rendering, to use the data structures used by simulation code, which removes the need to reorganize the simulation output and replicate data. Rendering is done in-situ using the same data partitioning made by the simulation, and thus no data movement is needed among processors. Similar to the traditional parallel volume rendering algorithms, our parallel in-situ rendering pipeline consists of two stages: parallel rendering and parallel image compositing. In the rendering stage, each processor renders its local data using software ray-casting. Note that this stage may not be balanced given a set of visualization parameters and the transfer function used. In the image compositing stage, a new algorithm is designed to build a communication schedule in parallel on the fly. The basic idea is to balance the overall visualization workload by carefully distributing the compositing calculations. This is possible because parallel image compositing uses only the data generated by the rendering stage and is thus completely independent of the simulation.

For implementation of in-situ visualization, no significant change is needed for the earthquake simulation code for the integration. The only requirement for the simulation is to provide APIs for the access of the simulation internal data structure, which does not require much effort in practice. Furthermore, because all the access is a read operation, the simulation context is not affected by the visualization calculations. The advantage of our approach is obvious. Scientists do not need to change their code to incorporate in-situ visualization. They only need to provide an interface for the visualization code to access their data, as everything else is taken care of by the visualization part. This approach is certainly the most acceptable by scientists.

Conclusion

We are not too far from peta- and exa-scale computing. Will we have the adequate tools for possibly extracting meaning from the data sets generated by such extreme-scale simulations? The investment made by the DOE SciDAC program in ultra-scale visualization[2] is timely and ensures that challenges will be addressed. In this article, we point out the grand challenges facing extreme-scale data analysis and visualization, and present several key technologies for gaining insights in ultra-scale simulations. While we have had some success in deploying some of these technologies, further research and experimental studies are still needed to make these new technologies benefit the scientific supercomputing community at large.

[9] Tu, T., Yu, H., Ramirez-Guzman, L., Bielak, J., Ghattas, O., Ma, K.-L., O'Hallaron, D. R. "From mesh generation to scientific visualization: an end-to-end approach to parallel supercomputing," in *Proceedings of ACM/IEEE Supercomputing 2006 Conference (SC '06).*

[10] Yu, H., Tu, T., Bielak, J., Ghattas, O., Lopez, J. C., Ma, K.-L. O'Hallaron, D. R. Ramirezguzman, L., Stone, N., Taborda-Rios, R., Urbanic, J. "Remote runtime steering of integrated terascale simulation and visualization," *HPC Analytics Challenge*, ACM/IEEE Supercomputing 2006 Conference (SC '06).

Acknowledgments
This work is supported in part by the DOE SciDAC program and NSF ITR program. The images displayed in this article were made by members of the Ultravis Institute and the VIDI research group at University of California at Davis. The supernova data set was provided by Dr. John Blondin at North Carolina State University. The turbulent combustion data set was provided by Dr. Jackie Chen at Sandia National Laboratory.

End-to-End Data Solutions for Distributed Petascale Science

Jennifer M. Schopf
University of Chicago
Argonne National Laboratory

Ann Chervenak
University of Southern California

Ian Foster
University of Chicago
Argonne National Laboratory

Dan Fraser
University of Chicago
Argonne National Laboratory

Dan Gunter
Lawrence Berkeley National Laboratory

Nick LeRoy
University of Wisconsin

Brian Tierney
Lawrence Berkeley National Laboratory

1. Petascale Science is an End-to-end Problem

Petascale science is an end-to-end endeavor, involving not only the creation of massive datasets at supercomputers or experimental facilities, but the subsequent analysis of that data by a user community that may be distributed across many laboratories and universities. The new Center for Enabling Distributed Petascale Science (CEDPS), supported by the US Department of Energy's Scientific Discovery through Advanced Computing (SciDAC) program, is developing tools to support this end-to-end process. In this brief article, we summarize the goals of the project and its progress to date. Some material is adapted from a longer article that appeared in the 2007 SciDAC conference proceedings.[1]

At a recent workshop on computational science, the chair noted in his introductory remarks that if the speed of airplanes had increased by the same factor as computers over the last 50 years, namely five orders of magnitude, then we would be able to cross the US in less than a second. This analogy communicates with great effectiveness the remarkable impact of continued exponential growth in computational performance, which along with comparable improvements in solution methods is arguably the foundation for SciDAC.

However, a participant was heard to exclaim following these remarks: "*yes—but it would still take two hours to get downtown!*" The serious point that this speaker was making is that science is an end-to-end problem and that accelerating just one single aspect of the problem solving process can inevitably achieve only limited returns in terms of increased scientific productivity.

These concerns become particularly important as we enter the era of petascale science, by which we mean science involving numerical simulations performed on supercomputers capable of a petaflop/sec or higher performance, and/or experimental apparatus—such as the Large Hadron Collider,[2] light sources and other user facilities,[3] and ITER[4] —capable of producing petabytes of data. Successful science using such devices demands not only that we be able to construct and operate the simulation or experiment, but also that a distributed community of participants be able to access, analyze, and ultimately make sense of the resulting massive datasets. In the absence of appropriate solutions to the end-to-end problem, the utility of these unique apparatus can be severely compromised.

The following example illustrates issues that can arise in such contexts. A team at the University of Chicago recently used the FLASH3 code to perform the world's largest compressible, homogeneous isotropic turbulence simulation.[5] Using 11 million CPU-hours on the LLNL BG/L computer over a period of a week, they produced a total of 154 terabytes of data, contained in 75 million files that were subsequently archived. Subsequently, they used GridFTP to move 23 terabytes of this data to computers at the University of Chicago; using four parallel streams, this took some three weeks at around 20 megabyte/sec. Next, they spent considerable time using local resources to tag the data, analyze it, and visualize it, augmenting the metadata as well. In a final step, they are making this unique dataset available for use by the community of turbulence researchers by providing analysis services so that other researchers can securely download portions of the data for their own use. In each of these steps, they were

[1] Baranovski, A., et al. "Enabling Distributed Petascale Science," *Journal of Physics: Conference Series*, 78. 2007.

[2] LHC - The Large Hadron Collider Project - http://lhc.web.cern.ch/, 2007.

[3] BES Scientific User Facilities, http://www.sc.doe.gov/bes/BESfacilities.htm, 2007.

[4] ITER - http://www.iter.org/, 2006.

[5] Fisher, R.T., et al. "Terascale Turbulence Computation on BG/L Using the FLASH3 Code," *IBM Systems Journal*. 2007.

ultimately successful—but they would be the first to argue that the effort required to achieve their end-to-end goals of scientific publications and publicly available datasets was excessive.

As this example illustrates, a complete solution to the end-to-end problem may require not only methods for parallel petascale simulation and high-performance parallel I/O (both handled by the FLASH3 code and associated parallel libraries), but also efficient and reliable methods for:

- high-speed reliable *data placement*, to transfer data from its site of creation to other locations for subsequent analysis;
- terascale or faster *local data analysis*, to enable exploration of data that has been fetched locally;
- high-performance *visualization*, to enable perusal of selected subsets and features of large datasets data prior to download;
- *troubleshooting* the complex end-to-end system, which due to its myriad hardware and software components can fail in a wide range of often hard-to-diagnose ways;
- building and operating *scalable services*,[6] so that many users can request analyses of data without having to download large subsets [this aspect of the project is not addressed in this article];
- *securing* the end-to-end system, in a manner that prevents (and/or can detect) intrusions and other attacks, without preventing the high-performance data movement and collaborative access that is essential to petascale science; and
- *orchestrating* these various activities, so that they can be performed routinely and repeatedly.

Each of these requirements can be a significant challenge when working at the petascale level. Thus, a new SciDAC Center for Enabling Technology, the Center for Enabling Distributed Petascale Science (CEDPS) was recently established to support the work of any SciDAC program that involves the creation, movement, and/or analysis of large amounts of data, with a focus on data placement, scalable services, and troubleshooting.

2. Current Data Placement Approaches

Large quantities of data must frequently be moved among computers in a petascale computing environment, whether because there are insufficient resources to perform analysis on the platform that generated the data, because analysis requires specialized resources or involves comparison with other data, or because the data must be published, that is, moved and augmented with metadata, to facilitate use by the community.

Our data placement work addresses three classes of application requirements. First, *staging to and from* active computations and workflows requires placement of data at advantageous locations. By using a data placement service to perform staging operations asynchronously with respect to a workflow or execution engine, rather than explicitly staging data at run time, we hope to demonstrate improved application performance, as suggested in simulations[7] and initial measurements of workflow execution.[8] A current example of where these methods can be applied is the visualization of the results of a combustion simulation at NERSC, which produces 100 TB of data. Smarter placement of the data during simulation execution will enable better use of the visualization component and let scientists understand the resulting data in a more timely fashion.

[6] Foster, I. "Service-Oriented Science," *Science*, 308. 814-817. 2005.

[7] Ranganathan, K. and Foster, I. "Simulation Studies of Computation and Data Scheduling Algorithms for Data Grids," *Journal of Grid Computing*, 1 (1). 2003.

[8] Chervenak, A., et al. "Data Placement for Scientific Applications in Distributed Environments," *8th IEEE/ACM International Conference on Grid Computing (Grid 2007)*, Austin, TX, 2007.

Second, *archival storage* is often the final location of data products that are staged out of a running application, and better data placement services can make archiving operations more efficient. When an application runs on a compute resource such as a cluster or supercomputer, data products must often be staged off the storage system associated with that computational resource onto more permanent secondary or archival storage. These staging out operations can limit application performance, particularly if the compute resource is storage-limited; using an asynchronous data placement service to stage out data products should improve performance. For example, the team running the CCSM climate simulation code at ORNL wants to publish its output data to the Earth System Grid (ESG).[9] They must both transfer the output data to an HPSS archive at NERSC (perhaps while the model is running) and also register each file in a metadata catalog for ESG.

Finally, we are interested in data placement services that maintain required levels of *redundancy* in a distributed environment. For example, it might be the policy of the data placement service to ensure that there are always three copies of every data item stored in the system. If the number of replicas of any data item falls below this threshold, the placement service is responsible for creating additional replicas to meet this requirement. An example of where this requirement arises in practice is the data produced by the CMS experiment at the LHC (at a sustained rate of 400 MB/s), which must be delivered to a Tier 1 site in the US for further processing and then distributed among several US domestic and 20 non-US Tier-2 sites.

Such scenarios, for which we can give many other examples across a wide range of applications, can involve many of the following six elements:

1. Data *registration* and metadata *tagging* as well as data movement;
2. *Bulk data transfer* over high-speed long-haul networks from different sources and sinks;
3. *Coordinated data movement* across multiple sources, destinations, and intermediate locations, including parallel file systems, virtual disks, and hierarchical storage, and among multiple users and applications;
4. *Failure reduction* techniques, such as storage reservation and data replication;
5. *Failure detection* techniques including online monitoring and operation retry to detect and recover from multiple failure modalities; and
6. A need for *predictability* and coordinated scheduling in spite of variations in load and competing use of storage space, bandwidth to the storage system, and network bandwidth.

To summarize the motivation for CEDPS in a sentence: *not only must we be able to transfer data and manage end-point storage systems and resource managers; we must also be able to support the coordinated orchestration of data across many community resources*.

Currently available tools address portions of this functionality. Basic high-performance data transfer (2) is supported by GridFTP,[10] which provides fast performance through parallelism and stripping between data sources. The Replica Location Service and associated Globus data services[11] can provide basic ways to look up where a replica is stored, but metadata tagging (1) is generally an application-specific tool. The NeST[12] and dCache[13] storage management services provide disk-side support for data placement and some of the reliability and error prevention required (4), but not the broader coordinated data movement (3) needed by today's applications. Failure detection (5) and performance prediction (6) are considered open areas of research by many. In general, these requirements go well beyond our current data transfer and

[9] Bernholdt, D., et al. "The Earth System Grid: Supporting the Next Generation of Climate Modeling Research," *Proceedings of the IEEE*, 93 (3). 485-495. 2005.

[10] Allcock, B., Bresnahan, J., Kettimuthu, R., Link, M., Dumitrescu, C., Raicu, I. and Foster, I. "The Globus Striped GridFTP Framework and Server," *SC'2005*, 2005.

[11] Chervenak, A., et al. "Giggle: A Framework for Constructing Scalable Replica Location Services," *SC'02: High Performance Networking and Computing*, http://www.globus.org/research/papers.html#giggle, 2002.

[12] Bent, J., et al. "NeST: A Grid Enabled Storage Appliance," *Grid Resource Management: State of the Art and Future Trends*, 2004.

[13] Shoshani, A., Sim, A. and Gu, J. "Storage Resource Managers: Essential Components for the Grid," In Nabrzyski, J., Schopf, J.M. and Weglarz, J. eds. *Grid Resource Management: State of the Art and Future Trends*, Kluwer Academic Publishers, 2003.

storage resource management capabilities. We will discuss the ways in which our new technology addresses these six elements in the following sections.

3. The CEDPS Managed Object Placement Service: MOPS

We are creating a new class of *data placement services* that can position data reliably across diverse systems and coordinate provisioning, movement, and registration across multiple storage systems to enable efficient and prioritized access by many users. A single, logical transfer may involve multiple sources and destinations necessitating the use of intermediate store and forward storage systems, or the creation of optimized overlay networks such as user level multicast networks. Concurrent independent placement operations may be prioritized and monitored in case of failures.

As a first step, we have recently released a prototype Managed Object Placement Service (MOPS), shown in Figure 1, which transforms storage into a managed resource. MOPS allows users to negotiate access to a certain quantity of storage for a certain time and with defined performance characteristics. Its design and implementation leverages GridFTP, NeST, and dCache.

GridFTP provides a flexible core architecture with a data interface component that allows different plug-ins for added functionality. It is well known for its high-speed data transfer capabilities. GridFTP gives MOPS the core functionality of fast, bulk file transfers, element 2 in our scenarios, which MOPS extends through its plug-in capability

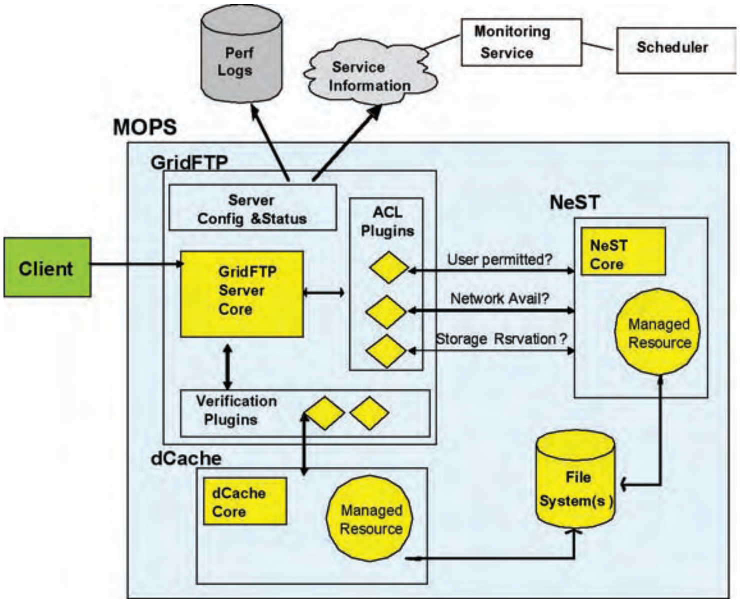


Figure 1. General MOPS architecture.

NeST provides guaranteed storage allocation by allowing the user and storage device to negotiate a size and duration and to specify access control lists (ACLs) for file access. In this way, a system can specify which users can access certain files or sets of files and also work with disk reservations when they are available. This feature helps address element 3, coordinated data movement, and element 4, failure reduction, by decreasing the chance of disk overflow errors.

dCache provides methods for managing backend (tertiary) storage systems including space management, hot spot determination, and recovery from disk or node failures. When connected to a tertiary storage system, dCache simulates unlimited direct access storage space; data exchanges to and from the underlying tertiary storage system are performed automatically and invisibly to the user. Recent CEDPS-funded work has implemented data transfer consistency verification features for verifying that individual transfers have completed correctly. dCache also addresses element 3, coordinated data movement, and element 4, failure reduction.

By combining these three tools with a single user interface using MOPS, CEDPS users can now work with their data in a more managed environment, especially in terms of reducing failures due to running out of disk space in the middle of a transfer, limiting the access to a set of files, or verifying that a transfer has completed successfully, while continuing to serve the data quickly across a wide variety of networks and back-end storage systems.

4. The CEDPS Data Placement Service

CEDPS is also developing the Data Placement Service (DPS) that will perform data transfer operations using MOPS. For data-intensive scientific applications running in a distributed environment, the placement of data onto storage systems can have a significant impact on the performance of scientific computations and on the reliability and availability of data sets. These scientific applications may produce and consume terabytes or petabytes of data stored in millions of files or objects, and they may run complex computational workflows consisting of millions of interdependent tasks. A variety of data placement algorithms could be used, depending on the requirements of a scheduler or workflow management system as well as the data distribution goals of the scientific collaboration, or *Virtual Organization* (VO). For example, a placement algorithm might distribute data in a way that is advantageous for application or workflow execution by placing data sets near high-performance computing resources so that they can be staged into computations efficiently; by moving data off computational resources quickly when computation is complete; and by replicating data sets for performance and reliability. These goals might be considered *policies* of the workflow manager or VO, and a *policy-driven data placement service* is responsible for replicating and distributing data items in conformance with these policies or preferences. A data placement service could also make use of hints from a workflow management system about applications and their access patterns, for example, whether a set of files is likely to be accessed together and therefore should be replicated together on storage systems.

To demonstrate the effectiveness of intelligent data placement, we integrated the Pegasus workflow management system[14] from the USC Information Sciences Institute with the Globus Data Replication Service,[15] which provides efficient replication and registration of data sets. We demonstrated[8] that using hints from the workflow management system allowed us to reduce the execution time of scientific workflows when we were able to successfully prestage necessary data onto appropriate computational resources.

This initial work has led us to design a general, asynchronous Data Placement Service (DPS) that will operate on behalf of a virtual organization and accept data placement requests from clients that reflect, for example, grouping of files, order of file requests, etc. Figure 2 illustrates the operation of a DPS for stage in requests issued by a workflow management system. We also plan to incorporate configurable policies into the data placement service that reflect the data distribution policies of a particular

[14] Deelman, E., et al. "Pegasus: A Framework for Mapping Complex Scientific Workflows onto Distributed Systems," *Scientific Programming, 13* (3). 219-237. 2005.

[15] Chervenak, A., Schuler, R., Kesselman, C., Koranda, S. and Moe, B., "Wide Area Data Replication for Scientific Collaborations," In *6th IEEE/ACM Int'l Workshop on Grid Computing* (2005).

VO. Our goal is to produce a placement service that manages the competing demands of VO data distribution policies, data staging requests from multiple competing workflows, and additional on-demand data requests from other clients.

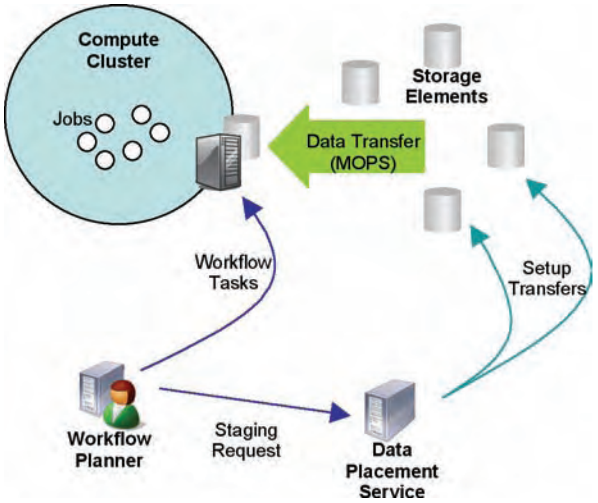


Figure 2. Shows a workflow management system acting as a client of a data placement service and issuing requests for staging of data sets. The DPS issues MOPS data transfers from appropriate storage elements to the compute cluster(s) on which workflow execution will take place.

We have implemented an initial version of the data placement service with a planned software release in October 2007. This implementation modifies and significantly extends the existing Globus Data Replication Service. The implementation uses several Globus components, including the Java WS Core that provides the basic infrastructure for supporting web service deployment and generic operation support such as basic state management, query operations, endpoint references, etc.; the Globus Replica Location Service that provides registration and discovery of of data items; the GridFTP data transfer service for secure and efficient data staging operations; and the Grid Security Infrastructure for secure access to resources in the distributed environment.

5. CEDPS Troubleshooting

Distributed data management involves end-to-end systems comprising of many different hardware and software components in different physical locations and administrative domains. Failures can occur and they can be hard to diagnose. Experience with current DOE distributed system deployments has shown that understanding behavior is a fundamental requirement, not just a desirable enhancement. Middleware may also mask performance faults, when applications produce correct results but experience degradation in performance.

In order to better understand failures and to increase the reliability of the end-to end system, we have developed tools to allow easer access to logs and additional log analysis software that performs anomaly detection. In addition, we have also deployed a higher-level monitoring tool that observes services and generates notifications when errors occur.

Figure 3 shows the CEDPS log management service based on the syslog-ng system.[16] We mine software and service logs (such as those from GridFTP, MOPS, or

[16] The syslog-ng Logging System - http://www.balabit.com/products/syslog-ng/, 2007.

other tools), which are filtered and forwarded to a common location. That combined set of data can then be analyzed. We have used NetLogger[17] to access performance data and discover faulty event chains where expected behavior does not occur. We have also developed prototypes of anomaly detection tools that can detect a missing event in an event stream and also identify unexpected performance variations that indicate an underlying problem that may not cause an out right failure.[18] This system is currently in the process of being deployed on the Open Science Grid (OSG).[19]

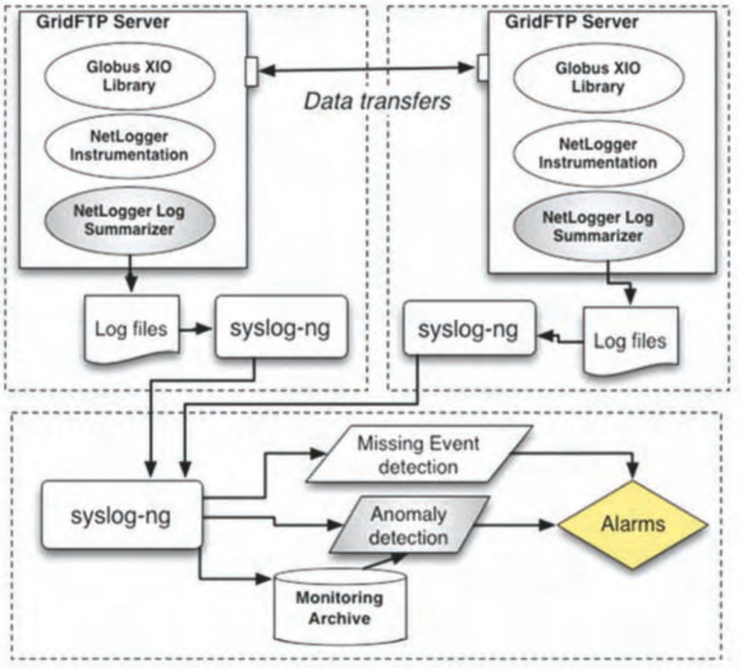


Figure 3. Syslog-ng deployment architecture, and interactions with anomaly detection and alarm tools.

Both of these tools have been aided by effort spent on improving the quality and consistency of available performance information. Specifically, we have codified a set of logging "Best Practices,"[20] and are modifying the Globus Toolkit[21] to follow these practices. In defining these guidelines, we have worked with the European EGEE project to achieve compatibility with their security logging guidelines,[22] an important requirement for LHC computing.

To compliment our log services and to assist further with our scenario elements 5 and 6 (failure reduction and detection), we have also developed a Trigger service[18] that runs small probes and notifies system administrators and end users when certain conditions are met. These can include a service failure or failure to respond to a ping, or a warning condition, such as a nearly full disk, overly long queue, or high load condition on a resource. The Trigger service has been used by ESG for over three years for system failure notifications and to help diagnose errors. We have re-architected this component to allow for additional trigger services, a separation of matching conditions and actions taken upon failure notification, and easier deployment through a Web interface.

These tools combine to give us additional support in the end-to-end data management environment.

[17] Tierney, B. and Gunter, D. *NetLogger: A Toolkit for Distributed System Performance Tuning and Debugging*. Lawrence Berkeley National Laboratory, Technical Report LBNL-51276, 2003.

[18] Chervenak, A., et al. "Monitoring the Earth System Grid with MDS4," *2nd IEEE Intl. Conference on e-Science and Grid Computing (e-Science 2006)*, Amsterdam, Netherlands, 2006.

[19] Pordes, R., et al. "The Open Science Grid," In *Scientific Discovery through Advanced Computing (SciDAC) Conference*, (2007).

[20] Grid Logging Best Practices Guide, CEDPS - http://www.cedps.net/wiki/images/6/6f/CEDPS-troubleshooting-bestPractices-16.doc, 2007.

[21] Foster, I. "Globus Toolkit Version 4: Software for Service-Oriented Systems," *Journal of Computational Science and Technology, 21* (4). 523-530. 2006.

[22] Groep, D. *Middleware Security Audit Logging Guidelines* EGEE Document 2006-11-07, https://edms.cern.ch/document/793208, 2006.

6. Revisiting the FLASH Example

We began this article with a discussion of the University of Chicago FLASH application experiment, in which it took three weeks at 20 MB/s to transfer less than 15% of the data produced in a three-week simulation. By using MOPS, it is possible that smarter disk allocation could have been done, allowing the FLASH group to transfer data of particular interest more quickly and as it was being generated due to smarter handling of the backend storage system. When performing local analysis and replication of the data, the FLASH team could now take advantage of the DPS, which would handle registering new files and distributing them according to the policy defined by the FLASH team, instead of having to do this work by hand. In addition, with the added centralized logging and trigger service deployed at the various sites, FLASH scientists would be able to detect any failures and debug any performance problems much more easily than the current environment. The effort required to achieve their end-to-end goals of scientific publications and publicly available datasets would be significantly reduced overall.

7. Summary

We have introduced the SciDAC Center for Enabling Distributed Petascale Science (CEDPS), which is addressing three problems critical to enabling the distributed management and analysis of petascale datasets: data placement, scalable services, and troubleshooting.

In data placement, we are developing tools and techniques for reliable, high-performance, secure, and policy driven placement of data within a distributed science environment. We are constructing a managed object placement service (MOPS)—a significant enhancement to today's GridFTP—that allows for management of the space, bandwidth, connections, and other resources needed to transfer data to and/or from a storage system. Building on this base, we are developing end-to-end data placement services that implement different data distribution and replication behaviors.

In troubleshooting, we are developing tools for the *detection and diagnosis of failures* in end-to-end data placement and distributed application hosting configurations. We are constructing an end-to-end monitoring architecture that uses instrumented services to provide detailed data for both background collection and run-time, event-driven collection. We are also constructing new monitoring analysis tools able to detect failures and performance anomalies and predict system behaviors using archived data and event logs.

These tools allow scientists to interact more easily with large data sets created during petascale computations, and allow faster end analysis of the data. More details can be found at http://www.cedps.net/.

Acknowledgements

This work is supported through the U.S. Department of Energy Office of Science, Office of Advanced Scientific Computing Research, through the SciDAC program. Work at Argonne is supported under Contract DE-AC02-06CH11357 and at Lawrence Berkeley National Laboratory, under Contract DE-AC02-05CH11231. We gratefully acknowledge the contributions of our fellow CEDPS participants Andrew Baranovski, Shishir Bharathi, John Bresnahan, Tim Freeman, Keith Jackson, Kate Keahey, Carl Kesselman, David E. Konerding, Mike Link, Miron Livny, Neill Miller, Robert Miller, Gene Oleynik, Laura Pearlman, and Robert Schuler.

Scientific Data Management: Essential Technology for Accelerating Scientific Discoveries

Arie Shoshani
Lawrence Berkeley National Laboratory

Ilkay Altintas
San Diego Supercomputer Center

Alok Choudhary
Northwestern University

Terence Critchlow
Pacific Northwest National Laboratory

Chandrika Kamath
Lawrence Livermore National Laboratory

Bertram Ludäscher
University of California, Davis

Jarek Nieplocha
Pacific Northwest National Laboratory

Steve Parker
University of Utah

Rob Ross
Argonne National Laboratory

Nagiza Samatova
Oak Ridge National Laboratory

Mladen Vouk
North Carolina State University

Introduction

Terascale computing and large scientific experiments produce enormous quantities of data that require effective and efficient management. The task of managing scientific data is so overwhelming that scientists spend much of their time managing the data by developing special purpose solutions, rather than using their time effectively for scientific investigation and discovery. Effectively generating, managing, and analyzing this information requires a comprehensive, end-to-end approach to data management that encompasses all of the stages, from the initial data acquisition to the final analysis of the data. Fortunately, the data management problems encountered by most scientific domains are common enough to be addressed through shared technology solutions. Based on community input, we have identified three significant requirements. First, more efficient access to storage systems is needed. In particular, parallel file system improvements are needed to read and write large volumes of data without slowing a simulation, analysis, or visualization engine. These processes are complicated by the fact that scientific data are structured differently for specific application domains, and are stored in specialized file formats. Second, scientists require technologies to facilitate better understanding of their data, in particular the ability to effectively perform complex data analysis and searches over large data sets. Specialized feature discovery and statistical analysis techniques are needed before the data can be understood or visualized. To facilitate efficient access, it is necessary to keep track of the location of the datasets, effectively manage storage resources, and efficiently select subsets of the data. Finally, generating the data, collecting and storing the results, data post-processing, and analysis of results is a tedious, fragmented process. Tools for automation of this process in a robust, tractable, and recoverable fashion are required to enhance scientific exploration.

The Scientific Data Management (SDM) Center,[1] funded under the DOE SciDAC program, focuses on the application of known and emerging data management technologies to scientific applications. The Center's goals are to integrate and deploy software-based solutions to the efficient and effective management of large volumes of data generated by scientific applications. Our purpose is not only to achieve efficient storage and access to the data using specialized indexing, compression, and parallel storage and access technology, but also to enhance the effective use of the scientist's time by eliminating unproductive simulations, by providing specialized data-mining techniques, by streamlining time-consuming tasks, and by automating the scientist's workflows. Our approach is to provide an integrated scientific data management framework where components can be chosen by the scientists and applied to their specific domains. By overcoming the data management bottlenecks and unnecessary information-technology overhead through the use of this integrated framework, scientists are freed to concentrate on their science and achieve new scientific insights.

[1] http://sdmcenter.lbl.gov, contains extensive publication lists, with access to full papers.

The Three-Layer Organization of the SDM Center

As part of our evolutionary technology development and deployment process (from research through prototypes to deployment and infrastructure) we have organized our activities in three layers that abstract the end-to-end data flow described above. We labeled the layers as Storage Efficient Access (SEA), Data Mining and Analytics (DMA), and Scientific Process Automation (SPA). The SEA layer is immediately on top of hardware and operating systems, providing parallel data access to files and transparent access to archival storage. The DMA layer, which builds on the functionality of the SEA layer, consists of indexing, feature selection, and parallel statistical analysis technology. The SPA layer, which is on top of the DMA layer, provides the ability to compose workflows from the components in the DMA layer as well as application specific modules. Figure 1 shows this organization and the components developed by the center and applied to various scientific applications.

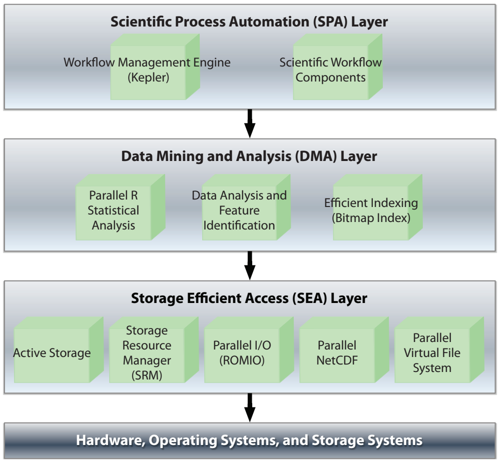

Scientific Process Automation (SPA) Layer
Workflow Management Engine (Kepler)
Scientific Workflow Components
Data Mining and Analysis (DMA) Layer
Parallel R Statistical Analysis
Data Analysis and Feature Identification
Efficient Indexing (Bitmap Index)
Storage Efficient Access (SEA) Layer
Active Storage
Storage Resource Manager (SRM)
Parallel I/O (ROMIO)
Parallel NetCDF
Parallel Virtual File System
Hardware, Operating Systems, and Storage Systems

Figure 1. The three-layer organization of technologies in the SDM Center

Over the last several years, the technologies supported by the SDM center have been deployed for a variety of application domains. Some of the most notable achievements are:

- More than a tenfold speedup in writing and reading netCDF files has been achieved by developing MPI-IO based Parallel netCDF software being utilized by astrophysics, climate, and Parallel VTK.
- An improved version of PVFS is freely available to the community and offered through cluster vendors. In addition to operating on clusters, it is routinely used on the IBM BlueGene/L and soon on the BlueGene/P.
- Methods for the correct classification of orbits in puncture plots and for "blob tracking" from the National Compact Stellarator eXperiment (NCSX) at PPPL were using a combination of image processing, statistics, and pattern recognition techniques.
- A new bitmap indexing method has enabled an efficient search over billions of collisions (events) in High Energy Physics, and is being applied to combustion, astrophysics, and visualization domains. It achieves more than a tenfold speedup in generating regions and tracking them over time.
- The development of a Parallel R, an open source parallel version of the popular statistical package R. This is being applied to climate, GIS, and mass spec proteomics applications.
- A scientific workflow management and execution system (called Kepler) has been developed and deployed within multiple scientific domains, including genomics and astrophysics. The system supports the design and the execution of flexible and reusable, component-oriented workflows.

Descriptions of technologies developed and used in the SDM Center

In this section we describe the SDM Center technologies, and include some examples of their application in various scientific projects. We proceed with technologies from the top layer to the bottom layer.

The Kepler Scientific Workflow System

A practical bottleneck for more effective use of available computational and data resources is often the design of resource access and use of processes, and the corresponding execution environments, i.e., in the scientific workflow environment of end user scientists. The goal of the Kepler system[2] is to provide solutions and products for effective and efficient modeling, design and execution of scientific workflows. Kepler is a multi-site open source effort, co-founded by the SDM center, to extend the Ptolemy system (from UC Berkeley) and create an integrated scientific workflow infrastructure. We have also started to incorporate data, process, system and workflow provenance and run-time tracking and monitoring. We have worked closely with application scientists to design, implement, and deploy workflows that address their real-world needs. In particular, we have active users on the SciDAC Terascale Supernova Initiative (TSI) team and an LLNL Biotechnology project, as well as at the Center for Plasma Edge Simulation (CPES) fusion project. While the Scientific Process Automation (SPA) layer uses Kepler to achieve workflow automation, it is the specific task components (called "actors" in Kepler) developed by the SDM center that makes our work unique in it usefulness to scientific applications.

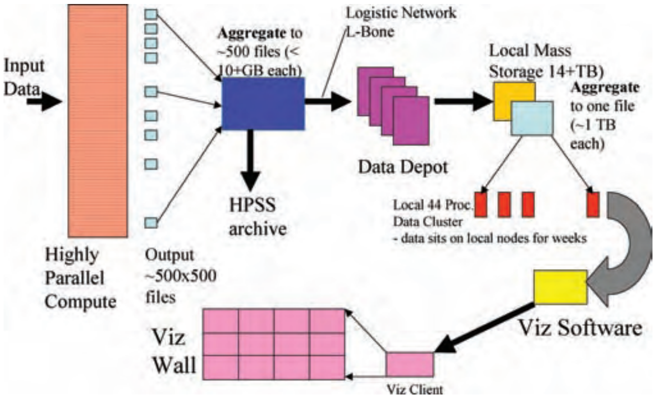

Input Data
Highly Parallel Compute
Output ~500x500 files
Aggregate to ~500 files (< 10+GB each)
HPSS archive
Logistic Network L-Bone
Data Depot
Local Mass Storage 14+TB)
Aggregate to one file (~1 TB each)
Local 44 Proc. Data Cluster - data sits on local nodes for weeks
Viz Software
Viz Wall
Viz Client

Figure 2. An abstract representation of a scientific workflow

Underlying challenges related to simulations, data analysis and data manipulation include scalable parallel numerical algorithms for the solution of large, often sparse linear systems, flow equations, and large Eigen-value problems, running of simulations on supercomputers, movement of large amounts of data over large distances, collaborative visualization and computational steering, and collection of appropriate process and simulation related status and provenance information. This requires interdisciplinary teams of application scientists and computer scientists working together to define the workflows and putting them into the Kepler workflow framework. The general underlying "templates" are often similar across disciplines: large-scale parallel computations and steering (hundreds of processors, gigabytes of memory, hours to weeks of CPU time), data-movement and reduction (terabytes of data), visualization and analytics (interactive, retrospective, and auditable). An abstraction of this and its Kepler translation are illustrated in Figure 2 and 3 for a particular astrophysics project, call the Terascale Supernova Initiative (TSI).[3] Figure 3 shows the capability of the

[2] http://kepler-project.org/

[3] http://www.phy.ornl.gov/tsi/

Kepler system to represent hierarchically structured workflows. In the center of the figure there are four simple high-level tasks; each is expanded into lower level tasks that manage the detailed processes.

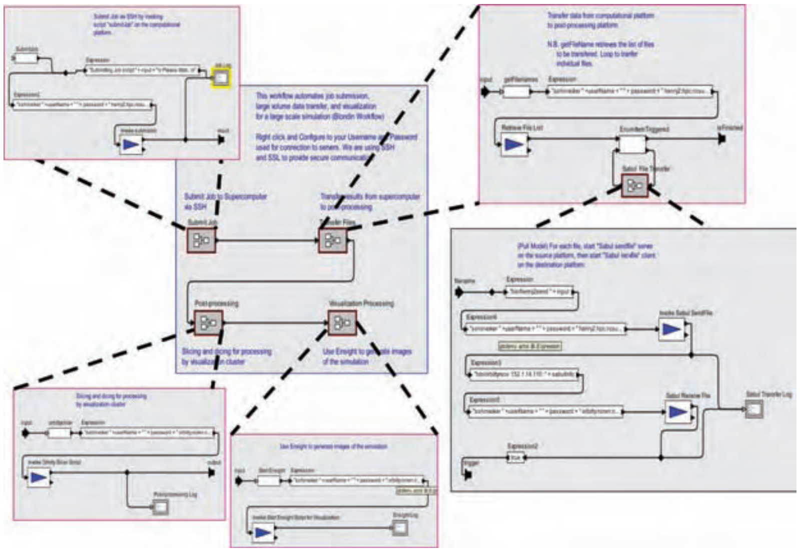

Submit Job as SSH by invoking script "submitJob" on the computational platform.
This workflow automates job submission, large volume data transfer, and visualization for a large scale simulation (Biondin Workflow)
Right click and Configure to your Username and Password used for connection to servers. We are using SSH and SSL to provide secure communication
Submit Job to Supercomputer via SSH
Submit Job
Transfer results from supercomputer to post-processing
Transfer Files
Post-processing
Visualization Processing
Slicing and dicing for processing by visualization cluster
Use Ensight to generate images of the simulation
Transfer data from computational platform to post-processing platform.
N.B. getFileName retrieves the list of files to be transfered. Loop to transfer individual files.
input
getFilenames
Expression
Retrieve File List
EnumberTrigger
isFinished
Sabul File Transfer
(Pull Model) For each file, start "Sabul sendfile" server on the source platform, then start "Sabul recvfile" client on the destination platform.
Expression
Invoke Sabul SendFile
Sabul Retrieve File
Sabul Transfer Log
Slicing and dicing for processing by visualization cluster
input
output
Postprocessing Log
Use Ensight to generate images of the simulation

Figure 3. Instantiation of the abstract workflow in Kepler

Feature Extraction and Tracking

As part of the Data Mining and Analysis (DMA) layer, the SDM center is developing scalable algorithms for the interactive exploration of large, complex, multi-dimensional scientific data. By applying and extending ideas from data mining, image and video processing, statistics, and pattern recognition, we are developing a new generation of computational tools and techniques that are being used to improve the way in which scientists extract useful information from data.[4] These tools were applied to problems in a variety of application areas, including separation of signals in climate data from simulations, the identification of key features in sensor data from the D-III-D Tokamak, and the classification and characterization of orbits in Poincaré plots in Fusion data.

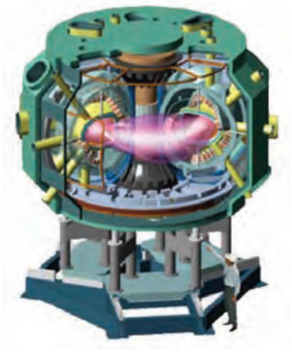


Figure 4. A schematic of the NSTX

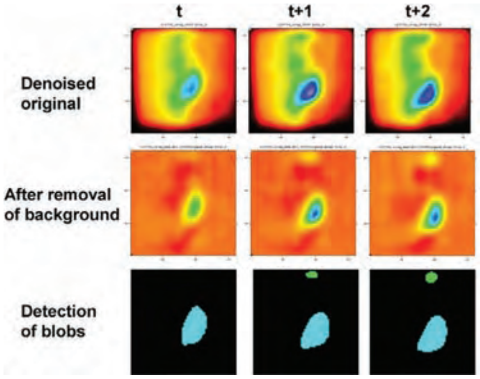


Figure 5. Tracking of "blobs" in Fusion images

A specific example of the effectiveness of such techniques is the identification of the movement of "blobs" in images from fusion experiments, using data from the National Spherical Torus Experiment (NSTX),[5] shown in Figure 4. A blob is a coherent structure in the image that carries heat and energy from the center of the torus to the wall. Figure 5 shows bright blobs extracted from experimental images from the NSTX. The blobs are high energy regions. If they hit the torus wall that confines the

[4] http://www.llnl.gov/casc/sapphire/sapphire_home.html

[5] http://nstx.pppl.gov/

plasma, it can vaporize. The figure shows movement of the blobs over time. A key challenge to the analysis is the lack of a precise definition for these structures. Figure 5 shows three consecutive images from an NSTX sequence. The original images are somewhat noisy and must first be processed to remove the noise. We have applied our background subtraction software to remove the quiescent background intensity in the sequences. Next, ambient background intensity, which is approximated by the median of the sequence, is removed, thus highlighting the blob regions, as shown in the second row of the figure. We then use image processing techniques to identify and track the blobs over time, as shown in the third row. The goal is to validate and refine the theory of plasma turbulence.

Parallel Statistical Analysis

Another area supported by the DMA layer is efficient statistical analysis. Present data analysis tools such as Matlab, IDL, and R, even though highly advanced in providing various statistical analysis capabilities, are not apt to handle large data-sets. Most of the researchers' time is spent on addressing data preparation and management needs of their analyses. Parallel R[6] is an open source parallel statistical analysis package developed by the SDM center that lets scientists employ a wide range of statistical analysis routines on high performance shared and distributed memory architectures without having to deal with the intricacies of parallelizing these routines. Through Parallel R, the user can distribute data and carry out the required parallel computation but maintain the same look-and-feel interface of the R system. Two major levels of parallelism are supported: data parallelism (k-means clustering, Principal Component Analysis, Hierarchical Clustering, Distance matrix, Histogram) and task parallelism (Likelihood Maximization, Bootstrap and Jackknife Re-sampling, Markov Chain Monte Carlo, Animations). Figure 6 shows a schematic of the concepts. ParallelR has been applied in multiple scientific projects including feature extraction for quantitative high-throughput proteomics, parallel analyses of climate data, and in combination with geographical information systems.

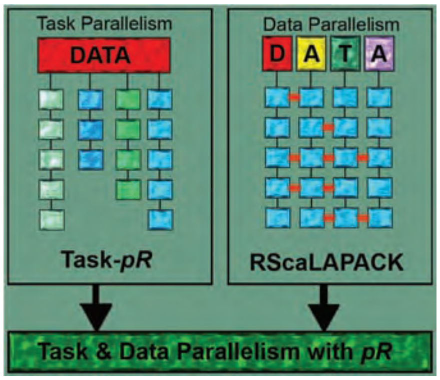


Figure 6. Providing data and task parallelism in ParallelR

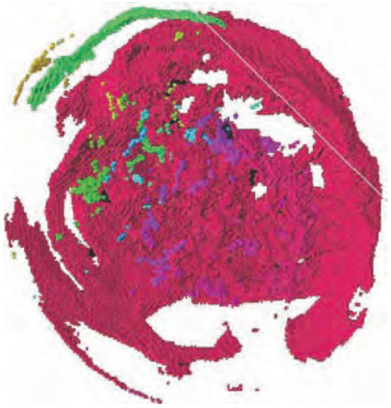
Specialized indexing technology for very large datasets

Another aspect of effective data analysis supported by the DMA technology in the SDM center, is the ability to identify, in real-time, items of interest from billions of data values in large datasets. This is a significant challenge posed by the huge amount of data being produced by many data-intensive science applications. For example, a high-energy physics experiment called STAR is producing hundreds of terabytes of data a year and has accumulated many millions of files in the last five years of operation. One of the core missions of the STAR experiment is to verify the existence of a new state of matter called the Quark Gluon Plasma (QGP). An effective strategy for this

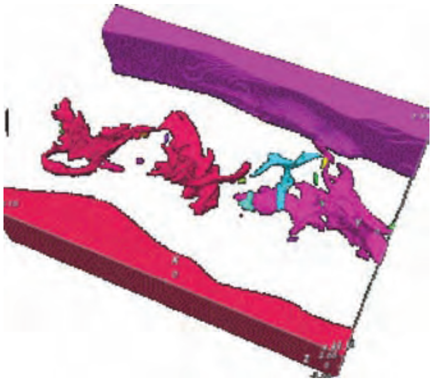
[6] http://cran.r-project.org/doc/packages/RScaLAPACK.pdf

task is to find the high-energy collisions that contain signatures unique to QGP, such as a phenomenon called jet quenching. Among the hundreds of millions of collision events captured, a very small fraction of them (maybe only a few hundreds) contain clear signatures of jet quenching. Efficiently identifying these events and transferring the relevant data files to analysis programs are a great challenge. Many data-intensive science applications are facing similar challenges in searching their data.

Over the last several years, we have been working on a set of strategies to address this type of searching problem. Usually, the data to be searched are read-only. Our approach takes advantage of this fact. We have developed a specialized indexing method based on representing the indexed data as a compressed bitmap. This indexing method, called FastBit,[7] is an extremely efficient bitmap indexing technology. Unlike other bitmap indexes that assume low cardinality of possible data values, FastBit is particularly useful for scientific data, since it is designed for high-cardinality numeric data. FastBit performs 12 times faster than any known compressed bitmap index in answering range queries. Because of its speed, Fastbit facilitates real-time analysis of data, searching over billions of data values in seconds. FastBit has been applied to several application domains, including finding flame fronts in combustion data, searching for rare events from billions of high energy physics collision events, and more recently to facilitate query-based visualization. The examples in Figure 7 (for astrophysics and combustion data) show the use of a tool, called DEX,[8] that used Fastbit in combination with VTK to achieve a very fast selection of features from large datsets and their display in real-time.



Exploding supernova



Methane jet flame

Figure 7. Examples of regions found by Fastbit indexes in real-time from very large datasets

Advanced I/O Infrastructure

As high-performance computing applications scale and move from performing simulation and computing to data analysis they become tremendously data-intensive, creating a potential bottleneck in the entire scientific discovery cycle. At the same time, it is a well-known phenomenon that I/O access rates have not kept pace with high-performance computing performance as a whole. Because of this phenomenon, it becomes increasingly important for us to extract the highest possible performance from the I/O hardware that is available to us. Even if the raw hardware capacity for storage and I/O is available in an infrastructure, the complexity arising from the scale and parallelism is daunting and requires significant advances in software to provide the required performance to applications.

[7] http://sdm.lbl.gov/fastbit/

[8] Stockinger, K., Shalf, J., Bethel, W., Wu, K. "DEX: Increasing the Capability of Scientific Data Analysis Pipelines by Using Efficient Bitmap Indices to Accelerate Scientific Visualization," *International conference on Scientific and Statistical Database Management (SSDBM 2005)*, Santa Barbara, California, USA, June 2005. Available at http://crd.lbl.gov/~kewu/ps/LBNL-57023.pdf

The Storage Efficient Access (SEA) component provides the software infrastructure necessary for efficient use of the I/O hardware by applications. This is accomplished through a sequence of tightly coupled software layers, shown in Figure 8, building on top of I/O hardware at the bottom and providing application-oriented, high-level I/O interfaces at the top. Three APIs are made available for accessing SEA components: Parallel netCDF at the high-level I/O library level, ROMIO at the MPI-IO level, and Parallel Virtual File System (PVFS) at the file level.

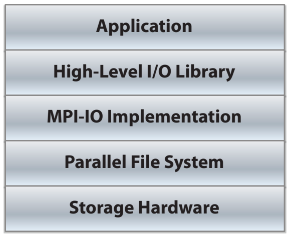


Figure 8. The I/O stack

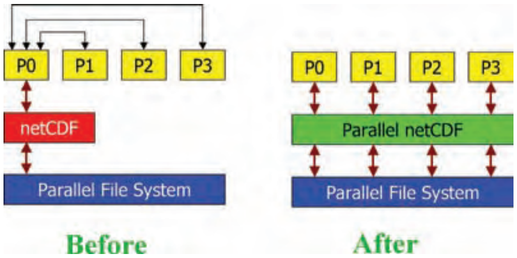


Figure 9. Serialization problems in original netCDF removed in Parallel netCDF to achieve a 10 fold performance increase

PVFS[9] can provide multiple GB/second parallel access rates, and is freely available. Above the parallel file system is software designed to aid applications in more efficiently accessing the parallel file system. Implementations of the MPI-IO interface are arguably the best example of this type of software. MPI-IO provides optimizations that help map complex data movement into efficient parallel file system operations. Our ROMIO[10] MPI-IO interface implementation is freely distributed and is the most popular MPI-IO implementation for both clusters and a wide variety of vendor platforms. MPI-IO is a powerful but low-level interface that operates in terms of basic types, such as floating point numbers, stored at offsets in a file. However, some scientific applications desire more structured formats that map more closely to the application's use, such as multi-dimensional datasets. NetCDF[11] is a widely used API and portable file format that is popular in the climate simulation and data fusion communities. As part of the work in the SDM center, a parallel version of NetCDF (pNetCDF) was developed. It provides a new interface for accessing NetCDF data sets in parallel. This new parallel API closely mimics the original API, but is designed with scalability in mind and is implemented on top of MPI-IO. Performance evaluations using micro-benchmarks as well as application I/O kernels have shown major scalability improvements over previous efforts. Figure 9 shows schematically the concept of adding a parallel netCDF layer to eliminate serialization through a single processor.

Upcoming systems will incorporate hundreds of thousands of compute processors along with support nodes. Using POSIX and MPI-IO interfaces, I/O operations will be forwarded through a set of I/O nodes to storage targets. Work is underway to develop efficient forwarding systems to match petascale architectures and to best connect to underlying file systems, including PVFS.

[9] http://www.parl.clemson.edu/pvfs2

[10] http://www.mcs.anl.gov/romio

[11] http://www.mcs.anl.gov/parallel-netcdf

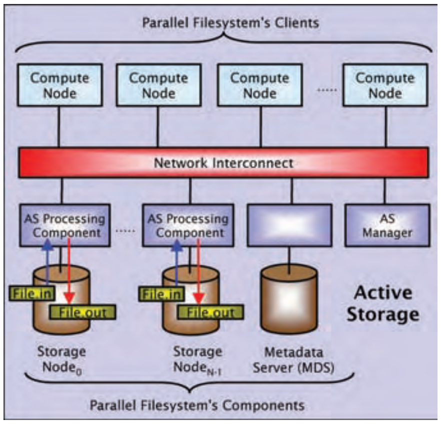


Figure 10. The Active Storage architecture

Active Storage

Despite recent advancements in storage technologies for many data intensive applications, analysis of data remains a serious bottleneck. In traditional cluster systems, I/O-intensive tasks must be performed in the compute nodes. This produces a high volume of network traffic. One option for data analysis is to leverage resources not on the client side, but on the storage side referred to as Active Storage. The original research efforts on active storage were based on a premise that modern storage architectures might include usable processing resources at the storage controller or disk; unfortunately, commodity storage has not yet reached this point. However, parallel file systems offer a similar opportunity. Because the servers used in parallel file systems often include commodity processors similar to the ones used in compute nodes, many Giga-op/s of aggregate processing power are often available in the parallel file system. As part of the SEA layer technology, our goal in the Active Storage project is to leverage these resources for data processing. Scientific applications that rely on out-of-core computation are likely candidates for application of this technique, because their data is already being moved through the file system. The Active Storage approach allows moving computations involving data stored in a parallel file system from the compute nodes to the storage nodes. Benefits of Active Storage include low network traffic, local I/O operations, and better overall performance. The SDM center has implemented Active Storage on Lustre and PVFS parallel file systems. We plan to pursue deployment of Active Storage in biology or climate application. 🌍

The Earth System Grid Center for Enabling Technologies: Enabling Community Access to Petascale Climate Datasets

Dean N. Williams
Lawrence Livermore National Laboratory

David E. Bernholdt
Oak Ridge National Laboratory

Ian T. Foster
Argonne National Laboratory

Don E. Middleton
National Center for Atmospheric Research

1. Introduction

Climate research is inherently a multidisciplinary endeavor. As researchers strive to understand the complexity of our climate system, they form multi-institutional and multinational teams to tackle "Grand Challenge" problems. These multidisciplinary, virtual organizations need a common software infrastructure to access the many large global climate model datasets and tools. It is critical that this infrastructure provide equal access to climate data, supercomputers, simulations, visualization software, whiteboard, and other resources. To this end, we established the Earth System Grid (ESG) Center for Enabling Technologies (ESG-CET),[1] a collaboration of seven U.S. research laboratories (Argonne, LANL, LBNL, LLNL, NCAR, NOAA/PMEL, and ORNL) and a university (USC/ISI) working together to identify and implement key computational and informational technologies for advancing climate change science. Sponsored by the Department of Energy (DOE) Scientific Discovery through Advanced Computing (SciDAC)-2[2] program, through the Offices of Advanced Scientific Computing Research (OASCR)[3] and the Offices of Biological and Environmental Research (OBER),[4] ESG-CET utilizes and develops computational resources, software, data management, and collaboration technologies to support observational and modeling data archives.

Work on ESG began with the "Prototyping an Earth System Grid" (ESG I) project, initially funded under the DOE Next Generation Internet (NGI) program, with follow-on support from OBER and DOE's Mathematical, Information, and Computational Sciences (MICS) office. In this prototyping project, we developed Data Grid technologies for managing the movement and replication of large datasets, and applied these technologies in a practical setting (an ESG-enabled data browser based on current climate data analysis tools), achieving cross-country transfer rates of more than 500 Mb/s. Having demonstrated the potential for remotely accessing and analyzing climate data located at sites across the U.S., we won the "Hottest Infrastructure" award in the Network Challenge event at the SC'2000 conference.

While the ESG I prototype provided a proof of concept ("Turning Climate Datasets into Community Resources"), the SciDAC Earth System Grid (ESG) II project[5] [6] made this a reality. Our efforts in that project targeted the development of metadata technologies[7] (standard schema, XML metadata extraction based on netCDF, and a Metadata Catalog Service), security technologies[8] (Web-based user registration and authentication, and community authorization), data transport technologies[9] [10] (GridFTP-enabled OPeNDAP-G for high-performance access, robust multiple file transport and integration with mass storage systems, and support for dataset aggregation and subsetting), and web portal technologies to provide interactive access to climate data holdings. At this point, the technology was in place and assembled, and ESG II was poised to make a substantial impact on the climate modeling community.

In 2004, the National Center for Atmospheric Research (NCAR), a premier climate science laboratory and lead institution for the Community Climate System Model (CCSM) modeling collaboration, began its first publication of climate model data into the ESG system, drawing on simulation data archived at LANL, LBNL, NCAR, and ORNL. Late that same year, the Program for Climate Model Diagnosis and Intercomparison (PCMDI), an internationally recognized climate data center at LLNL, launched a production service providing access to climate model data germane

[1] Earth System Grid (ESG) - Turning Climate Model Datasets into Community Resources - http://www.earthsystemgrid.org/, 2004 – 2007.

[2] Scientific Discovery through Advanced Computing (SciDAC) - http://www.scidac.gov/, 2007.

[3] Office of Advanced Scientific Computing Research (OASCR) - http://www.science.doe.gov/ascr/, 2007.

[4] Offices of Biological and Environmental Research (OBER) - http://www.science.doe.gov/ober/, 2007.

[5] Bernholdt, D., Bharathi, S., Brown, D., Chanchio, K., Chen, M., Chervenak, A., Cinquini, L., Drach, B., Foster, I., Fox, P., Garcia, J., Kesselman, C., Markel, R., Middleton, D., Nefedova, V., Pouchard, L., Shoshani, A., Sim, A., Strand, G. and Williams, D. The Earth System Grid: Supporting the Next Generation of Climate Modeling Research. *Proceedings of the IEEE*, 93 (3). 485-495. 2005.

[6] Foster, I., Alpert, E., Chervenak, A., Drach, B., Kesselman, C., Nefedova, V., Middleton, D., Shoshani, A., Sims, A. and Williams, D., The Earth System Grid: Turning Climate Datasets Into Community Resources. in *82nd Annual American Meteorological Society Meeting*, (Orlando, FL., 2002).

[7] Eaton, B., Gregory, J., Drach, B., Taylor, K. and Hankin, S. NetCDF Climate and Forecast Metadata Conventions. http://cf-pcmdi.llnl.gov, 2007.

[8] Welch, V., Siebenlist, F., Foster, I., Bresnahan, J., Czajkowski, K., Gawor, J., Kesselman, C., Meder, S., Pearlman, L. and Tuecke, S., Security for Grid Services. *12th IEEE International Symposium on High Performance Distributed Computing*. 2003.

[9] Fox, P., Garcia, J. and West, P. OPeNDAP for the Earth System Grid. *Data Science Journal*. 2007.

[10] Allcock, B., Bresnahan, J., Kettimuthu, R., Link, M., Dumitrescu, C., Raicu, I. and Foster, I., The Globus Striped GridFTP Framework and Server. *Supercomputing 2005 (SC '05) conference proceedings*. 2005.

to the Intergovernmental Panel on Climate Change (IPCC) 4th Assessment Report (AR4).[11] (Because of international data requirements, restrictions, and timelines, the NCAR and PCMDI ESG data holdings were separated.) ESG has since become a world-renowned leader in developing technologies that provide scientists with virtual access to distributed data and resources.

In its first full year of production (late 2005), the two ESG sites provided access to a total of 220 TB of data, served over 3,000 registered users, and delivered over 100 TB of data to users worldwide. Analysis of just one component of ESG data holdings, those relating to the Coupled Model Intercomparison Project phase 3 (CMIP3), resulted in the publication of over 100 peer-reviewed scientific papers.

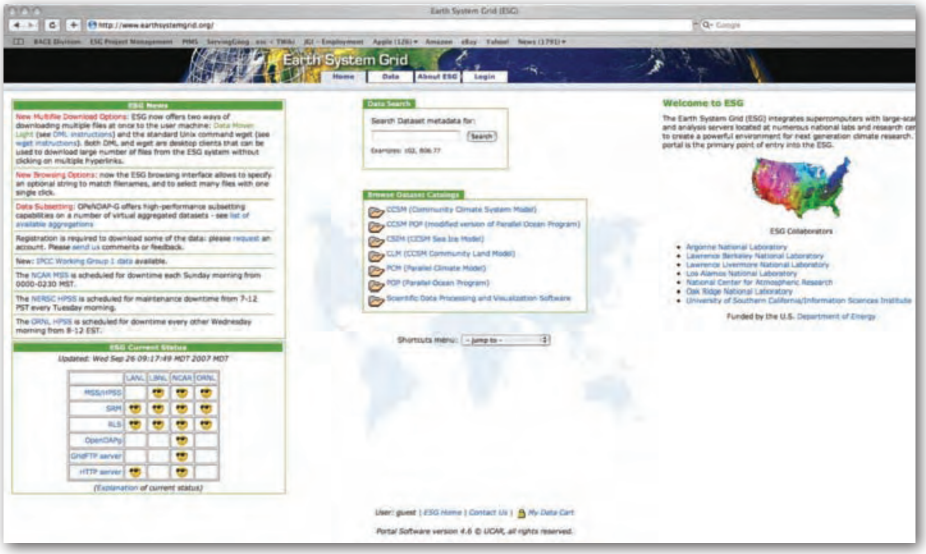
In 2006 we launched the current phase of the ESG effort, the ESG Center for Enabling Technologies (ESG-CET). The primary goal of this stage of the project is to broaden and generalize the ESG system to support a more broadly distributed, more international, and more diverse collection of archive sites and types of data. An additional goal is to extend the services provided by ESG beyond access to raw data by developing "server-side analysis" capabilities that will allow users to request the output from commonly used analysis and intercomparison procedures. We view such capabilities as essential if we are to enable large communities to make use of petascale data. However, their realization poses significant resource management and security challenges.

2. Overview of ESG

ESG is a large, production, distributed system – a Data Grid – with primary access points via three web portals: one for general climate research data; another dedicated to the IPCC activity; and a third for the Community Climate System Model (CCSM) Biogeochemistry (BGC) Working Group, which is just going into production at ORNL. The deployment of these three separate portals is driven by international data requirements, restrictions, and timelines. However, they are all based on the same underlying software system. Our goal in ESG-CET is to achieve complete integration of these focused archives, while providing the tailored access and other controls required by the various data owners. In this way, we will provide ESG users with coherent access to ever-growing and increasingly diverse collections of global community climate data.

Users of the ESG portal must first register, at which time they are granted appropriate privileges and access to data collections. The main portal page, shown in Figure 1, provides news, status, and live monitoring of ESG. Once logged in, users may either search or browse ESG catalogs to locate desired datasets, with the option of browsing both collection-level and file/usage-level metadata. Based on this perusal of the catalogs, users may gather a collection of files into a "DataCart" or request an "aggregation," which allows them to request a specific set of variables subject to a spatiotemporal constraint. Selected data may then be downloaded to the user's system, including datasets that are on deep storage at multiple sites behind security firewalls. Group-based authorization mechanisms allow the ESG administrators to control which users can access which data. These capabilities are made possible by a collection of ESG management, data publishing, and large-scale data transport tools.

[11] Intergovernmental Panel on Climate Change (IPCC) 4th Assessment Report - http://www.ipcc.ch/activity/ar.htm, 2007.



ESG News

New Multifile Download Options: ESG now offers two ways of downloading multiple files at once to the user machine: Data Mover Light (see DML instructions) and the standard Unix command wget (see wget instructions). Both DML and wget are desktop clients that can be used to download large number of files from the ESG system without clicking on multiple hyperlinks.

New Browsing Options: now the ESG browsing interface allows to specify an optional string to match filenames, and to select many files with one single click.

Data Subsetting: OPeNDAP-G offers high-performance subsetting capabilities on a number of virtual aggregated datasets - see list of available aggregations

Registration is required to download some of the data: please request an account. Please send us comments or feedback.

New: IPCC Working Group 1 data available.

The NCAR MSS is scheduled for downtime each Sunday morning from 0000-0230 MST.

The NERSC HPSS is scheduled for maintenance downtime from 7-12 PST every Tuesday morning.

The ORNL HPSS is scheduled for downtime every other Wednesday morning from 8-12 EST.

ESG Current Status

Updated: Wed Sep 26 09:17:49 MDT 2007 MDT

	LANL	LBNL	NCAR	ORNL
MSS/HPSS		😎	😎	😎
SRM	😎	😎	😎	😎
RLS	😎	😎	😎	😎
OpenDAPg			😎	
GridFTP server			😎	
HTTP server	😎		😎	

(Explanation of current status)

Data Search

Search Dataset metadata for:

Examples: t02, 806.77

Browse Dataset Catalogs

- CCSM (Community Climate System Model)
- CCSM POP (modified version of Parallel Ocean Program)
- CSIM (CCSM Sea Ice Model)
- CLM (CCSM Community Land Model)
- PCM (Parallel Climate Model)
- POP (Parallel Ocean Program)
- Scientific Data Processing and Visualization Software

Welcome to ESG

The Earth System Grid (ESG) integrates supercomputers with large-sca
and analysis servers located at numerous national labs and research cer
to create a powerful environment for next generation climate research. T
portal is the primary point of entry into the ESG.

ESG Collaborators

- Argonne National Laboratory
- Lawrence Berkeley National Laboratory
- Lawrence Livermore National Laboratory
- Los Alamos National Laboratory
- National Center for Atmospheric Research
- Oak Ridge National Laboratory
- University of Southern California/Information Sciences Institute

Funded by the U.S. Department of Energy

User: guest | ESG Home | Contact Us | My Data Cart

Portal Software version 4.6 © UCAR, all rights reserved.

Figure 1. ESG Portal

The ESG system includes a metrics-gathering capability that keeps track of user activity. Interactive displays as well as reports allow us to track what data is downloaded, how often, and by whom. The resulting data has proved invaluable not only for reporting to sponsors and data owners on degree of use (its initial intent), but also as a guide to system development and optimization.

3. Overall Impact

ESG has had a significant impact upon the national and international climate community by enabling broad dissemination of important data holdings, including the Community Climate System Model (CCSM) data archive, the Intergovernmental Panel on Climate Change (IPCC) 4th Assessment Report (AR4) data archive, and now the CCSM BGC Carbon-Land Model Intercomparison Project (C-LAMP)[12] data archive. All three archives are well known to the user community and, since ESG's official release, the community has downloaded well over 300 TB of data, well over 1 million files, and reported over 300 journal articles,[13] all in a short time span.

The ESG team works closely with the CCSM community to publish CCSM model data into the ESG archives. Collaborating with CCSM scientists and data providers, the ESG team developed and utilized Grid technology that interfaces into the ESG metadata database allowing the CCSM community to view and manage all information related to generating, defining, and archiving CCSM model simulation runs. This interface allows scientists to impose selective access control on project runs, to sort information by any type, and to enter data collaboratively. The long-term goal is to tie the metadata ingestion process to the actual CCSM run workflow, so that model simulation metadata can be added automatically into the ESG data holdings.

The ESG user base comprises climate scientists, analysts, educators, governments (both domestic and abroad), private industry, and many others. CCSM data, along with other important datasets accessible via ESG, such as those produced by the Parallel Climate Model (PCM)[14] and the Parallel Ocean Program (POP),[15] have been used in numerous scientific papers, impact analyses, urban planning and ecosystem monitoring studies, education, and other activities. By allowing access, ESG enables scientists, hardware and software engineers, universities and others to examine and learn how a state-of-the-art climate model works, and to provide suggestions and

[12] CCSM Carbon LAnd Model intercomparison Project (C-LAMP) - http://www.climatemodeling.org/c-lamp/, 2007.

[13] World Climate Research Program (WCRP) CMIP3 (IPCC AR4) Subproject Publications - http://www-pcmdi.llnl.gov/ipcc/subproject_publications.php, 2007.

[14] The Parallel Climate Model - http://www.cgd.ucar.edu/pcm/, 2007.

[15] The Parallel Ocean Program (POP) ocean circulation model - http://climate.lanl.gov/Models/POP/, 2007.

enhancements for its scientific accuracy, portability, and performance. We even receive occasional queries from the general public, asking how they can use data published in ESG to better understand climate change issues or local impacts.

ESG was thrust into international collaboration when it was asked in late 2003 to support the IPCC/Working Group on Coupled Models (WGCM) need to distribute data to the international climate community. The IPCC, which was jointly established by the World Meteorological Organisation (WMO) and the United Nations Environment Programme, carries out periodic assessments of the science of climate change. Fundamental to this effort is the production, collection and analysis of data from climate model simulations carried out by major international research centers. Analysis of a set of standard climate-change simulations from many modelling centers provides comprehensive understanding of the strengths and weaknesses of climate models, as well as which aspects of the simulation results may be due to characteristics of specific models and which are generally observed across multiple models. The IPCC and WGCM requested that PCMDI at LLNL collect model output data from these IPCC simulations and distribute these to the community via ESG. Since this effort began, IPCC model runs published to the climate community via the CMIP3 (IPCC AR4) ESG portal total just over 35 TB (78,158 files), and some 1,400 users have registered to receive IPCC data for analysis. Figure 2 shows the daily download rate over time.

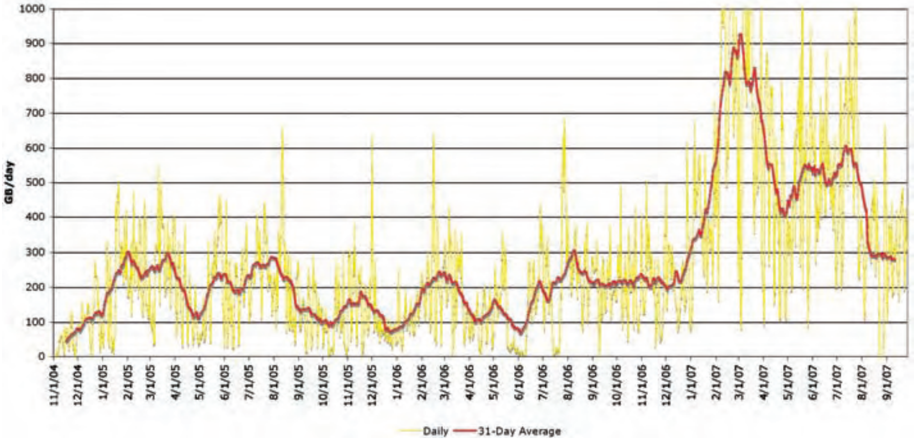


Figure 2. CMIP3 (IPCC AR4) Download Rates in Gigabytes/day

New to ESG is the dissemination of C-LAMP[12] biogeochemistry data. This model inter-comparison project has two terrestrial BGC modules linked to the same set of prescribed ocean BGC fluxes, together with the CCSM's interactive atmosphere and interactive land surface modules. The C-LAMP effort involves two separate experiments: one in which atmospheric data comes from observations, the other in which it is calculated by CAM3, the current atmospheric component of the CCSM. The first experiment will determine how well land-air fluxes of CO2 are simulated by the two BGC modules, given the observed climate. The second will determine the effect of the atmospheric model's climate bias (notably in precipitation) on the simulated CO2 fluxes. The C-LAMP experimental output is now being archived and disseminated on an ESG C-LAMP site modelled after the ESG CMIP3 (IPCC AR4). This archive will initially be open only to members of the BGC Working Group, but ultimately the working group will open up the data to any interested researcher.

Knowledge and expertise gained from ESG have helped the climate community plan effective strategies to manage a rapidly growing data environment. Approaches and technologies developed under the ESG project have also impacted data-simulation integration in other disciplines, such as astrophysics, molecular biology, and materials science.

4. The Next-Generation ESG

Building upon ESG's success to date, ESG-CET is developing a next-generation environment targeted at enabling flexible, efficient, and universal access to yet larger datasets, and to harnessing distributed worldwide resources for the purpose of advancing climate and related impacts research and assessment. In creating this new community infrastructure, ESG-CET will turn even more climate model data into true community resources and place advanced capabilities in the hands of a substantial user base community.

Our high-level goals for this next phase of ESG are driven by scientific objectives relevant to DOE's scientific priorities over the next several years. In brief, they are, firstly, to sustain successful existing ESG services and, secondly, to address scientific needs related to projected future data management and analysis requirements, with a particular focus on:

- Preparing for the CMIP4 IPCC 5th Assessment Report (AR5) in 2010.
- Publishing and enabling processing of the massive data produced by the Climate Science Computational End Station (CCES) at ORNL's NCCS/LCF.
- Supporting a wide-range of climate model evaluation activities aimed at improving climate change research.

To support this effort, we will broaden ESG to support multiple types of model and observational data, provide more powerful (client-side) ESG access and analysis services, enhance interoperability between common climate analysis tools and ESG, and enable end-to-end simulation and analysis workflow. Figure 3 depicts the scientific data management and analysis requirements in relationship to the ESG development timeframe. We specifically note that a distributed testbed for CMIP4 (IPCC AR5) must be in place by early 2009.

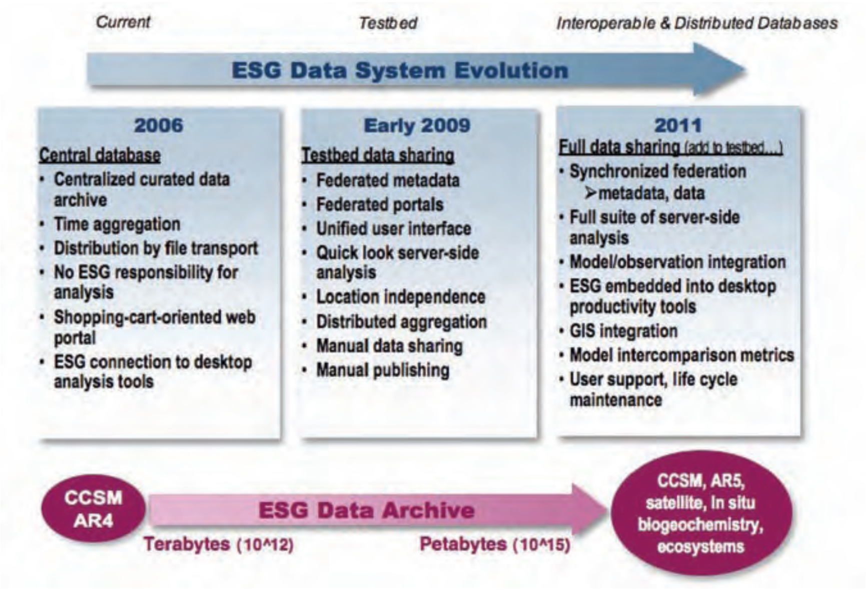

Current
Testbed
Interoperable & Distributed Databases
ESG Data System Evolution
2006
Central database
• Centralized curated data archive
• Time aggregation
• Distribution by file transport
• No ESG responsibility for analysis
• Shopping-cart-oriented web portal
• ESG connection to desktop analysis tools
Early 2009
Testbed data sharing
• Federated metadata
• Federated portals
• Unified user interface
• Quick look server-side analysis
• Location independence
• Distributed aggregation
• Manual data sharing
• Manual publishing
2011
Full data sharing (add to testbed...)
• Synchronized federation
➤metadata, data
• Full suite of server-side analysis
• Model/observation integration
• ESG embedded into desktop productivity tools
• GIS integration
• Model intercomparison metrics
• User support, life cycle maintenance
CCSM AR4
ESG Data Archive
Terabytes (10^12)
Petabytes (10^15)
CCSM, AR5, satellite, In situ biogeochemistry, ecosystems

Figure 3. Evolving ESG to the Petascale: High-level ESG-CET Roadmap

The ESG-CET architecture must be generalized to enable a larger number of sites with more diverse capabilities to selectively federate, cooperate, or operate in a standalone fashion as individual sites desire. The architecture must support a variety of user access mechanisms, including multiple portals and service- or API-based access, and data delivery mechanisms. This architecture must also be robust in the face of system and network failures at the participating sites.

To address these concerns, we designed the federated ESG-CET architecture (see Figures 4 and 5) to provide interoperability and enhanced functionality to users, and are now implementing the new design through a combination of evolution of existing software, development of new tools, and integration with third-party software. The much wider deployment anticipated for the next generation system means that software deployability and maintainability are vital considerations in determining the most effective implementation.

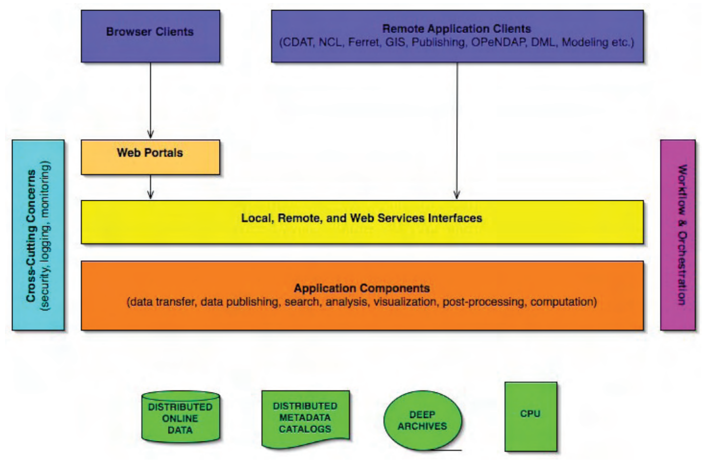

Browser Clients
Remote Application Clients
(CDAT, NCL, Ferret, GIS, Publishing, OPeNDAP, DML, Modeling etc.)
Web Portals
Cross-Cutting Concerns
(security, logging, monitoring)
Local, Remote, and Web Services Interfaces
Application Components
(data transfer, data publishing, search, analysis, visualization, post-processing, computation)
Workflow & Orchestration
DISTRIBUTED ONLINE DATA
DISTRIBUTED METADATA CATALOGS
DEEP ARCHIVES
CPU

Figure 4. Future ESG-CET Architecture

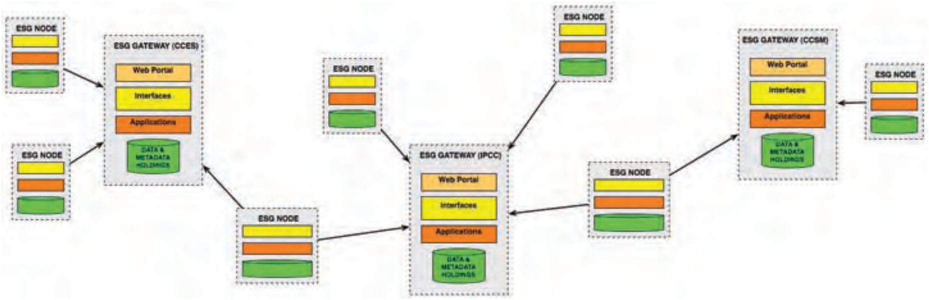


Figure 5. The ESG-CET Federated System

5. Conclusion

ESG has made significant progress towards the definition of the federated metadata, security, and data services required to enable distributed access to, and analysis of, large quantities of climate simulation data. The current production-level ESG system primarily addresses the tasks of publishing and cataloging terabytes of climate model data for a diverse set of registered users. We are now working to take ESG-CET to the next level of distributed environments with an even greater emphasis on federation and server-side capabilities. ESG-CET will build upon the current ESG system and target flexibility, efficiency, and more universal access while expanding to serve much larger archives (petabytes), as required for CMIP4 (IPCC AR5), CCSM, and CCES. To this end, ESG-CET is working with disparate technologies and partnering with national and international leaders in the computer and climate communities to build a robust data and analysis distributed infrastructure in support of advancing climate change research.

Acknowledgements

This work is supported through the U.S. Department of Energy Office of Science, Offices of Advanced Scientific Computing Research and Biological and Environmental Research, through the SciDAC program. Argonne National Laboratory is managed by Argonne Chicago LLC under Contract DE-AC02-06CH11357. Information Sciences Institute is a research institute of the Viterbi School of Engineering at the University of Southern California. Lawrence Berkeley National Laboratory is managed by the University of California for the U.S. Department of Energy under contract No. DE-AC02-05CH11231. Lawrence Livermore National Laboratory is managed by the University of California for the U.S. Department of Energy under contract No. W-7405-Eng-48. Los Alamos National Security is managed by LLC (LANS) for the U.S. Department of Energy under the contract No. DE-AC52-06NA25396. National Center for Atmospheric Research is managed by the University Corporation for Atmospheric Research under the sponsorship of the U.S. National Science Foundation. Oak Ridge National Laboratory is managed by UT-Battelle, LLC for the U.S. Dept. of Energy under contract DE-AC-05-00OR22725. Pacific Marine Environmental Laboratory is under the National Oceanic and Atmospheric Administration's line office of Ocean and Atmosphere Research, which lies within the U.S. Department of Commerce.

The ESG-CET executive committee consists of David Bernholdt, ORNL; Ian Foster, Argonne; Don Middleton, NCAR; and Dean Williams, LLNL.

This document was reviewed and released under the Lawrence Livermore National Laboratory guidelines and has been assigned the UCRL number: UCRL-TR-235089.

PUBLISHERS
Fran Berman, Director of SDSC
Thom Dunning, Director of NCSA

EDITOR-IN-CHIEF
Jack Dongarra, UTK/ORNL

MANAGING EDITOR
Terry Moore, UTK

CENTER SUPPORT
Warren Froelich, SDSC
Bill Bell, NCSA

PRODUCTION EDITOR
Scott Wells, UTK

GRAPHIC DESIGNER
David Rogers, UTK

DEVELOPER
Don Fike, UTK

CTWatch QUARTERLY

ISSN 1555-9874 VOLUME 3 NUMBER 4 NOVEMBER 2007

Software Enabling Technologies for Petascale Science

GUEST EDITOR **FRED JOHNSON**

AVAILABLE ON-LINE:
http://www.ctwatch.org/quarterly/

E-MAIL CTWatch QUARTERLY:
quarterly@ctwatch.org

CTWatch is a collaborative effort

http://icl.cs.utk.edu/ http://www.ncsa.uiuc.edu/ http://www.sdsc.edu/

CTWatch Quarterly is a publication of

http://www.ci-partnership.org/

Sponsored by

http://www.nsf.gov/

Any opinions expressed in this publication belong to their respective authors and are not necessarily shared by the sponsoring institutions or the National Science Foundation (NSF). Any trademarks or trade names, registered or otherwise, that appear in this publication are the property of their respective owners and do not represent endorsement by the editors, publishers, sponsoring institutions or agencies of CTWatch.