Report of the Workshop on Petascale Systems Integration for Large Scale Facilities

San Francisco, California
May 15-16, 2007

Workshop Sponsors
Dan Hitchcock, Barbara Helland - DOE Office of Science – ASCR
Workshop Organizer: William T.C. Kramer – NERSC
Report Author: William T.C. Kramer – NERSC
Report Editors: Ucilia Wang, Jon Bashor

Speakers
Mark Seager (LLNL), Tom Bettge (NCAR), Jeff Beckelhimer (Cray), Chulho Kim (IBM), Wayne Vieira (Sun), Dave Sundestrom (Linux Networx), Ray Bair (ANL), Patricia Kovatch (SDSC), Brad Comes (DODMod), Bob Ciotti (NASA), William T.C. Kramer (NERSC)

Break Out Group Leaders
Group 1 – Howard Walter (NERSC), Gary New (NCAR)
Group 2 - Tom Engle (NCAR), Rob Pennington (NCSA)
Group 3 - Brad Comes (DOD), Buddy Bland (ORNL)
Group 4 - Bob Tomlinson (LANL), Jim Kasdorf (PSC)
Group 6 - Dave Skinner (NERSC), Kevin Regimbal (PNNL

Workshop Attendees/Report Contributors
Bill Allcock, Anna Maria Bailey, Ron Bailey, Ray Bair, Ann Baker, Robert Ballance, Jeff Becklehimer, Tom Bettge, Richard Blake, Buddy Bland, Tina Butler, Nicholas Cardo, Bob Ciotti, Susan Coghlan, Brad Comes, Dave Cowley, Kim Cupps, Thomas Davis, Daniel Dowling, Tom Engel, Al Geist, Richard Gerber, Sergi Girona, Dave Hancock, Barbara Helland, Dan Hitchcock, Wayne Hoyenga, Fred Johnson, Gary Jung, Jim Kasdorf, Chulho Kim, Patricia Kovatch, William T.C. Kramer, Paul Kummer, Steve Lowe, Tom Marazzi, Dave Martinez, Mike McCraney, Chuck McParland, Stephen Meador, George Michaels, Tommy Minyard, Tim Mooney, Wolfgang E. Nagel, Gary New, Rob Pennington, Terri Quinn, Kevin Regimbal, Renato Ribeiro, Lynn Rippe, Jim Rogers, Jay Scott, Mark Seager, David Skinner, Kevin Stelljes, Gene Stott, Sunny Sundstrom, Bob Tomlinson, Francesca Verdier, Wayne Vieira, Howard Walter, Ucilia Wang, Harvey Wasserman, Kevin Wohlever, Klaus Wolkersdorfer

http://www.nersc.gov/projects/HPC-Integration/

Summary
There are significant issues regarding Large Scale System integration that are not being addressed in other forums such as current research portfolios or vendor user groups. Unfortunately, the issues in the area of large-scale system integration often fall into a netherworld; not research, not facilities, not procurement, not operations, not user services. Taken together, these issues along with the impact of sub-optimal integration technology means the time required to deploy, integrate and stabilize large scale system may consume up to 20 percent of the useful life of such systems. Improving the state of the art for large scale systems integration has potential to increase the scientific productivity of these systems.[1]

Sites have significant expertise, but there are no easy ways to leverage this expertise among them . Many issues inhibit the sharing of information, including available time and effort, as well as issues with sharing proprietary information. Vendors also benefit in the long run from the solutions to issues detected during site testing and integration.

There is a great deal of enthusiasm for making large scale system integration a full-fledged partner along with the other major thrusts supported by funding agencies in the definition, design, and use of a petascale systems. Integration technology and issues should have a full "seat at the table" as petascale and exascale initiatives and programs are planned.

The workshop attendees identified a wide range of issues and suggested paths forward. Pursuing these with funding opportunities and innovation offers the opportunity to dramatically improve the state of large scale system integration.

Introduction
As high-performance computing vendors and supercomputing centers move toward petascale computing, every phase of these systems, from the first design to the final use, presents unprecedented challenges. Activity is underway for requirements definition, hardware and software design, programming models modifications, methods and tools innovation and acquisitions. After systems are designed and purchased, and before they can be used at petascale for their intended purposes, they must be installed in a facility, integrated with the existing infrastructure and environment, tested and then deployed for use. Unless system testing and integration is done effectively, there are risks that large scale systems will never reach their full potential.

To help lay the foundation for successful deployment and operation of petascale systems, the National Energy Research Scientific Computing Center at Lawrence Berkeley National Laboratory hosted a two-day workshop on "Petascale Systems Integration into

[1] If a useful life of a system is five year and large systems take a year to fully deploy, they are limited to only 80% of their full impact. Note, however, early user science is often done on these early systems somewhat mitigating the loss. None the less, shortening the integration time while still deploying systems that meet the highest quality expectations would directly impact all science.

Large Scale Facilities." Sponsored by DOE's Office of Advanced Scientific Computing Research (ASCR), the workshop examined the challenges and opportunities that come with petascale systems deployment. Nearly 70 participants from the U.S and Europe, representing 29 different organizations, joined the effort to identify the challenges; search for the best practices; share experiences and identify lessons learned. The workshop assessed the effectiveness of tools and techniques that are or could be helpful in petascale deployments and sought to determine potentially beneficial tools areas for research and development. The workshop also addressed methods to ensure that systems operate and perform at increasingly better levels throughout their lifetime. Finally, the workshop sought to add the collective experience and expertise of the attendees to the HPC community's body of knowledge as well as to establish a network of experts who will share information on an ongoing basis as petascale systems come on line.

Specifically, the goals of the workshop were to:

- Identify challenges and issues involved in the installation and deployment of large scale HPC systems
- Identify best practices for installing large-scale HPC systems into scientific petascale facilities
- Identify methods to assure system performance and function continue after initial testing and deployment
- Identify systematic issues and research issues for vendors, sites and facilities that would improve the speed and quality of deployment
- Share tools and methods that are helpful in expediting the installation, testing and configuration of HPC systems
- Establish communication paths for technical staff at multiple sites that might make HPC installations more effective
- Make recommendations to DOE and other stakeholders to improve the process of HPC system deployment

Workshop High Priority Summary Findings and Recommendations (*Sub recommendations are in italics)*

The workshop participants developed many suggestions in the course of their exploration. They evaluated the priorities for all of the recommendations and agreed on the highest priority ones that determine the success of petascale systems. These highest priority items are summarized below.

1. Improve the ability to record and process log data.

Recommendation: There will be a tremendous amount of logging data about system use and system status. *There must be new ways process this data – in real time. Specific improvements are:*

- *Develop common log data models and data formats that enable integrated analysis tools and easier sharing*
- *Develop consolidation of events*
- *Increase the amount of "Intelligent" to do broad analysis with automated detailed focus as needed.*
- *Increase the sites' and vendors' ability to do forensic analysis*

The vendors and sites are most likely to make progress in this area since a large amount of detailed knowledge and context is needed

Responsibility: Vendors and sites are most likely to make progress in this area since there is a large amount of detailed knowledge and context needed.

Recommendation: Share tools and system integration and operational data across sites. This will enable the ability to look for larger trends. Six sites agreed to share current data at the workshop with others indicating a willingness to consider sharing. Sharing of data also enables understanding different systems and code behavior. To accomplish this recommendation the following must happen:

- *A common/standard format must be defined.*
- *There will probably be a need to develop tools to convert from proprietary and/or current formats to the standard format.*
- *It is important to look beyond common file formats to the entire data model of log file analysis. This means involving expertise in data management, curation and data protection.*
- *This single effort, if undertaken seriously, would require an entire workshop or a series of workshops to make progress.*

Responsibility: The creation of common log file data models and formats requires new expertise not typically found in HPC communities. Stakeholders have the responsibility to fund teams that consist of HPC expertise and data description expertise to make progress.

Recommendation: *Build a repository of log files to start building tools*. At the workshop, eight sites agreed to share system log data under an agreement that the data would not be further shared. Indiana University agreed to host the site which the participating facilities will be responsible for creating. Steps to achieve this recommendation include:

- *Defining the data models and low level information to be captured.*
- *Make the data model extensible because unanticipated and new data may become available. This includes performing research on how to sanitize logs to release them to the general academic community for further contributions.*
- *Develop analysis tools. There should be an effort to look at applying statistical analysis to data.*
- *An initial, very useful tool would be to use the Simple Event Correlator – an open source package – to analyze events.*

- *Create improved methods to visualize the system state.*

This recommendation overlaps some of the security logging issues developed at the February Security Workshop http://www.dsd.lbl.gov/Workshops/CyberWorkshop/.

Responsibility: The creation of common log file data models and formats requires new expertise not typically found in HPC communities. Stakeholders have the responsibility to fund teams that consist of HPC expertise and data description expertise to make progress.

2. **The ability to have multiple versions of system software on the petascale systems.**

Recommendation: It is important to test system software at scale and to quickly go back and forth between versions of software. This includes the entire software stack and microcode. If there were *system software transition requirements in RFPs* for being able to switch between software levels, would vendors be able to provide a response. Two sites include such requirements in their RFPs and about six would consider adding such a requirement. *Sites suggested sharing RFPs* so they can learn from each other

It would be *beneficial for vendors to isolate system level software changes to provide limited test environments*. One problematic area is being able to separate firmware upgrades from system software upgrades. In many cases, if firmware is upgraded, older software versions can not be used. Having the *ability to partition the system, including the shared file system* without the cost of replicating all the hardware would be very helpful in many cases.

Responsibility: Creating these features is primarily a vendor responsibility.

3. Have better hardware and software diagnostics

Recommendation: *Better proactive diagnostics are critical* to getting systems into service on time and at the expected quality. Vendors are responsible for providing diagnostics to determine root causes. One very important requirement is to be *able to run diagnostics without taking the entire system* into dedicated mode.

Better definition is needed before vendors can develop better tools. There is a difference between testing and diagnosis. Testing is the process deciding if there is a problem. Diagnostics are used to diagnose the system once it is known that there is a problem. *Diagnostics should be able to point to a field replaceable unit so a repair action can take place*. Diagnostics should be created for all software as well as hardware.
To improve the areas of diagnostics, the *HPC community and vendors have to work together* to determine what already exists, prioritize what tools are needed and determine what can be done with system administration issues and with hardware issues.

Recommendation: HPC systems need shorter MTTR. The vendors at the workshop indicated that, if sites put MTTR requirements into RFPs to define the requirements for vendors, they could, over time, could respond with specific improvements. Currently, it is unclear what can be specified and the best manner to do so.

All sites want *on-line diagnostics to be as comprehensive as off-line diagnostics.* Sites also desire the ability to analyze soft errors to predict problems rather than just correcting for them. There is also a need to test diagnostics as part of an acceptance test, but this process is not at all clear, other than to observe their use during acceptance.

Responsibility: Sites have the responsibility to request these features and to value them when promised in RFP responses. Vendors have the responsibility to provide the features and stakeholders have the responsibility to fund research into soft error analysis.

4. I/O Activity and Analysis Tools

Recommendation: Data is an increasingly large issue and will possibly become unmanageable at the petascale. For example, some sites are taking more than a week to backup large file systems using the methods supplied by vendors. It is more difficult to get parallel data methods right for performance.

To manage the aggregate data on these systems, better information on what is going on at any moment is necessary. *Storage system tools that are the equivalent to HPM/PAPI for CPU performance data gathering are needed*. Where tools do exist, there is a wide variance in what they do and there are no common standards for data.

There are two purposes for monitoring and understanding data from storage systems. The first is to diagnose code performance. This enables the creation of new tools that can be used for application I/O performance understanding and improvement. Early tools of this type are being developed at Dresden ZIH. The second is to determine system performance and enable system managers to tune the system for the overall workload. This data would also make it easier for data system designers to make improvements in data systems.

There are significant challenges in getting to the information that resides in many layers of the systems. Visualizing and correlating I/O system data is challenging. Today's data tools mostly test point-to-point performance. They are not able to monitor and assess overall system behavior, true aggregate performance and conflicting use and demands. Many sites, including NERSC, LBNL, ORNL, SNL, LANL and LLNL are working on this in different ways, using tools such as IOR and AMR I/O benchmarks.

There should be another workshop focused on this issue.

Responsibility: This will take a concerted, funded research and development effort, much the same as the Performance API (PAPI) did. Once defined, vendors will have the responsibility to implement the functionality.

5. Peer Interactions and Best Practices

In the past, the facility and building staffs at major HPC facilities have not had the opportunity for detailed technical interactions to exchange practices. HPC facility management differs from other building management in significant ways, including the rate of change that occurs with new technology being incorporated into the building. Petascale systems will stress facilities, even specially designed facilities. *Having facility manager peers brainstorm before designs and major changes to assure all issues are covered and the latest approaches are taken is critical to the ability of facilities to operate close to the margins*.

There is a long history of sharing benchmark performance data but not for other data such as reliability, software and hardware problems, security issues and user issues. Now all sites are multi-vendor sites. Sharing of system problem reports (SPRs) is something that is not done across sites. This was a common practice in formal organizations such as user groups, but they have become less effective in this as the HPC business model changes and as sites become more multi-vendor. Sites now have many more vendors so there is no single (or even a few) places to share information. Furthermore, no mechanism exists to share problem information across vendor-specific systems. Such information sharing is not deemed appropriate for research meetings so rarely appears on the agenda. In some cases, vendors actively inhibit sharing of problems reports, so facilitating community exchange is critical to success.

Similar issues exist when systems are going through acceptance at different sites. While this may be more sensitive, some form of sharing may be possible. Even having a summary report of the experiences of each large system installation after integration would be useful. There should be a common place to post these reports, but access must be carefully controlled so vendors are not afraid to participate. For facilities-specific issues, the APCOM Facility conference may have some value.

Indicative of the need and interest in this area, some sites already mark their SPRs public rather than private when possible. Five sites expressed willingness to share their SPRs if a mechanism existed, and more indicated they would consider it.

To have more study of this area and to generate wider participation, it may be useful to *explore a special journal edition devoted to Large Scale System Integration.* Likewise, *creating an on-line community*, referred to "HPC_space.com" by the attendees, will provide the ability to share more real time issues through mechanisms such as twikis or blogs. However, the "rules of engagement" must carefully consider the needs of vendors, systems, areas of interest and sites. Look at *Instructions to America* that creates communities of practice on many topics.

Responsibility: This is primarily a site and community responsibility.

6. Tools for Understanding Performance and Function for storage (disk) management and reliability and other Key components.

Improved tools must be developed to move and manage data. Other *tools are needed to get information from data*. Data insight tools should operate in real time with application improvements. The storage community is working on improved tools to manage data, and some of these are already being included in SciDAC projects. *The science research community can and should deal with the performance of small data writes and new ways to handle very large numbers of files. Vendors and others are dealing with Information Lifecycle Management, including hierarchies of data movement.* They also should *provide the ability to monitor large numbers of storage and fabric devices* as well as tools for disk management.

Other key components are to *create tools for understanding interconnects such as PAPI/IPM and performance profiling tools for other programming models such as PGAS languages.*

Responsibility: This will take a concerted, funded research and development effort, much the same as the Performance API (PAPI) did. Once defined, vendors will have the responsibility to implement the functionality.

7. Parallel Debuggers

Parallel debuggers can be a significant help with integration issues as well as help users be more productive. It is not clear that sites understand how users debug their large codes. Although this topic was partly covered at a DOE-sponsored workshop on petascale tools in August 2007http://ft.ornl.gov/~vetter/2007-08-01_sdtpc/, it is not clear there is a connection with the systems integration community.

Responsibility: Sites have the responsibility for competitively require improved debugging features. Stakeholders may need to fund research to develop new methods of debugging at the 10,000 to 100,000 process level.

Additional key observations were made by the attendees.

All sites face many similar integration issues, independent of the vendors providing the systems. Many issues identified during the workshop span vendors, and hence are persistent. Vendors will never see the number and diversity of the problems in their own testing environment that are seen during integration at sites because they concentrate only on the issues that are immediately impacting their systems. There currently is no forum to deal with applied issues of large scale integration and management that span vendors.

The problem of integrating petascale systems is too big for any one site to solve. The systems are too big and complex for a single workload to uncover all the issues. Often

simply fully understanding problems is a challenge that takes significant effort. Increased use of open and third party software makes each system unique and compounds the problem. Funding agencies are not willing to support needed improvements in the area of large scale integration because it does is not fit into their research portfolios.

Site priorities for when and how systems are to be considered ready for production must apply to acceptance and integration decisions rather than vendor priorities. It would be useful to create a framework for sites to perform the quality assurance that can be used by vendors in pre-installation checklists. The competition between sites should not be allowed to get in the way of improving integration methods.

A number of suggestions for post workshop activities were made. One is to create an infrastructure for continuing the momentum made in this workshop. Another is to provide further forums/formats for future events that continue to address these issues. It may be possible to follow on with a workshop at SC 07 and PNNL is willing to help organize this.

NSF indicated they believe more collaboration is needed in this area of large scale integration. Many pressures are involved with fielding large scale systems. Cost pressures come from electrical costs increasing dramatically and the costs to store data will "eat us alive" as the community goes to petascale unless data and cost issues are addressed systematically.

Some systematic problems prevent more rapid quality integration. Areas that need additional work and more detailed investigation are interconnects, I/O, large system OS issues and facilities improvements. Having system vendors at this first workshop was very beneficial, and future meetings should include interconnect and storage vendors.

The basic question remains, "Have we looked far enough in advance or are we just trying solving the problems we have already seen?" It is not clear whether the integration issues for Petascale systems were completely identified, but the findings and recommendations of the workshop are an excellent starting point. It is clear however that even the issues identified will require a persistent community wide response to address adequately for Petascale systems.

Workshop Overview

The two-day meeting in San Francisco attracted about 70 participants from roughly 30 supercomputer centers, vendors, other research institutions and DOE program managers The discussions covered a wide-range of topics, including facility requirements, integration technologies, performance assessment, and problem detection and management. While the attendees were experienced in developing and managing supercomputers, they recognized that the leap to petascale computing requires more creative approaches.

> "We are going through a learning curve. There is fundamental research to be done because of the change in technology and scale," said Bill Kramer, NERSC's General Manager who led the workshop.

In his welcoming remarks, Dan Hitchcock, Acting Director of the Facilities Division within ASCR, urged more collaboration across sites fielding large scale systems, noting that various stakeholders in the high-performance computing community have historically worked independently to solve thorny integration problems.

Mark Seager from Lawrence Livermore National Laboratory and Tom Bettge from the National Center for Atmospheric Research (NCAR) helped kick-start the workshop by sharing their experiences with state-of-the-art computer systems and deploying their most powerful systems. Both Seager and Bettge said having sufficient electrical power to supply massive supercomputers is becoming a major challenge. A search for more space and reliable power supply led NCAR to Wyoming, where it will partner with the state of Wyoming and the University of Wyoming to build a $60-million computer center in Cheyenne. One of the key factors in choosing the location is the availability of stable electrical power. (Interestingly, the stability comes as much from the fact there is a Wal-Mart refrigeration plant nearby that draws 9 MW of power)

Seager advocated the creation of a risk management plan to anticipate the worst-case scenario. "I would argue that the mantra is 'maximizing your flexibility,'" Seager said. "Integration is all about making lemonade out of lemons. You need a highly specialized customer support organization, especially during integration."

Six breakout sessions took place over the two days to hone in on specific issues, such as the best methods for performance testing and the roles of vendors, supercomputer centers and users in ensuring the systems continue to run well after deployment.

The workshop program included a panel of vendors offering their views on deployment challenges. The speakers, representing IBM, Sun Microsystems, Cray and Linux Networx, discussed constraints they face, such as balancing the need to invest heavily in research and development with the pressure to make profit.

A second panel of supercomputer center managers proffered their perspectives on the major hurdles to overcome. For example, Patricia Kovatch from the San Diego Supercomputer Center hypothesized that the exponential growth in data could cost her center $100 million a year for tapes in order to support a petascale system unless new technology is created. Currently the center spends about $1 million a year on tapes.

> "We feel that computing will be driven by memory, not CPU. The tape cost is a bigger problem than even power," Kovatch told attendees.

The amount and performance of computer memory is clearly one of key challenges. Leaders from each breakout sessions presented slides detailing the daunting tasks ahead, including software development, acceptance testing and risk management.

Breakout Session Summaries

Charge to Breakout Group #1: Integration Issues for Facilities

Petascale systems are pushing the limits of facilities in terms of space, power, cooling and even weight. There are many complex issues facility managers must deal with when integrating large scale systems and these will get more challenging with Petascale systems. Example issues the group could address:

- *While we all hope technology will reverse these trends, can we count on it?*
- *Besides building large facilities (at Moore's Law rates) how can we optimize facilities?*
- *How can the lead times and costs for site preparation be reduced?*
- *Can real time adjustments be made rather than over- design?*

Report of Break Out Group #1

The discussion covered three major scenarios:

- Designing a new building
- Planning of facilities
- Upgrading an existing facility

Designing a new building

Although participants agreed that designing a new building with extensive infrastructure up front can significantly reduce long term costs, this is quite a challenge when the machines to be procured and deployed are not known during the design phase. However, the cost savings can be substantial.

Computing facilities have several aspects that do not exist in standard building projects. The most obvious is the rate of change computing buildings must support due to the rapid technology advances associated with Moore's Law and computing. It is typical for a computing facility to receive major new equipment every two to three years. It is also typical that the entire computing complex turns over in 6 to 8 years. This new equipment makes substantial demands on building infrastructure. As with construction, the cost to retrofit can far exceed the cost of original implementation.

Another difference in computing facilities for the giga-, tera- and petascale is there are expected to be significant changes in cooling technology, with cycles of air cooling and liquid cooling cycling once every 10 to 15 years. Standard buildings have a 30 to 50 year life cycles. Unless designed to be highly flexible, computing facilities will have to be built on a much shorter time cycle.

One important recommendation is to close the gap between systems and facilities staff at centers. This could help each center develop a planning matrix to determine the correlation between major categories of integrating effort and cost to the system technology. The major categories are space, power, cooling, networking and storage.

Each category is a determination of cost vs. benefit and mission vs. survival. What are the tradeoffs in terms of costs vs. benefits? What are the most painful areas of upgrade later in the facility's lifecycle? What is the anticipated life of the facility? How many procurement cycles are going to be involved with this facility? What is the scope of major systems and the associated storage?

Each organization will need to make its own determination of what they are willing and/or able to fund at a given point in time. With petascale facilities, consideration needs to be given to the equipment itself, regardless of the mission it performs. Underlying these decisions is a need to protect the investment in the system.

Petascale integration requires that all things be considered during the planning phases of all projects. The end game must be considered to determine what level of infrastructure investment needs to be made up front. Invariably, decisions need to be made on a "pay me now or pay me more later" basis. If later, the price could be considerably higher. Retrofit costs more due to many reasons, including removing older infrastructure, working around on-going operations and change to building codes. Costs also increase not just due to general inflation but to increases in construction materials that increased at 10-20% per year in recent years. For example, major power feeders and pipe headers should be installed up front to reduce costs later. Designating locations for future plumbing and conduit runs during initial design may require relocation of rebar or other components in the design, but this avoids having to drill through them in the future. Designing and building in a modular fashion, such as installing pads, conduit, etc. in advance, can also reduce future costs and disruptions. In summary, flexibility should be a very significant factor in the cost and effort of creating and re-designing these sites.

Serious consideration needs to be given to value engineering. When items are cut during design, it needs to be understood what the ramifications of the cuts are on the long-term costs of the facility. It has been proven that forward engineering yields significant cost savings down the road.

On the other hand, upgrading existing facilities typically involves a number of difficult retrofitting tasks. These include drilling, digging and jack-hammering; moving large water pipes; removing asbestos; increasing a raised floor. Currently, a raised floor of 36 inches is considered minimum for a large scale facility, with 48 inches (or more) preferred. Issues such as floor loading can be factors as newer, denser systems appear. Upgrading the facility, whether in terms of infrastructure or computing resources, can also tax users if existing systems in service are disrupted. An overlap of available systems, such as keeping an existing machine in service while an older one is removed to make way for a newer one, can provide uninterrupted access for users but means there needs to

be a buffer of facilities and services available (floor space, electrical power, cooling, etc.). An interesting question is what duty cycle should the facilities design target– 100 percent all the time? 100 percent duty cycle require a reduction in other parameters such as quality of service, reliability, utilization, time to transition new systems, etc.

There is a desire to see better interaction with vendors and facilities to determine and share the direction of new systems (shared knowledge base). One suggestion is to invite peer center personnel to review building design plans to get as much input as possible before and during design, which capitalizes on lessons learned and benefits all participants. Another recommendation is to have a computer technical facility present as part of the facility design and construction team during the entire process.

Environmental Issues

Power consumption and, conversely, power conservation by computing centers is becoming an increasingly important consideration in determining the total cost of ownership as HPC systems grow larger and more powerful. The group noted that new tools and technologies could help in this area. These include:

- Good CFD monitoring tools to better identify hotspots and other problems.
- Monitoring systems that are better integrated with the "controls" for responding to events, including a 3D sensor net monitoring system throughout the entire facility.
- Better tools for monitoring system hardware.
- Better tools for environmental monitoring on a finer grain basis.

Here are some additional observations/suggestions/issues concerning environmental designs.

Cooling

- Cooling office space using different (and possibly more efficient) systems than the machine room should be considered. Maintaining the environment in the machine room is paramount since the lifespan of systems can be reduced due to thermal events. Vendors typically can charge extra for issues stemming from poor environments.
- Liquid cooling will likely make a comeback for petascale systems, but some combination of air and liquid cooling will be required for all systems. In particular, it is unlikely that immersive cooling will return so memory storage and interconnects will remain air cooled in the future.
- Water cooling has an extremely low tolerance for cooling outages or fluctuations.
- Under-floor partitions or other design options should be considered when mixing systems with high and low cooling requirements in a common area.
- At some point, internals of air-cooled hardware will "melt down" if cooling/power is lost with no spinning fans to remove hot air. Technology and design has to ensure that chilled water continues flowing during these events.
- Chiller sequencing problems for redundant chillers during momentary and multiple momentary power interruptions can cause chillers to get into "loops"

where neither will start. Some solutions may be cold water reservoir systems, very large water header pipes and rotary systems.

Power

- 480V systems are coming. Facilities may need to make this a requirement for vendor bids to be considered due to prohibitive costs for copper and other power transforming components. There are issues with transient suppression at this level of voltage that do not currently have commercial solutions.
- Is AC and DC power distribution practical/possible? Many vendors not in favor of DC power for practical reasons.
- Commodity computing data centers (such as. Google's facilities) may be designed in a more modular fashion due to the uniformity of racks and components. Petascale facilities have fewer options in regard to generator and UPS backup power. Some group members expressed their UPS backs up only "single point of failure" components.
- Flywheel power conditioning was discussed. Reliability issues are a concern. Perhaps use of this technology for mechanical equipment is practical, but there is little formal study of this.
- There was a suggestion to move major network components onto UPS backup because routers and switches are non-tolerant of momentary power interruptions.
- Close the gap between systems and facilities staff at centers.
- Invite peer center personnel to review building design plans to get as much input as possible, which capitalizes on lessons learned and benefits all participants.
- Have a project manager from your facility participate in the entire planning process.

Charge to Breakout Group 2: Performance Assessment of Systems

There are many tools and benchmarks that help assess performance of systems, ranging from single performance kernels to full applications. Performance tests can be kernels, specific performance probes and composite assessments. What are the most effective tools? What scale tests are needed to set system performance expectations and to assure system performance? What are the best combinations of tools and tests?

Report of Break Out Group #2

Benchmarks and system tests not only have to evaluate potential systems for evaluation, but then must be able to ensure selected systems perform as expected both during and after acceptance. The challenges in this develop into three primary categories:

1.) Traditional performance metrics fail to adequately capture important characteristics that are increasingly important for both existing supercomputers and emerging petascale systems; such characteristics include space, power, cooling, reliability and consistency. However, the team noted that any new metrics must properly reflect real application performance, during both procurement and system installation. Precise benchmarking is

needed to emphasize the power consumed doing "useful work," rather than peak flops/watt or idle power.

2.) Although some researchers have had success with performance modeling, there remain significant challenges in setting reasonable performance expectations for broadly diversified workloads on new systems in order to realistically set accurate expectations for performance at all stages of system integration. This is particularly important when trying to isolate the cause of performance problems as system testing uncovers problems.

3.) There is a challenge associated with deriving benchmark codes capable of adequately measuring — and reporting in a meaningful way — performance in an increasingly diverse architectural universe that spans the range from homogeneous, relatively low-concurrency systems with traditional "MPI-everywhere" programming models, to hugely concurrent systems that will embody many-core nodes, to heterogeneous systems of any size that include novel architectural features such as FPGA. The key element in all of these is the need to have the metrics, tests, and models all be science-driven; that is, to have them accurately represent the algorithms and implementations of a specific workload of interest.

The Performance Breakout Group established a flowchart for system integration performance evaluation that called out a series of specific stages (beginning at delivery), goals at each stage, and a suggested set of tools or methods to be used along the way. Such methods include the NERSC SSP and ESP metrics/tests for application and system-wide performance, respectively, NWPerf (a system-wide performance monitoring tool from PNNL), and a variety of low-level kernel or microkernel tests such as NAS Parallel Benchmarks, HPC Challenge Benchmarks, STREAM, IOR, and MultiPong. A key finding, however, is that many sites use rather ad-hoc, informal recipes for performance evaluation and the community could benefit from an effort to create a framework consisting of a set of standardized tests that provide decision points for the process of performance debugging large systems. Additional mutual advantage might be gained through the creation and maintenance of a central repository for performance data.

Charge to Breakout Group 3: Methods of Testing and Integration

There is a range of methods for fielding large-scale systems, ranging from self integration, cooperative development, factory testing, and on-site acceptance testing. Each site and system has different goals and selects from the range of methods. When are different methods appropriate? What is the right balance between the different approaches? Are there better combinations than others?

Report of Break Out Group #3

The major challenges in the testing and integration area are as follows:

1. Contract Requirements Balancing contract requirements and innovation with vendors. If the vendor is subject to risks, the vendor is likely to pad the costs and the schedule. A

joint development model, with risks shared between the institution and vendor, may alleviate this but also places significant burdens on institutions. In order to be able to manage this and still reach their expectations, some institutions need more and different expertise. The cost of supporting this effort has to be borne by the institutions' stakeholders. An example of procurement with lots of future unknowns is NSF track 1, with delivery 4+ years in the future. Here the NSF is actually funding the institutions to accumulate expertise to alleviate some of the risk. Suggestions for meeting this challenge are:

- focus more on performance and less on detailed design description in RFPs
- replace specific requirements with engineering targets and/or flexible goals
- have go/no-go milestones for the engineering targets, at which point you negotiate the next round of targets (this is especially useful if the vendor is in a development cycle with their product) or the system is multi-phased.

2. Testing: Vendors will never have systems as large as those installed at centers; so much of the testing will happen at sites. At the very least, however, vendors should have an in-house system that is 10-15 percent of the size of their largest customer's system. Suggestions for meeting this challenge:

- Vendors will need time on your machine. For example, at ORNL, Cray gets their full XT4 every other weekend, and half of the system on the alternate weekends.
- Need a way to swap in and out between the production environment and the test environment (ask the vendor for proof of the ability to do this) in an efficient manner.
- Share system admin tests across sites (just as we've started to share benchmarks) in the areas of:
    - swapping disks
    - managing large disk pools
    - managing system dumps
    - booting process
    - job scheduling (note: need to include job scheduling in integration testing)
    - NERSC ESP (http://www.nersc.gov/projects/esp.php)
- Need for at least one, if not multiple, development systems (one needs to be identical to the production system; ideal to have different development systems for systems and application work) that are sufficient in size and scope to be representative of a full system.

3. Lead times for procurements: This makes it difficult to develop performance criteria, as well as predict what performance will be. A process is needed to determine performance and milestone dates for solidifying the specifications. Another problem is that the longer a system stays in acceptance testing mode, the higher the cost both to the vendor and the site. Suggestions for meeting this challenge:

- Negotiate a milestone approach rather than one monolithic acceptance test
- In some cases, it may be useful to procure subcomponents and accept them separately, then federate them, perhaps allowing vendors to recognize revenue

earlier. However, this approach should be used carefully, as it may be in conflict with Sarbanes-Oxley rules and also blurs lines of responsibility between which organization if responsible for performance and reliability.

4. Debugging problems at scale: For both the scientific users of systems and the system managers, many problems are not identified or detectable at smaller scales. This is particularly true for timing problems. For the users, the number of tools is limited (two debuggers exist for programs at scale) and are complex to use. For system issues, there are no debugging tools, and often custom patches are needed to even get debugging information. Issues to consider:

- How to do fault isolation?
- How do you debug problems at scale when they only show up intermittently or after long run times?
- How to do quality control and consistency at scale (you can get inconsistent performance due to different batches of components)?

Suggestions to meet this challenge include

- Fund reliability and root cause analysis studies, particularly for system software. Finding root causes is typically difficult and expensive – requiring large scale resources at times. Specialized "deep dive" expertise might be established within the community – maybe at multiple sites – that can be deployed specifically to deal with such root cause analysis.
- Create new technology and more competition in the area of debugging tools. For many years, only one tool existed, and most vendors got out of the business of application debugging. The approaches that worked at 10-100's processors become very difficult at 1,000 processors and impossible at 10,000 processors. New funding for debugging tools – both application and system – is critical.
- Funding to accumulate error information is important. It was noted that the traditional approach of sharing error information and solutions between sites is not working any more. This is in part because most sites have multiple vendors, yet see similar problems and because much of the software now in place is horizontally integrated. Hence, funding the effort to create, maintain and refine repositories of information is important.
- Use the influence of the general community to encourage/require the sharing of problems across sites with systems from the same vendors. Most vendors keep problem reporting private and often multiple sites have to find and troubleshoot the same issue.

5. Determining when a system is ready for production: In some ways, "production quality" is in the eye of the beholder. What is useful and reliable to one application or user may not be to another. Issues include determining when to go from acceptance testing to production (suggest that acceptance testing only deal with systemic problems, not every last bug), and determining who is responsible for performance/stability when multiple vendors have different pieces.

A related issue is how to manage the responsibilities of having systems created and built by multiple vendors. Very few vendors have complete control over all the technology in their systems. Suggestions to meet this challenge:

- Use contracts that require the vendors to cooperate and that their piece has to work. Approach the system as a partnership (e.g., ORNL has a contract with CFS independent of the contract with Cray).
- Need more risk management and contingency strategies explicitly included in system planning and management. With multiple vendors, sites run the risk. Many sites explicitly decide to become the integrator and assume some of the risk. There is a need to share more information about this as to the boundaries, management and contractual arrangements that work best. *(Perhaps this can be a subject of the next workshop.)*

6. What tools and technology do we wish we had? At busy sites, with multiple requirements, there are always things that would be good to have. Although these tool requirements are too much for any one vendor to provide, the HPC community could look to universities and laboratories to develop the frameworks that vendors can then plug into. All these areas are good candidates for stakeholder funding, even though they may not be "research". Some of these are:

- Ability to fully simulate a system at scale before building it — better hardware and software diagnostics
- Hardware partitioning so that one partition cannot impact another partition
- Ability to virtualize the file system so different test periods do not endanger user data
- System visualization — a way to see what the whole system is doing (and display it remotely)
- Support for multiple versions of the operating system and the ability to quickly boot between them — tools that verify the correctness of the OS and the integrity of system files and that do consistency checking among all the pieces of the system
- System backups at scale and at rates that make backup tractable
- More holistic system monitoring (trying to see the forest despite the trees)
- Consolidated event and log management and the ability to analyze logs and correlate events. This must be provided in an extensible open framework.
- Parallel debugger that works at scale
- Better tools for dump analysis
- Parallel I/O test suite
- Ability to better manage large numbers of files
- More fault tolerance and fault recovery

Charge to Breakout Group 4: Systems and User Environment Integration Issues
Points to consider: Breakout session #2 looked at performance and benchmarking tools. While performance is one element of successful systems, so are effective resource management, reliability, consistency and usability, to name a few. Other than performance, what other areas are critical to successful integration? How are these evaluated?

Report of Break Out Group #4
The major areas addressed by the group included:

- Effective Resource Management
- Reliability
- Consistency
- Usability

1. Resource Management: Systems and system software remain primarily focused on coordinating and husbanding CPU resources. However, many applications and workloads find other resources equally or more critical to their effective completion. These include coordinated scheduling of disk space, memory and locations of CPUs in the topology. Resource management and human management of petascale disk storage systems may exceed the effort to manage the compute resources for the system. We need more attention to the implementation of disk quotas (and quotas in general) to manage data storage. Likewise, future topologies that have more limited bandwidth may benefit from tools such as job migration to help with scheduling and fault tolerance.

Considerations:

- With current schedulers, can the sites effectively create policies that meet their user needs?
- On large systems, what proportion of the resource should be allocated to the development-size (small) jobs for the user community?
  - Users are pushed to take advantage of the very large system, perhaps to the detriment of developmental jobs.
  - Integration of batch and interactive scheduling

2. Reliability: Reliability is a significant concern for petascale systems not only because of the immense number of components, but also because of the complex (almost chaotic) interactions of the components. Discussion focused on three areas: event management, RAS (remote access service) and system resiliency for the issues and suggestions.

Recording and managing events deal with acquiring and efficiently understanding the complexity of the systems. There is a critical need for a uniform framework for recording and analyzing events. This framework, which generated a significant discussion at the workshop, might possibly use API/XML tags for easier data sharing and analysis. Technical issues for such a framework include questions of granularity (core, CPU, node, or cluster) and who defines the levels of granularity (sites, vendors, etc.) The

goal is to coordinate events that happen across multiple systems/subsystems. For example: correlate batch job logs with other system events. Both asynchronous and synchronous event polling (proactive and reactive) need to be considered.

This area may also be a topic for a future workshop.

Suggested tools/technologies for this area include:

- Monitoring at appropriate levels (avoid overwhelming volumes that tend to be ignored)
- Developing tools that help process the large volumes of data to help forecast, detect, and provide forensic support for system anomalies. These tools would enable
  - Root cause analysis tools
  - Critical event triage tools
  - Statistical analysis tools for analyzing events
    - Failure prediction
- Resource tracking to have a mechanism for feeding these new resource management requirements back to the vendors.

-

Reliability, Availability and Serviceability (RAS) system: How much complexity and level of effort is introduced by the RAS system for these large systems? RAS systems have introduced many issues in large-scale systems and have taken significant effort to diagnose. On the other hand, they seem to play a key role in all system design. Issues about RAS being too complex, onerous and bulky are important to explore.

- In some cases, RAS subsystems have been observed to cause more problems than they solve. In other cases, complex, hardware-oriented RAS systems have been subverted by software that does not match the same level of responsiveness. Are extensions to current RAS systems adequately designed for petascale size systems?
- Most formal RAS study and most common RAS features are based on hardware. Maybe because of the success RAS oriented hardware design features, systems often have the majority of their critical system-wide failures caused by software. Unfortunately, there is little data and less study of software failures in petascale systems.
- The major issue is what RAS features are required for petascale size systems? Conversely, what features work only at smaller scale, and can or should be discounted/discarded at larger scales? Again, there is a critical need to understand the software issues in addition to the hardware.
- Disk RAS is separate from the system RAS. Problems that affect one subsystem may affect the other system. With no integration, the identification of the error is much more complicated.

Suggestions for addressing these issues:

- Root cause analysis that would efficiently respond to complex queries such as "What event triggered the other 10,000 alarms?"
- Correlation of events is very difficult given all currently disparate RAS systems (disk, network, nodes, system hardware and software)
- More holistic system monitoring tools are necessary because fine grain monitoring (e.g. per core) would over whelm system managers with information.
- Correlation of performance events with reliability events

System Resiliency - From the system and performance perspective, it is best to keep the compute node simple. This means few or none of the features seen in commodity or even special servers such as virtual memory with paging. When systems lose components, can they degrade gracefully? At what impact to the users? PNNL is attempting to identify failing components prior to loss (mitigation strategy), but this approach requires job migration which is not a common feature in today's software world. A different approach would be to allocate a larger node or CPU pool than necessary so that if a node is lost, that work can be migrated to a 'spare' node.

Resiliency from the user perspective includes the tools/techniques the users need to create more fault-tolerant applications. Today, our applications are very fragile, meaning that the loss of one component causes the loss of all the work on all the components. This causes applications to put stress on other resources, such as when applications do "defensive IO" in order to checkpoint their work.

There are methods to allow applications to survive component outages and recover, such as journaling, work queues and duplication of function. None of these are used in HPC since they sacrifice performance for resiliency. Is it time at the petascale to change applications to do this?. Will CPUs be so cheap, that it becomes feasible? How do we educate the users to write more fault-tolerant applications? Can you reconstruct lost data from nearest neighbors?

Suggestions to address resiliency issues are:

- Fund research for new applications and algorithms that not only scale but also improve resiliency.
- Accept lower performance on applications to improve resiliency

Consistency: Large-scale systems can produce inconsistent results in terms of answers and run times. Inconsistency negatively impacts system effectiveness and utilization. From the system perspective, it becomes very expensive (and maybe impossible) for vendors to keep a completely homogeneous set of spare parts. When components are replaced, for whatever reason, the replacement frequently is slightly different (lot, firmware, hardware).

Consistency is also desired in the user environment. When a science code behaves differently the reasons may not be clear. It could be due to compiler changes, changing libraries or jitter in the system. If the user mistakenly guesses the run of a job incorrectly, it may take longer to be scheduled or, worse, it may be killed for going over its time limit. Either way, the user's productivity is undermined.

Hence suggestions are:

- Sites may have to develop new maintenance strategies since it may not be possible to prevent heterogeneity of the system components over its life span.
- Configuration management tools to manage this component-level knowledge base, but which will be executed on a site-by-site, case-by-case basis, will be critical to understanding and predicting inconsistency
- Methods to allow a site to return to a previous state of the system will benefit users and sites once inconsistency is detected. This would have broad impacts across kernel, firmware upgrade processes that are currently used.

Usability: System reliability and environmental stability directly affect the perceived "usability" by the user community. Other features are necessary for a large-scale system to be usable by the science community. Access to highly optimizing compilers and math libraries at the petascale is key. Good parallel debugging tools are also important, along with integrated development environments such as the eclipse parallel tool kit platform. Essentially, tools that will help users understand job status and workflow management (When will it run? Why did it abort?, etc.) are needed.

A system with good performance balances will be important for supporting a range of science. This includes memory size and bandwidth, interconnect bandwidth and latency and the number of threads or concurrency supported. Consistency of the environment across multiple systems (scheduler, file systems) contributes to usability.

Resource Management: Systems and user environments have not made much real progress in terms of system resource/system management software over the last 30 years. The scarcity of CPUs is no longer the issue for scheduling, but systems still focus on that. Rather, bandwidth, disk storage, and memory use are the limiting factors for Petascale. Even monitoring usage in these areas is difficult, let alone managing it. Can schedulers be made aware of machine-specific features that will impact performance of the code? XT series machines are a specific example. The location of the code on the machine will impact its performance because of bandwidth limitations in the torus links.

Job scheduling logs are underutilized. Error detection tools should be able to correlate system failures with jobs. There should be better ways to separate these job failures from user error or system issues. Job failure due to system failure should be calculable. In-house tools are being used at some sites to try to correlate batch system events with system events. This is limited and the algorithms are not sophisticated. For example, research has shown system failures in large systems often have precursor symptoms, but sophisticated analysis is needed to detect them. Little work has been funded in this area

and few tools are available that may allow adaptive behavior (e.g., they don't schedule work on that job, etc. until diagnostics determine failure cause). There are many issues, including how to deal with false positives. Managing multi-thousand-node log files is a challenge due to the sheer volume of data.

Research has given interesting hints of what may be possible. For example, statistical learning theory was used to analyze http logs and was shown to detect pending failures well before they occurred. Statistical methods are deployed by Google and Yahoo to address reliability and adaptive systems requirements.

Hard drive failure analysis is another area that offers substantial research issues, as well as practical issues to use in real environments. What forewarning technology exists? Will virtualization assist in any of the management issues?

Suggestions for these issues include:

- The community needs a tool that can analyze the logs, perhaps drawing from tools like LBNL's Bro, which can analyze network traffic for anomalies that indicate attacks, etc. It is not possible for any single operational site or vendor can fund such research on their own, and new approaches are needed.
- There are basic research questions in the areas of artificial intelligence, data management, operating systems, statistics, reliability research, and human factors that need to be addressed before useful tools can be developed.
- Understanding failure modes is not well funded in large-scale computing. There is some funding in the Petascale Data Storage Institute – a SciDAC project – but that is mostly focused on data.

Best Practices: The group developed a list of best practices that many sites use to address the challenges in the reliability arena. These include:

- Have a non-user test/development system that matches the hardware configuration of any large system. It can be used for testing new software releases, doing regression testing, and exploring problems. It is important the test/development system be identical in HW to the full system, as well as having all the same configuration components. All the software components and layers are needed as well, albeit periodically at different versions.
- Trend analysis is important. One example of a problem cited was a system that showed performance degradation at 5 percent per month since reboot. This took a long term trend analysis effort to detect.
- Establish reference baselines of performance and services at the component level. This allows proactive testing and detection of anomalous conditions with periodic consistency checking at the component level.
- Proactively check performance over time. The time periods and testing vary, but many problems that are difficult to detect early in the general workload have been found with proactive performance testing in a consistent (i.e., automated) manner.

- Perform regression testing for all significant changes – both before the change on the test system and during system time, and after the change. (see Breakout #5 as well).
- File systems should be implemented with multipath I/O connections and redundant controllers. RAID arrays are almost mandatory for any large file systems. As the number of components in storage systems continues to increase it may be that RAID-5 is no longer sufficient.
- Multipath power connections to computing, storage and networking equipment from two independent panels or PDUs, even without UPSs, improves the ability for a site to continue operation through standard facilities repair and changes.
- Proactive memory resource management

System Deployment: Do not deploy subsystems until they are actually ready for deployment and the expected workload. The pressure for some use of new systems should be balanced by the need to provide a quality experience for the early science community. Some metrics or parameters for successful integration may include:

- Are the users happy with the tools that are available to them to assist with their work?
- Middleware must be tested as it is expected to be used. For example, schedulers must be tuned to the requirements of the user community. Are there repeatable run times, queue wait times, etc. on the system?
- System management and user environment: Are there site-specific tools available that will help the users effectively use the system and understand problems?
- System Balance: What is the compute versus file system performance, network performance, etc.?
- CPU scheduling is not integrated with disk scheduling. There's never enough temp space, and there's never enough bandwidth to scratch storage. Can users accurately forecast their temporary space requirements, when they still have difficulty forecasting their run time/CPU requirements?

Charge to Breakout Group 5: Early Warning Signs of Problems – Detecting and Handling

Fielding large scale systems is a major project in its own right, and takes cooperation between site staff, stakeholders, users, vendors, third party contributors and many more. How can early warning signs of problems be detected? When they are detected, what should be done about them? How can they be best handled to have the highest and quickest success? How do we ensure long-term success versus the pressure of quick milestone accomplishment? Will the current focus on formal project management methods help or hinder?

Report of Break Out Group #5

Each system is unique and may consist of many components that must all be successfully integrated, perhaps at a scale never before attempted. It may be built on-site without a factory test. Setting realistic schedules and meeting milestones can pose challenges. Risk assessment and management plans must be developed to deal with the uncertainties inherent in installing novel large-scale systems.

It is important to quickly detect and correct problems that arise during system integration. The method for doing so can be somewhat of an art — "something doesn't feel right" — based on experience. More well-defined procedures are desired, but the community does not have broad experience with formal project management procedures and tools. Project management processes have been imposed by funding agencies and are likely to remain. However, these have not yet proved to be effective tools for detecting and handling early warning signs of problems in the realm of HPC. It is not clear what are the most effective modifications to traditional PM methods to accommodate the needs of HPC.

Suggestions for addressing these issues include the following:

- A good management plan that measures overall progress is needed to guard against the perception that progress is being made solving day-to-day problems, while the number of problems is not decreasing. The plan should be able to retire risks systematically. The plan must have buy-in from sites, vendors and funding agencies. Paths for escalating problems to higher levels of management should be explicit and agreed upon by all parties. It has proven valuable to have site and vendor owners for identified major risks.

- Detailed tracking of individual components and failures, along with root cause analyses, can help identify problems and ensure component consistency across the system. Testing the functionality and performance of scientific applications should be part of the standard system tests and acceptance plans. The integration plan needs to contain an adequate number of agreed-upon milestones to ensure early detection of problems.

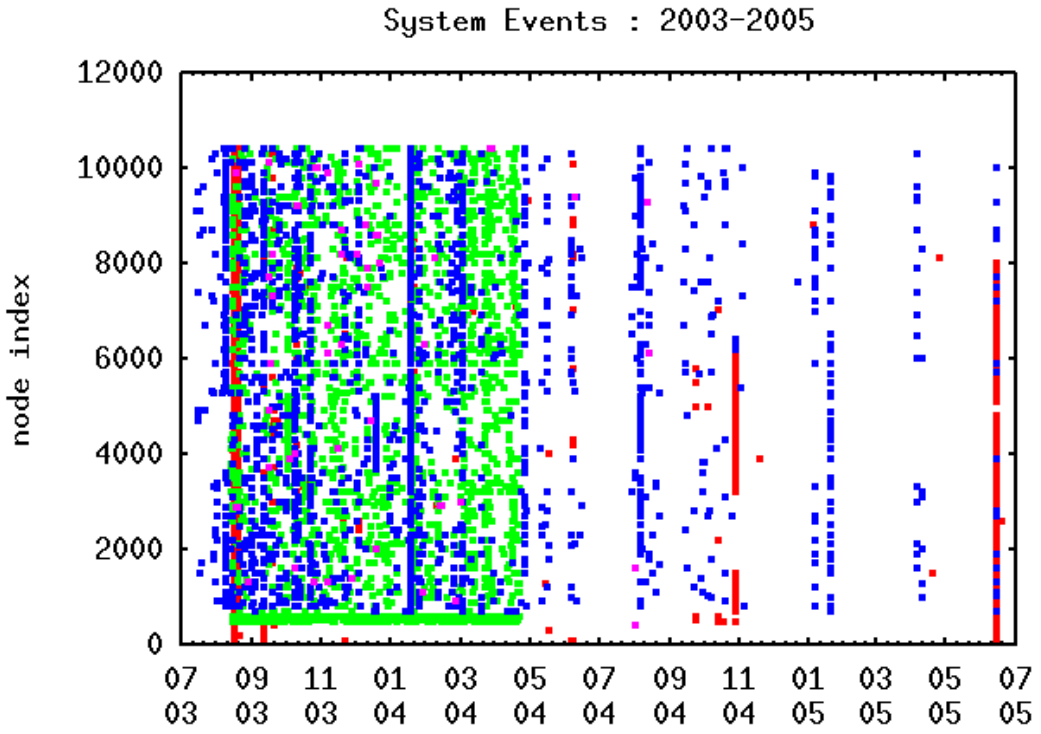
- While each installation is unique, it is valuable for sites to share experiences. Sites can exchange representatives to observe and advise during integration. (It should be possible to deal with issues if the plans are proprietary.) Outside review committees can be used. Workshops or community events sponsored by funding agencies can facilitate information-sharing, including making this workshop an ongoing event. An online community site would be useful.

- A hierarchy of project review meetings can serve as checks at different levels within organizations. Meetings between site and vendor technical staff should be encouraged, separate from management meetings.

With current revenue-reporting rules, there may be conflicts between vendors, who want a system accepted before a fiscal year deadline and under Sarbanes-Oxley rules, and sites, who want a well-tested and validated machine. The project plan should try to mitigate these conflicts as much as possible ahead of time and ensure success for all parties. Having a plan that tracks overall progress is important in this context.

Charge to Breakout Group 6: How to Keep Systems Running up to Expectations. *Once systems are integrated and accepted, is the job done? If systems pass a set of tests, will they continue to perform at the level they start at? How can we assure systems continue to deliver what is expected? What levels and types of continuous testing are appropriate?*

An important first step in addressing sustained performance of complex HPC resources is to first realize that performance is often not static. The following figure shows performance and failure events over two years for a 6000 CPU machine. The vertical axis (node index) spans the extent of the machine, events are color coded corresponding to performance degradation (blue and green) or node failure (red). The frequency and extent of such events depends strongly on attention to monitoring and resolving events which impact the sustained performance of an HPC resource.



The topics of discussion in Breakout Group 6 were how to detect such changes in system performance and maintaining system health and integrity. All of the sites represented

conduct some form of regression testing after upgrades to system software or hardware. Most of the sites do periodic performance monitoring using a defined group of benchmarks. Some of the sites do system health checking either at time of job launch or on a periodic basis via cron commands. The degree to which these measures are implemented varies greatly. Maintaining performance histories is an important step towards providing reliable performance as well as leveraging performance reliability knowledge between HPC centers.

Several challenges to sustaining the performance of HPC resources were identified and include:

- Equating machine performance with peak performance
  - The community is almost over this misleading approach
- Lack of integrated SW change control.
- Constrained resources, particularly staff time and system resources, devoted to identifying and resolving performance issues
- Petascale component chaos where minor change in one component yields major unexpected impact on seemingly unrelated components.
  - Needle in haystack issues, performance sleuthing
  - Scaling of data analytics on performance data and health logs
- Means of motivating vendors to correct performance degradation post-acceptance?

Currently various levels of performance monitoring are in use by HPC centers. The scope of those methods and specific implementations in parenthesis are listed below:

- Ongoing monitoring of node health via an external interface (NAGIOS)
- Per batch job application performance (HPMstat-NCAR, IPM-NERSC, NWPERF-PNNL)
- Center initiated benchmark runs (SSP-NERSC)
- Ongoing pre-execution perf tests (node, interconnect, filesystem)- PSC
- Using intelligent syslog methods instead or in addition to log files.

A major issue for petascale systems is how to manage the information collected by system monitoring such as the above methods. The amount of information generated by system logging and running diagnostics is hard to manage now – if current practices are continued, data collected on a petascale system will be so voluminous that the resources needed for storing and analyzing it will be intractable

The participants in the breakout session started developed a draft of best practices for continuous performance assessment including suggested techniques and time scales. Suggestions to help address these issues include:

- Identified needs are standardizing and intelligently filtering system log data, an API or standard data format for system monitoring, and the development of tools for analyzing and displaying system performance and health using the standard formats and API.
- Performance tools that implement abstractions which provide simple performance profiles across many nodes/tasks. Methods of analyzing performance which circumvent dealing with problems on a per node/task basis. For petascale, solving classes of problems become more valuable than a list of the locations of problems.
- Application, microbenchmarks, and component level tests to confirm enduring performance are regularly conducted. Increased specificity as to resource performance.
- Tying the evaluation of efforts on the part of centers and vendors to resolution of dynamical changes in performance as seen by HPC customers.

List of Attendees

Bill Allcock, Argonne National Laboratory
Phil Andrews, San Diego Supercomputer Center/UCSD
Anna Maria Bailey, Lawrence Livermore National Laboratory
Ron Bailey, AMTI/NASA Ames
Ray Bair, Argonne National Laboratory
Ann Baker, Oak Ridge National Laboratory
Robert Ballance, Sandia National Laboratories
Jeff Becklehimer, Cray Inc.
Tom Bettge, National Center for Atmospheric Research (NCAR)
Richard Blake, Science and Technology Facilities Council
Buddy Bland, Oak Ridge National Laboratory
Tina Butler, NERSC/Lawrence Berkeley National Laboratory
Nicholas Cardo, NERSC/Lawrence Berkeley National Laboratory
Bob Ciotti, NASA Ames Research Center
Susan Coghlan, Argonne National Laboratory
Brad Comes, DoD High Performance Computing Modernization Program
Dave Cowley, Battelle/Pacific Northwest National Laboratory
Kim Cupps, Lawrence Livermore National Laboratory
Thomas Davis, NERSC/Lawrence Berkeley National Laboratory
Brent Draney, NERSC/Lawrence Berkeley National Laboratory
Daniel Dowling, Sun Microsystems
Tom Engel, National Center for Atmospheric Research (NCAR)
Al Geist, Oak Ridge National Laboratory
Richard Gerber, NERSC/Lawrence Berkeley National Laboratory

Sergi Girona, Barcelona Supercomputing Center
Dave Hancock, Indiana University
Barbara Helland, Department of Energy
Dan Hitchcock, Department of Energy
Wayne Hoyenga, National Center for Supercomputing Applications (NCSA)/University of Illinois
Fred Johnson, Department of Energy
Gary Jung, Lawrence Berkeley National Laboratory
Jim Kasdorf, Pittsburgh Supercomputing Center
Chulho Kim, IBM
Patricia Kovatch, San Diego Supercomputer Center
Bill Kramer, NERSC/Lawrence Berkeley National Laboratory
Paul Kummer, STFC Daresbury Laboratory
Jason Lee, NERSC/Lawrence Berkeley National Laboratory
Steve Lowe, NERSC/Lawrence Berkeley National Laboratory
Tom Marazzi, Cray Inc.
Dave Martinez, Sandia National Laboratories
Mike McCraney, Maui High Performance Computing Center
Chuck McParland, Lawrence Berkeley National Laboratory
Stephen Meador, Department of Energy
George Michaels, Pacific Northwest National Laboratory
Tommy Minyard, Texas Advanced Computing Center
Tim Mooney, Sun Microsystems
Wolfgang E. Nagel, Center for Information Services and High Performance Computing ZIH, TU Dresden, Germany
Gary New, National Center for Atmospheric Research (NCAR)
Rob Pennington, National Center for Supercomputing Applications (NCSA)/University of Illinois
Terri Quinn, Lawrence Livermore National Laboratory
Kevin Regimbal, Battelle/Pacific Northwest National Laboratory
Renato Ribeiro, Sun Microsystems
Lynn Rippe, NERSC/Lawrence Berkeley National Laboratory
Jim Rogers, Oak Ridge National Laboratory
Jay Scott, Pittsburgh Supercomputing Center
Mark Seager, Lawrence Livermore National Laboratory
David Skinner, NERSC/Lawrence Berkeley National Laboratory
Kevin Stelljes, Cray Inc.
Gene Stott, Battelle/Pacific Northwest National Laboratory
Sunny Sundstrom, Linux Networx, Inc.
Bob Tomlinson, Los Alamos National Laboratory
Francesca Verdier, NERSC/Lawrence Berkeley National Laboratory
Wayne Vieira, Sun Microsystems
Howard Walter, NERSC/Lawrence Berkeley National Laboratory
Ucilia Wang, Lawrence Berkeley National Laboratory
Harvey Wasserman, NERSC/Lawrence Berkeley National Laboratory

Kevin Wohlever, Ohio Supercomputer Center
Klaus Wolkersdorfer, Forschungszentrum Juelich, Germany

List of priorities from attendees based on a ranking algorithm.

Description	Summary ranking
Provide methods for regular interaction between peer centers to avoid potential pitfalls and identify best practices.	187
Consolidated event and log management with event analysis and correlation. Must be an extensible, open framework. Common format to express system performance (at a low level), health, status, and resource utilization (e.g. <machine name="“franklin”>; <cab	142
Tools for ongoing “Intelligent” syslog / data reduction and analysis.	123
Develop methods to combine machine and environment monitoring.	113
Ability to have multiple versions of the OS that can be easily booted for testing and development. Ability to do rolling upgrades of the system and tools. Can you return to a previous state of the system? Very broad impacts across kernel, firmware, etc.	113
Develop standard formats and ability to share information about machines and facilities (Wiki?). Community monitoring API (SIM?)	112
Develop better hardware and software diagnostics	108
Develop tools to track and monitor message performance (e.g. HPM/PAPI for the interconnect and I/O paths, hybrid programming models)	105
Create better parallel I/O performance tests and/or a Parallel I/O test suite	101
Funded Best Practices Sharing (Chronological list of questions and elements of project plan, Top 10 Risk Lists, Commodity equipment acquisition / performance)	99
Better ways to visualize what the system is doing with remote display to a system administrator for more holistic system monitoring.	96
Visualization and Analytics tools for log and performance data. Tool that can analyze the logs, perhaps drawing a parallel to Bro, which can analyze network traffic for anomalies that indicate attacks, etc	90
Invite peer center personnel to review building design plans to achieve as much input as possible and so reviewers can benefit as well.	86
Tools for storage (Disk) Systems management and reliability	85
Develop/identify computer facility modeling and analysis software (Air flow, cooling system, etc. – e.g. Tileflow)	81
Develop automated procedures for performance testing	81
Share problem reports with all sites – a vendor issue rather than a site issue.	80
Improved parallel debugger for large scale systems including dump analysis	77
Develop tools to verify the integrity of the system files and to do consistency checking among all the pieces of the system.	67
Develop accurate methods for memory usage monitor/OS intrusion	65
Tools to monitor performance relative to energy power draw	63
Failure data fusion and statistical analysis	60
Develop realistic interconnect tests	59
Scalable configuration management	58
Job failure due to system failure should be calculable. In house tools are being used in some cases to try and correlate batch system events with system events.	57
Share (Sanitized) Project Plan experience among sites as part of project closeout – RM	56

activities required	
Implement facility sensor networks in computer rooms including analysis and data fusion capabilities	55
Develop accurate performance models for non-existent systems including full system including I/O, interconnects, software	55
Develop tools to measure real Bit Error Rate (BER) for interconnects and I/O	52
Developing improved methods of backing up a systems of this size	52
Develop *matrix* to correlate the four major facility categories (Space, Power, Cooling, Networking) with each of these phases of a facility for Greenfield (new), Retrofit, Relocate, Upgrade for planning purposes.	48
Create the ability to fully simulate a full system at scale without having to build it (e.g. UCB RAMP)	46
Develop ways to receive real data for power consumption from vendors	42
Hard partitioning of systems so one partition cannot impact another. Virtualized the I/O partition so that it can be attached to multiple compute partitions.	42
Have external participation in proposal and plan reviews	42
Fund studies about systemic facility design (What voltage? AC or DC? CRAC Units, Etc.) including the level of detail needed for monitoring?	41
Statistical analysis of logs can detect pending failures. Is deployed by Google, Yahoo to address reliable and adaptive systems requirements. Currently based on http:, but may be extensible to this situation.	40
Coordinate scheduling of resources in addition to CPUs	39
Share WBS, RM, Communications Plans, etc. among sites	37
Improve Project Management expertise in organizations	36
Create a framework with general acceptance by most of the community consisting of a set of tests that provide decision points on the flow chart for debugging	17
Benchmarks in new programming models (UPC, CAF, …)	17
Reporting that shows just the important (different) areas in the machine	15
Develop the methods and roles for statistician with PSI projects	6
Configuration management tools can manage this component-level knowledge base, but will be executed on a site-by-site, case-by-case basis.	2

Agenda

May 15, 2007 – 8 am start

—First Session - Plenary

—Introduction and Logistics – Bill Kramer/Yeen Mankin

—Welcome – Dan Hitchcock

—Motivation for the Workshop – Bill Kramer

- System Integration at LLNL – Mark Seager
- System Integration at NCAR – Tom Bettge

- Break

- Second Session – Breakouts (more later)

- Third Session – Plenary

—Reports from breakouts

—Panel – The Vendor Side of Deployments : Jeff Beckelhimer - Cray, Chulho Kim - IBM, Wayne Vieira - Sun, Dave Sundestrom - Linux Networx

- Working Dinner

- Panel of the Whole –

—If only I had known! –the biggest blunders/mistakes and humorous experiences in large system deployments – All

- Day 2 - May 16, 2007 – 8 am start

- Fourth Session – Breakouts

- Fifth Session – Plenary

—Reports from breakouts

—Panel Session – How will Petascale systems change what we have been doing? - Ray Bair (ANL), Patricia Kovatch (SDSC), Brad Comes (DODMod), Bob Ciotti (NASA)

- Sixth Session – Plenary

—Report Summary

- Conclusion