

# A Failure Predictive and Policy-Based High Availability Strategy for Linux High Performance Computing Cluster

Chokchai Leangsuksun<sup>1</sup>, Tong Liu<sup>1</sup>, Tirumala Rao<sup>1</sup>,  
Stephen L. Scott<sup>2</sup>, and Richard Libby<sup>3</sup>

<sup>1</sup>*Computer Science Department, Louisiana Tech University  
Ruston, LA 71272, USA*

<sup>2</sup>*Computer Science and Mathematics Division, Oak Ridge National Laboratory  
Oak Ridge, TN 37831, USA*

<sup>3</sup>*Enterprise Platforms Group, Intel Corporation*

<sup>1</sup>*{box, tli001, trb013}@latech.edu*

<sup>2</sup>*scottsl@ornl.gov*

<sup>3</sup>*rml@hpc.intel.com*

## Abstract

The Open Source Cluster Application Resources (OSCAR) is a fully integrated cluster software stack designed for building, and maintaining a Linux Beowulf cluster. As OSCAR has become a popular tool for building the cost effective HPC cluster, undoubtedly, High Availability (HA) will equally be an important aspect that enables HPC systems, as clearly an unavailable cluster equals no performance. To embrace both HA and HPC features, we created the HA-OSCAR solution which eliminates the numerous single-point-of-failure in HPC systems and alleviates unplanned downtime through sophisticated self-healing mechanisms and component redundancy. In this paper, we report our newly introduced ideas and experiments on hardware level failure detection and prediction based on the Service Availability Forum's Hardware Platform Interface (OpenHPI). Service monitoring and Policy-based head node recovery is also discussed in detail. Furthermore, we investigate a Network File system issue during server failover and resolution via the High Reliable Network File System (HR-NFS), without the need for an expensive hardware-based shared storage solution. Furthermore, our solution enables a graceful recovery with a deliberate job checkpointing and migration upon head node failure prediction. Finally, we introduce our web-based management module that provides a customizable service monitoring, recovery/failover management mechanism with an effective cluster monitoring ability.

**Key words:** *High Performance Computing, High-Availability, Linux cluster, Predictive Failure Analysis, High-Reliability, IPMI.*

---

<sup>1</sup> Research supported by Center for Entrepreneurship and Information Technology, Louisiana Tech University.

<sup>2</sup> Research supported by the Mathematics, Information and Computational Sciences Office, Office of Advanced Scientific Computing Research, Office of Science, U. S. Department of Energy, under contract No. DE-AC05-00OR22725 with UT-Battelle, LLC.

## **Introduction and Motivation:**

Due to the performance needs of ultra-scale and long-running applications, cluster size continues to increase dramatically. However, a linear increase of HPC cluster size results in an exponential failure rate. Typically, the HPC cluster structure is comprised of a single head node and multiple compute nodes. There are a number of techniques available today that will permit a cluster to continue operation should compute nodes fail. “Continued operation” is defined here, as the cluster is still responsive to user requests. It is important to note that a “responsive cluster” does not ensure the continued unfettered execution of a currently executing application. Thus, while a cluster may continue to operate throughout a compute node failure, an executing application may fail. Other research areas such as checkpoint/restart address the continued execution of an application during computation node failure[4]. This is not the case for the typical cluster head node. Generally when the HPC cluster head node fails, the entire cluster is brought to a standstill. Without a head node to dispatch jobs and manage the computation nodes, the compute nodes will be like the body separated from its brain. Thus grew the realization for the need of combining High Availability and High Performance computing resulting in the High Availability-OSCAR (HA-OSCAR) effort. With a dual head architecture, HA-OSCAR aims to reduce cluster system downtime [5][6][7] and therefore improve service availability and system manageability for the Beowulf class architecture.

The goal of the HA-OSCAR project is to provide high-availability capability to the high-performance computing market. The development approach is to leverage the existing OSCAR technology while extending it to include high-availability capabilities to Beowulf architectures. Following the OSCAR objectives for easy installation and management, HA-OSCAR provides an easy-to-install package with a GUI based interface to build the Linux-based HA Beowulf cluster. High availability is typically not a feature of a HPC software stack, thus its inclusion would usually a tedious manual package installation/configuration cycle.

In addition, HA-OSCAR brings a sophisticated recovery/failover strategy which allows users to add their own monitored components to our enhanced MON®’s [8] framework. Our failover methodology uses a network cloning strategy, which guarantees both head nodes are identical so that the secondary node may impersonate the primary during a failure. This strategy differs slightly from normal IP-alias failover [9]. Moreover, failover is not only based on a system crash or service failure but also through a user configurable threshold with failure predictive analysis ability. Our Data Gathering and Analysis Engine (DGAE) establishes mechanisms to gather sensor data and events of various hardware, evaluates that information with pre-defined principles, and derives statistical result to our policy-based failover management module. Analyzed events are dealt with via recovery module by initiating associated alarms or triggering an appropriate handler; e.g. a failover event with application checkpointing and real-time data synchronization. Our goal is to provide a powerful and flexible mechanism to ensure minimum interruptions.

### A Failure Predictive and Policy-Based High Availability Strategy 3

In recent years, more and more HPC cluster nodes exploit their intelligent hardware platform interface (IPMI) that allows for better hardware management and more accurate hardware information access (e.g. Intel IPMI-based server [12], IBM Blade Center [13], etc.). We leverage these state-of-the-art platform interfaces by enhancing our High Availability and Performance Computing (HAPC) solution to support those capabilities such as temperature, voltage, fans, power supplies and chassis monitoring, hot-swap support and hardware-based remote system management in addition to power control (on / off / reset) via a universal hardware platform interface (OpenHPI) [14].

The original HA-OSCAR implementation used a reliable shared storage solution to address the issue of storage access during a head node failure. Figure 1 illustrates that hardware architecture. Recent experiments with High Reliable Network File System (HR-NFS) have shown promise in eliminating the need for the special hardware. This test was with the HR-NSF kernel patch from Lifekeeper[10]. Our findings indicate that this approach provides better NFS handlings (e.g. re-attachment on clients) as the NFS server switches from one head node to the alternate head node. In doing this, the head node failover process is more transparent to the computing nodes that typically have home directories mounted on the NFS server. These are then simply remounted on the alternative standby head node following the failover process. In addition, we propose a checkpointing/restart and job queue migration framework to achieve application fault tolerance with minimum job data loss by triggering a checkpointing operation provided by PBS [11] appropriately and restoring the checkpoint file automatically after head node switchover.

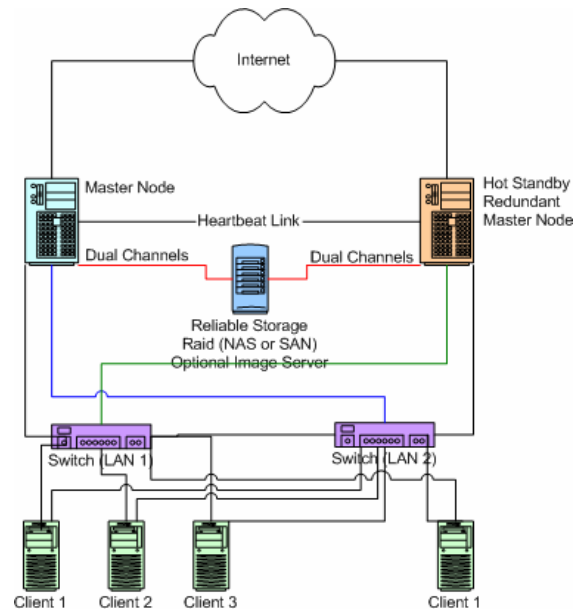


Figure 1: HA-OSCAR architecture.

#### 4 Box Leangsuksun, Tong Liu, Tirumal Roa, Stephen Scott, and Richard Libby

Lastly, HA-OSCAR supports a system resource and outage monitoring and recovery mechanism. It provides a Web-based service monitoring and configuration module incorporated with Webmin [15]. This web-based interface can be used to configure and monitor new services and resources on HA-OSCAR. It also provides a Web-based service monitoring and configuration program, which can be used to configure and manage services and resources on HA-OSCAR. Furthermore, this module will automatically generate all necessary alerts and monitoring files for HA-OSCAR MON-based self-healing mechanism.

### Related work

There had been certain related work in HA clustering: for example, Lifekeeper [10], Kimberlite [16], Linux Failsafe [17], Mission Critical Linux [18] and HP Serviceguard [19]. Most of these existing applications focused on developing sophisticated strategies to enhance system availability but did not consider high performance computing services. In addition, none of them supports policy-based failover management with hardware level failure prediction analysis. HA-OSCAR project recognizes the importance of combined high-availability and high-performance computing aspects. Our methodology and infrastructure is the first that we know of that provides a field-grade HA Beowulf cluster that provides high-availability and a critical failure prediction and analysis capability. Moreover, our approach provides a flexible and extensible interface for customizable fault management, policy-based recovery operation, and alert management. With a layering approach, HA-OSCAR encapsulates its self-healing framework into both vertical and horizontal abstractions that ensure extensive coverage and ease in development and upgrade. Figure 2 shows HA-OSCAR layering architecture.

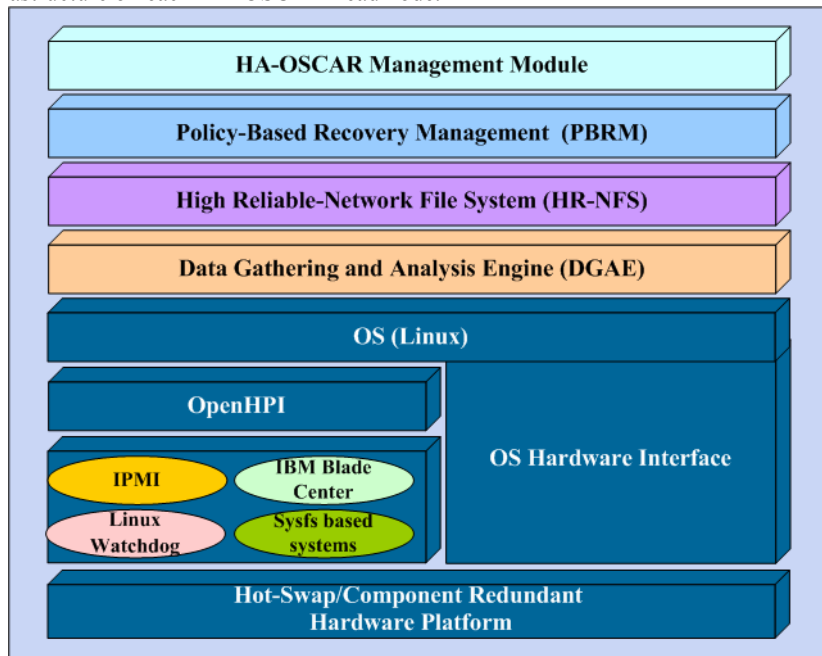
There are several packages in the open source community that we evaluated to help in our quest for high availability best practices to include in HA-OSCAR. Heartbeat is an open source package, often used on clusters to provide a software solution that monitors system/application “health” [9]. Although the heartbeat is a widely used HA technique, it does not meet our stringent high-availability requirements. Due to these stringent requirements, more powerful software packages were evaluated for our system health monitoring mechanism. The package selected, MON<sup>®</sup> is a general-purpose scheduler and alert management tool used for monitoring service availability and triggering alerts upon failure detection. It is designed to be open and extensible such that it supports arbitrary monitoring facilities and alerting methods via a common interface [8]. Our current HA-OSCAR monitoring and alert management framework is the MON<sup>®</sup> enhancement and provides distinguished user-configurable properties.

Supermon [20] is a tool used to monitor the status and behavior of cluster’s computing nodes. It is a high-speed cluster monitoring system with high data rate and low impact from a system resource perspective. But it does not provide ability and interface for analyzing monitoring data for failure prediction used in performing high

availability system administration. Details of cluster statuses and events are critical for HA-OSCAR failover policy administration and Management module.

## HA-OSCAR Head node Infrastructure

Our system architecture is based on the idea of interoperable building blocks that are necessary to build our HAPC system. Figure 2 illustrates an architectural infrastructure on each HA-OSCAR head node.



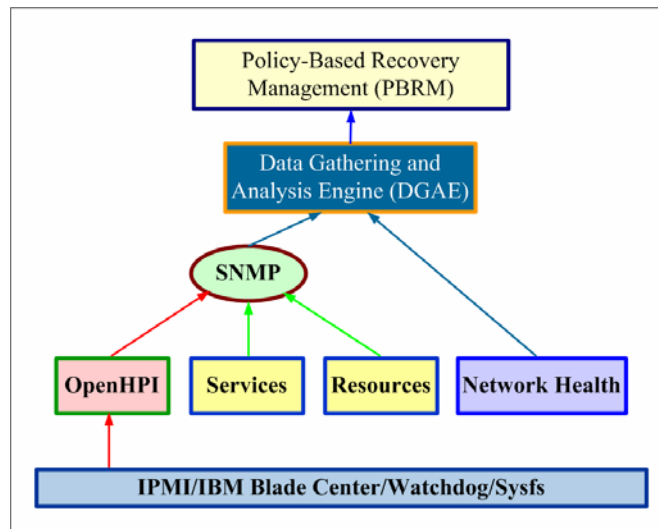
**Figure 2:** HA-OSCAR head node infrastructure.

HA-OSCAR head node infrastructure consists of multiple layers. Each of these layers includes a combined set of software and hardware solution stack that make up the High Availability infrastructure. The lower most layer is the hardware platform layer that provides the low-level hot-swap capability and component redundancy. In this environment, hardware components may be removed and replaced without affecting the operation of the remainder of the system. Such hardware platforms include those supporting the Intel IPMI based interface. HA-OSCAR takes advantage of the hardware platform interface and monitoring capabilities of OpenHPI, an encapsulation standard, as an open interface to access hardware information, status and management. OpenHPI is an open source project whose goal is to provide an implementation of the SA forum's Hardware Platform Interface (HPI). An application may embed OpenHPI calls to access sensor values, control system state, utilize watchdog and inventory data associated with resources, get the current status of hardware and receive alert when hardware failure occurs. The advantage of OpenHPI

over other solutions is that it is intended to be a universal and vendor-independent interface to various platforms, which allows the HA-OSCAR solution to support more hardware without vendor specific constraints. Experimentation with OpenHPI and the examination of real-time and historical data on the Intel server board hardware platform guided our effort to a design of our new failure predictive analysis module. As a result of this effort, we plan to build a production-grade system management middleware with capabilities to analyze the generic data and hardware events following user-defined policies and thresholds of cluster reliability to be used by the Policy-Based Recovery Management (PBRM) layer (see Figure 2). As an OpenHPI hardware-based management proof-of-concept, we constructed failover modules that can monitor hardware status via polling data from OpenHPI. The analysis module enables us to predict imminent failures, performs system failover with data backup in advance and achieves a faster failover than other detection methods that only depend on the network connection timeout or application status tests. In addition, we dealt with data loss issues by incorporating a High Availability-Network File System patch in our HAPC system. Our experiment suggests that client nodes could continue accessing data and running parallel programs with the very brief NFS outage after the failover recovery process.

### **HAPC Failure Detection, Predictive Analysis and Failover Strategy**

In this section, we will describe each critical component in our failure detection, predictive analysis and recovery mechanism. It entails layering methods including system event gathering (hardware, service, resources etc.), recovery policy and corresponding software packages as shown in Figure 3.



**Figure 3:** Failure Detection, Predictive Analysis and Recovery Module.

### Failure Detection and Predictive Analysis

Previously, HA-OSCAR failure detection was based on network and service availability. Failover or other appropriate recovery would be triggered when associated outages were detected. With our recent OpenHPI experiment, we recognized that additional hardware events and ability to manage such information will further improve cluster management and reliability. Moreover, predictive failure analysis will provide an opportunity to gracefully handle failures before potential outages occur. This prediction allows checkpointing and job migration before the failover occurs. Aggregation of these two main functions will deliver a critical precondition for the HA system.

### Hardware Platform Failure Detection and Analysis

To discover a fault or symptom, the detection module retrieves system information through Simple Network Management Protocol (SNMP). SNMP is one of the most widely used management frameworks used to monitor and control local and remote devices or services on a network. In HA-OSCAR, we first implemented various SNMP Management Information Base (MIB) for monitoring hardware, resource and services. The Data Gathering and Analysis Engine (DGAE) collects the SNMP-based status and events including hardware resources such as CPU usage, load average, disk space usage, memory capacity, and network errors via OpenHPI.

Our SNMP module can poll data from hardware (cards, fans, memory, power supplies, temperature sensors, alarm displays, storage devices) and software. It can also query various vendor specific hardware sensors providing they are OpenHPI-compliant. We gathered all event information and classified it into categories based on sensor type, threshold level, and severity. In DGAE, we define rules to aggregate various event categories. Once analyzed, DGAE filters important events and forwards them to PBRM module. In our current implementation, the DGAE forwards event info including resource name, event type, and event level (warning / critical) to PBRM. After parsing the event info, the PBRM module will generate an alarm and initiate the appropriate action with respect to the corresponding event level.

01/25/2004	00:31:19	Sys Fan 1	critical
01/25/2004	00:31:19	Sys Fan 3	critical
01/25/2004	00:31:19	Sys Fan 4	critical
01/25/2004	00:31:19	Processor 1 Fan	ok
01/25/2004	00:31:20	Processor 2 Fan	ok

**Table 1:** Hardware event.

Our prototype testbed was implemented using an Intel serverboard-based cluster, consisting of two head and four compute nodes. Each node was equipped with dual Xeon CPUs and IPMI-based sensors. DAGE captured sensor events such as system fan and processor fan. Table 1 shows samples of hardware events via OpenHPI. In these examples, they indicate that there were three events of temperature category exceeding a critical threshold at time 00:31:19. We established rules in PBRM such that if an event rate of the same kind is more than a defined threshold, say, 3 or more

of the same critical events within 5 minutes, it will consider that such event will lead to a system failure; otherwise we only classify it as a warning level event. This, of course could vary depending on the type of failure condition detected (for example, single-bit memory errors would need to have less time before action is taken).

In our PBRM module, we defined a few experimental policies to handle various event types. For example, if a system fan were to fail, the head node would failover and send out an alarm for maintenance. We intend to expand the failure predictive rules to cover events from CPU temperature, memory and other hardware resources that may also be monitored with a similar strategy. Our production-release solution will provide a set of default configuration parameters and rules for various hardware resources and their events. However, users will have the flexibility to define their own preferred set of rules. With appropriate predictive and recovery rules, our hardware failure prediction will allow dual head node servers enough time to checkpoint or synchronize data and application state prior to failover. Thus, it circumvents loss of valuable data while recovering a running cluster system with a very brief service interruption.

#### **Service Failure Detection and Analysis**

In addition to our hardware management, service fault management is certainly a critical feature necessary to ensure cluster high availability. To this extent, HA-OSCAR continuously monitors a set of predefined core services including: PBS, Maui, DHCP, NFS, and others. It also provides a web-based service monitoring and configuration module that enables users to add their own monitored services and set associated recovery actions. DAGE retrieves a service failure event whenever a specified service outage is detected. Employing the same principle with hardware failure detection and prediction, users can define predictive and recovery rules for critical services in the PBRM modules.

#### **Recovery Strategy**

As mentioned earlier, HA-OSCAR provides unique policy-based recovery management. We categorized hardware platform, service and resource high-watermark events and their severity used by the Failure Detection and Analysis Module. In an occurrence of a specific critical event, the PBRM module can trigger different recovery actions based on a user-configurable policy. For instance, the system recovery mechanism can choose to send out an email to a system administrator, restart a failed service, or failover. Figure 4 shows a system at a normal state, whereas figure 5 illustrates the failover.

### A Failure Predictive and Policy-Based High Availability Strategy 9

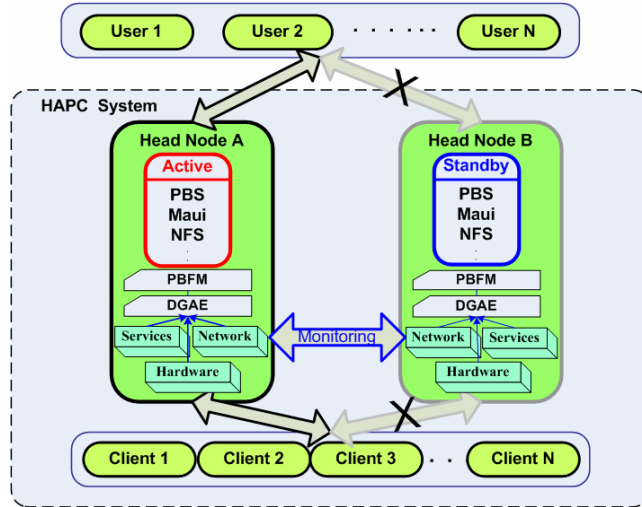


Figure 4: Initial Working State.

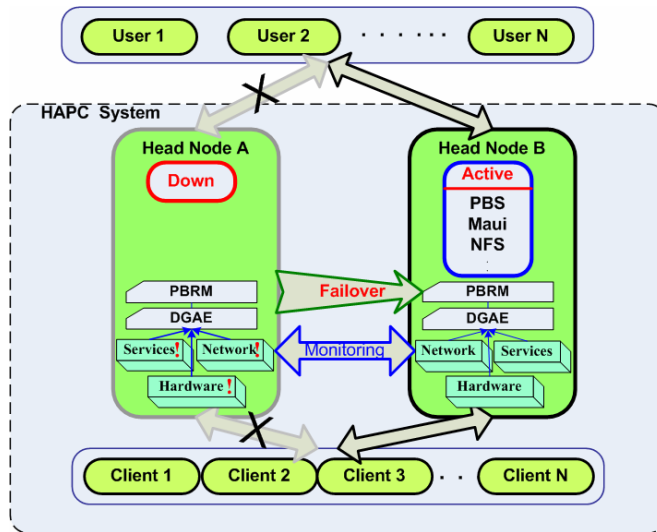


Figure 5: Head Node Takeover---Network Cloning.

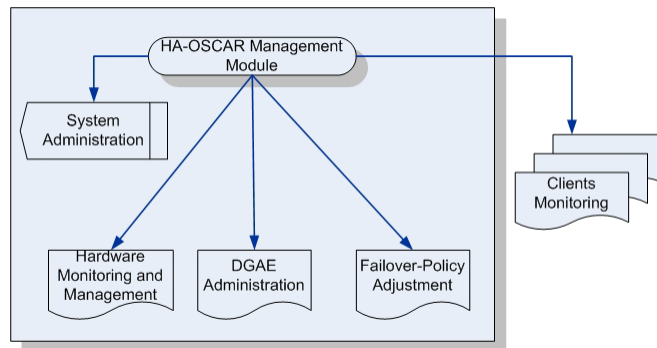
### High Reliable Network File System (HR-NFS)

In most cases, common critical data and system configurations should be stored in a reliable storage facility. HA-OSCAR recommends an optional shared reliable or networked storage as shown in figure 1. However, this hardware solution may be cost-prohibitive to some organizations. HR-NFS is an alternative solution. In a cost-

conscious configuration, client nodes normally access common file systems via a mounted point at the head node exported by the NFS server. Our experiments with NFS outages were with a current NFS release with the 2.6 Linux kernel. The HR-NFS kernel from Lifekeeper has solved the open file handle problem on the computing clients when restarting NFS service on the failover head node.

## HA-OSCAR Management

The HA-OSCAR Management module is responsible for system administration and cluster system status display including statistic and failure predictive information for selected hardware and core services. We incorporated a web-based management package based on Webmin[22], a very popular Linux administration tool. The HA-OSCAR Management tool provides users HA related functionality such as setting or modifying parameters for failover policy and remote management for the hardware platform with OpenHPI compliance. Figure 7 outlines HA-OSCAR management functionality.



**Figure 7:** HA-OSCAR Management Module.

These control methods for hardware platform include server power-off, power-on, reset, components such as unit identification on / off and obtaining sensor status. These features will provide a great benefit and enhance system manageability, especially for very large cluster deployments. For instance, when the system administrator receives an alarm email from our policy-based failover module, she can remotely force component reset by clicking the reset button on the web-based console that initiates the underlying IPMI reset function. In a case of multiple component failures, we can integrate a workflow management system with our HA-OSCAR Management module to create a work order based on event types and severity levels.

## Conclusion and Future Work

Our proof-of-concept implementation and experimental results show that HA-OSCAR provides a significant enhancement and promising solution to enable high-availability for the Beowulf cluster architecture. The introduction of the hot-standby server is clearly a cost-effective method when compared to the overall investment in typical large cluster. Therefore, HA-OSCAR clusters provide low-cost failure insurance to a cluster owner. Network cloning is used by HA-OSCAR to solve the important issue that Beowulf cluster operators face when trying to combine other simple IP-alias failover strategies. OpenHPI not only provides a universal interface to retrieve data and events from various vendor hardware resources, but also helps in managing those components with well-defined abstractions. Our DGAE and PFRM modules provide flexible self-healing benefits, based on the analysis of real-time data and events from hardware, resources, and services. Thus, it will increase system availability and fault tolerance. Moreover, HR-NFS will ensure NFS services both head nodes with a minimum of interruption and process overhead. All these features make HA-OSCAR an effective and easy method to implement an HA solution for high performance computing clusters.

The current implementation of HA-OSCAR supports an active-hot/standby dual-head node failover solution. This version, in beta at the writing of this paper, will be publicly released by the time of publication. Due to performance considerations that may impose by the single active head, the n+1 active-active architecture is currently under investigation. This version of HA-OSCAR has the inherited restriction that it operate on mostly homogeneous hardware due to the use of the node cloning software, System Installer Suite (SIS)[23]. We further plan to integrate with the various emerging technology checkpoint/restart MPI suites.

## 8 References

- [1] J Hsieh, T Leng, Y C Fang, *OSCAR: A Turnkey Solution for Cluster Computing*, Dell Power Solutions, Issue 1, pp. 138-140, 2001.
- [2] [oscar.sourceforge.net/](http://oscar.sourceforge.net/)
- [3] The Open Cluster Group, *OSCAR Cluster User's Guide*, Software Version 2.2, Documentation Version 2.2, February 27, 2003.
- [4] G. Stellner. CoCheck: Checkpointing and Process Migration for MPI. In *Proceedings of the 10th International Parallel Processing Symposium*, Honolulu, HI, 1996.
- [5] C. Leangsuksun, L. Shen, T. Liu, H. Song, S. Scott, *Availability Prediction and Modeling of High Availability OSCAR Cluster*, submitted to IEEE International Conference on Cluster Computing (Cluster 2003), Hong Kong, December 2-4, 2003.
- [6] [www.openclustergroup.org/HA-OSCAR/](http://www.openclustergroup.org/HA-OSCAR/)
- [7] C. Leangsuksun, L. Shen, T. Liu, H. Song, S. Scott, *Dependability Prediction of High Availability OSCAR Cluster Server*, The 2003 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'03), Las Vegas, Nevada, USA, June 23-26, 2003.
- [8] [www.kernel.org/software/mon/](http://www.kernel.org/software/mon/)

12 **Box Leangsuksun, Tong Liu, Tirumal Roa, Stephen Scott, and Richard Libby**

- [9] A Blackmon, J Nguyen, *High-Availability File Server with Heartbeat*, The Journal for UNIX and Linux Systems Administration, Vol. 10, No. 9, Sept., 2001.
- [10] [www.steeleye.com/](http://www.steeleye.com/)
- [11] [www.openpbs.org/](http://www.openpbs.org/)
- [12] Hewlett Packard\*: “*IPMI: Intelligent Platform Management Interface White Paper*”, February 1998.
- [13] The BlueGene/L Team. *An Overview of the BlueGene/L Supercomputer*, 2002.
- [14] [openhpi.sourceforge.net](http://openhpi.sourceforge.net)
- [15] [Jamie Cameron](#), *Managing Linux Systems with Webmin: System Administration and Module Development*, Prentice Hall PTR, August 14, 2003, pp. 30-40.
- [16] [oss.missioncriticallinux.com/projects/kimberlite/](http://oss.missioncriticallinux.com/projects/kimberlite/)
- [17] [oss.sgi.com/projects/failsafe/](http://oss.sgi.com/projects/failsafe/)
- [18] [www.missioncriticallinux.com/](http://www.missioncriticallinux.com/)
- [19] [www.hp.com/go/ha](http://www.hp.com/go/ha)
- [20] Ronald Minnich and Karen Reid. *Supermon: High performance monitoring for linux clusters*. In The Fifth Annual Linux Showcase and Conference, 2001.
- [21] [www.saforum.org/home](http://www.saforum.org/home)
- [22] [www.webmin.com](http://www.webmin.com)
- [23] [www.sisuite.org](http://www.sisuite.org)