

The OSCAR Toolkit:

Current and Future Developments

The Open Source Cluster Application Resources (OSCAR) software helps simplify cluster management by offering the best-known standards-based tools in one toolkit. This article discusses recent and proposed improvements to develop an OSCAR framework, such as increased modularity and support for diskless and high-availability clusters.

BY THOMAS NAUGHTON; STEPHEN L. SCOTT, PH.D.; YUNG-CHIN FANG; PHIL PFEIFFER, PH.D.; BENOÎT DES LIGNERIS, PH.D.; AND CHOKCHAI LEANGSUKSUN, PH.D.

Managing computing clusters is a challenging, often daunting task. A clustered system can take hours to install, configure, and test—even for experts. After installation, more hours of work follow: managing accounts, configuring applications, upgrading software, and solving or working around problems. The Open Source Cluster Application Resources (OSCAR) toolkit was developed to simplify the installation, configuration, and management of computing clusters.

The OSCAR toolkit, an open source software project, is the product of the OSCAR working group—industry, academic, and research organizations that contribute to OSCAR design and development. The group’s fundamental goal is to simplify cluster management by integrating the best-known standards-based tools.

Since the November 2002 release of OSCAR 2.0, the toolkit has evolved toward a more modular framework with enhancements to the build system and package facilities.¹ Two new projects build on this framework: Thin OSCAR (a diskless version) and High-Availability OSCAR.

Increasing modularity: The OSCAR framework

OSCAR has evolved from a simple bundle of standard clustering software into a framework for configuration management. This framework, like the original bundle, supports the fundamental goal of using the best-known standards-based tools to simplify cluster computing. The framework developed in response to the need for a more modular distribution, which became apparent as users requested new packages and OSCAR grew in popularity. Discrepancies among the various Linux® distributions also served to hasten the change.

Moreover, because of increasing demand, the number of popular high-performance computing (HPC) architectures has grown to include the following three classes: Intel® Pentium® processor-based architecture (IA-32); Intel Itanium® processor-based architecture (IA-64); and AMD® Opteron™ processor-based architecture (x86-64). Although OSCAR fully supports only IA-32 (and experimentally supports IA-64), the OSCAR working group plans to fully support all the successful HPC architectures.²

¹ For more information, see “Looking Inside the OSCAR Cluster Toolkit” by Thomas Naughton; Stephen L. Scott, Ph.D.; Brian Barrett; Jeff Squyres; Andrew Lumsdaine, Ph.D.; Yung-Chin Fang; and Victor Mashayekhi, Ph.D., in *Dell Power Solutions*, November 2002. The “References” section at the end of this article lists additional information resources.

² At the time of publication, the current release of OSCAR is version 2.3.

To increase modularity, developers began reworking OSCAR by decoupling the procedures for building and configuring packages from the software ultimately installed on clusters. This decoupling has allowed developers to reuse the framework for other purposes, including diskless and high-availability cluster system configurations. The framework has since evolved to include a build and configuration system, a database, and an environment management tool.

Build system facilitates cluster setup

A characteristic cluster management tool is a *build system*, a mechanism for configuring and installing cluster software. The OSCAR framework's build system, the System Installation Suite (SIS), uses the SystemImager[®] tool to support cluster node initialization. SystemImager generates a representative directory structure that can be traversed and manipulated just as a standard file system. This structure, or *image*, is downloaded to a cluster's nodes during system installation and then dynamically configured with host-specific values like IP and network interface card (NIC) addresses. Administrators can then change and redistribute the master image to a cluster's nodes as part of ongoing cluster maintenance. SystemImager also supports multicast-based installation, a new feature included in the OSCAR 2.3 release.

The OSCAR build system includes a graphical user interface (GUI-) based wizard that guides users through cluster setup and enables them to download and select packages for installation. Once a cluster's nodes have been initialized, OSCAR runs a series of simple tests to verify the installation. Administrators can later add or remove nodes by using this wizard.

All components (except the build system and database) of the OSCAR framework use the Oak Ridge National Laboratory (ORNL) Cluster Command and Control (C3) tools to perform actions across the entire cluster in parallel.³ The C3 tools facilitate parallel execution and scatter/gather operations that are essential for administrators and users alike, and the tools can operate on single or multiple clusters simultaneously.

OSCAR packages facilitate application installation

An OSCAR package provides a simple way to wrap an application for use in a target system. A package comprises an application, a meta file, and an optional set of scripts, tests, and documentation. Administrators configure packages by using the OSCAR package application programming interface (API). This API enables a package to query for data during cluster installation and execute configuration and setup routines specific to the application. The applications themselves are bundled using the standard RPM[™] (Red Hat[®] Package Manager) format.

The Open Source Cluster Application Resources (OSCAR) toolkit was developed to simplify the installation, configuration, and management of computing clusters.

OSCAR's modular packaging facility enables it to install software obtained from either the standard release or online package repositories. The OSCAR Package Downloader (OPD) aids in the acquisition of updated or third-party packages. OPD, a command-line tool, is similar in nature to tools that access the Comprehensive Perl Archive Network (CPAN). The installation wizard uses a GUI version of OPD called OPDer.

The package API supports package-specific interactive operations during installations. Administrators can, for example, configure packages to prompt for a default version if more than one instance of a package is selected. A package also can convey its platform dependencies to OSCAR by using the XML meta file. OSCAR uses these dependencies to generate the list of installable packages for the current platform and omits any packages whose dependencies are not met.

OSCAR database provides central repository

The OSCAR database (ODA) provides a central, package-updatable repository of information about a cluster's nodes and packages. Users access ODA through a command-line interface (CLI), which abstracts the underlying database engine (MySQL[®]) and provides a simpler interface for data access, including shortcuts for common package operations. The API scripts use these shortcuts to gather information such as available packages and number of nodes. In turn, any OSCAR package can use this information to initialize configuration files, such as the `/etc/c3.conf` configuration file in C3, and other package-specific settings.

OSCAR framework supports environment management

The task of managing user execution environments is a critical part of normal systems administration. Several common applications such as Parallel Virtual Machine (PVM), Message Passing Interface (MPI), and Portable Batch System (PBS) use environment variables to integrate their operations with their host platforms. More generally, easy command-line access to any application depends on the proper setting of a shell's command-path environment variable. If the relevant variables are missing or undefined, the system becomes extremely difficult to use.

³ Cluster Command and Control (C3) uses a configuration file to characterize the logical cluster structure. In OSCAR, a default configuration file is created for the user. See the C3 documentation cited in the "References" section for details on how to customize the configuration to achieve better scalability.

AD
pg. 31
Dell
(IBM LTO)

Users commonly employ two methods to maintain environment variables. The first method, which requires users to maintain their own environments, is cumbersome and error-prone. The second, which requires users to source a system-wide file at login, is superior to the first—but must be done well to help ensure proper system operation.

The OSCAR framework supports environment management with Env-Switcher, an extension of the Modules utility for cluster environments. Env-Switcher adds support for environment variable persistence over successive logins, and for cluster-wide environment consistency through an administrator-configurable script. Env-Switcher uses Modules to update system shells, propagating these modifications throughout a cluster. A package that must add items to the environment may do so by providing an Env-Switcher script that is loaded into the user's shell upon login.

Env-Switcher supports user-configurable defaults by providing a CLI for querying and modifying system-wide and user-level settings. Administrators could, for example, set a path for a system-wide default MPI implementation and let users inherit this default or override it with alternative user-level selections.

Developing OSCAR: New working groups

Since November 2002, the Open Cluster Group, OSCAR's parent group, has added two new working groups that are developing OSCAR-based configurations: Thin OSCAR for diskless environments and High-Availability OSCAR for high-availability system operation (see Figure 1).

Thin OSCAR supports diskless clustering technique

The three classic rationales for diskless operation are decreased cost from eliminating the local hard drive, which is no longer needed to host the system's initial image; increased reliability from eliminating the local hard drive; and speed, if the network connection can be accessed more quickly than the local disk. Thin OSCAR supports diskless operation by extending standard OSCAR with three new classes of nodes:

- **Diskless:** Has no local disk; uses the master node for long-term storage over the network
- **Diskful:** Supports local disks
- **Systemless:** Uses local disks only for temporary storage or swap space, not to store resident operating systems

Currently, Thin OSCAR directly supports only diskless and diskful nodes; systemless nodes require manual configuration.

Diskless clusters use the Dynamic Host Configuration Protocol (DHCP) for image distribution to manage system initialization. At startup, a diskless node uses its network interface to download a

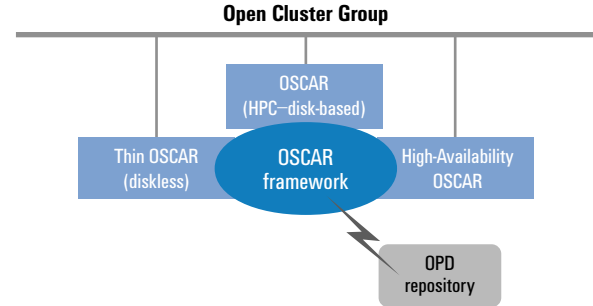


Figure 1. The OCG working groups share the OSCAR framework for cluster installation and management

small boot image. The diskless bootstrap procedure then uses this image to load a complete run image on the target node.

The current Thin OSCAR implementation uses a collection of Perl scripts and supporting libraries to transform a regular SIS image into the two RAM disks necessary for diskless and systemless operation. The first RAM disk—the boot image—ensures that a node has network connectivity. The second RAM disk—the run image—is built directly from the SIS image and contains the complete system that will run on the node. Some material is directly copied from the SIS image, while other parts are exported to the nodes through the Network File System (NFS) in a read-only mode, directly from the SIS image.

Currently, Thin OSCAR is provided through an OSCAR package. The standard OSCAR installation wizard handles most, but not all, of the steps needed for Thin OSCAR installation. Administrators can use the CLI to build the two RAM disks necessary for diskless and systemless operation; this step occurs outside the actual installation wizard. The OSCAR installer (currently a simple wizard approach) is being redesigned to be more modular, as was the case with the package API. Once the installer development is complete, the external steps will be integrated into the installation process.

Most of the Thin OSCAR concepts have been validated; they are now being used in a production environment—a 180-node diskless cluster at the Université de Sherbrooke in Québec, Canada. This cluster is fully functional and will grow to approximately 300 nodes by the end of 2003.

High-Availability OSCAR helps eliminate single points of failure

High-availability computing has become critical to the fundamental mission of high-performance computing. HPC systems are being tasked to run very large and complex applications, whose run-times exceed their host systems' aggregated mean time between failures (MTBF) rate. To effectively run code on HPC machines, high-availability computing techniques are necessary to prevent *code thrashing*—the rerunning of many code segments when attempting to recover from a failure. Instead, systems must maximize the underlying HPC environment.

High-availability computing differs somewhat from fault-tolerant computing: the first is proactive by sensing and preventing potential failures, whereas the second is usually costly and reactive. In fault-tolerant computing, replicated components execute the same instructions at the same time, so even if one fails, the application keeps running, with no difference other than perhaps a reduced performance level based on the percent of resources lost. A significant cost of fault tolerance is the lockstep redundant execution and frequent checkpointing that must occur regardless of failure state.

To support high-availability requirements, clustered systems must eliminate single points of failure. During its first phase of operation, the new HA-OSCAR working group will seek to eliminate single points of failure in the current HPC release of OSCAR through active/hot-standby configurations and eventually through implementing $n + 1$ active/active distributions. These strategies involve hardware duplication and network redundancy, common techniques for improving the reliability and availability of computer systems.

Initial efforts in this area will focus on supporting a duplicate cluster master node. Various techniques currently exist for implementing such an architecture, which includes active/active, active/hot standby, and active/cold standby. Figure 2 shows the High-Availability OSCAR cluster system architecture.

Members of the HA-OSCAR group who have experimented with these techniques plan to incorporate Linux Virtual Server and heartbeat mechanisms into an initial active/hot-standby High-Availability OSCAR distribution. This architecture will be extended to support active/active high availability after release of the hot-standby distribution. The active/active architecture will provide for better resource utilization, because both master nodes will be simultaneously active and providing services.

The dual master nodes will run redundant OpenPBS, Maui, DHCP, Network Time Protocol (NTP), Trivial FTP (TFTP), NFS, rsync, and Simple Network Management Protocol (SNMP) servers. If a master node fails, all functions provided by that node will fail over to the second, redundant master node. All service requests will continue to be met, although at a reduced performance rate (in theory, at 50 percent of the master node's peak or busy hours).

HA-OSCAR also will support a high-availability network through redundant Ethernet ports on every machine, and duplicate switching fabrics (such as network switches and cables) for the entire network configuration. This functionality is designed to enable every node in the cluster to be present on two or more data paths within the cluster networks. Backed with this Ethernet redundancy, the cluster will achieve higher network availability. Furthermore, when the entire network is up, techniques such as channel bonding of messages across the redundant communication paths may improve communication performance.

Enhancing OSCAR: Future developments

The current OSCAR development path includes enhancements to the package API and a simplified procedure for package addition and removal. These enhancements will support more precise characterizations of package and target information (such as supported distribution and architecture), which in turn will support package updates between full OSCAR releases. The developers also are reworking the toolkit to simplify upgrading packages between releases. These enhancements will eliminate dependencies between packages and full OSCAR releases, streamlining package management.

Developers are making the OSCAR Installer more flexible by adding support for a user-configurable front end—for example, a CLI interface—along with a user-configurable build engine specific to Thin or High-Availability OSCAR. The Scientific Discovery Through Advanced Computing (SciDAC) Scalable System Software (SSS) project is using OSCAR as a deployment vehicle to simplify the evaluation and eventual adoption of the SSS suite. The OSCAR framework itself also is being fitted with an SSS interface for use as a build and configuration system.

The need for better cluster management software has spurred the development of OSCAR over the past three years and has led to the Thin OSCAR and HA-OSCAR working groups. The efforts to modularize the toolkit for better reusability have enabled these new groups to reuse the existing framework for cluster installation and management. In addition to the modularized package system, the build system has added multicast (SIS) support and improved scalability for the parallel toolset (C3).

Managing clusters is a time-consuming task. OSCAR helps to reduce the installation, configuration, and management costs of a cluster. The OSCAR framework assists with integrating reusable

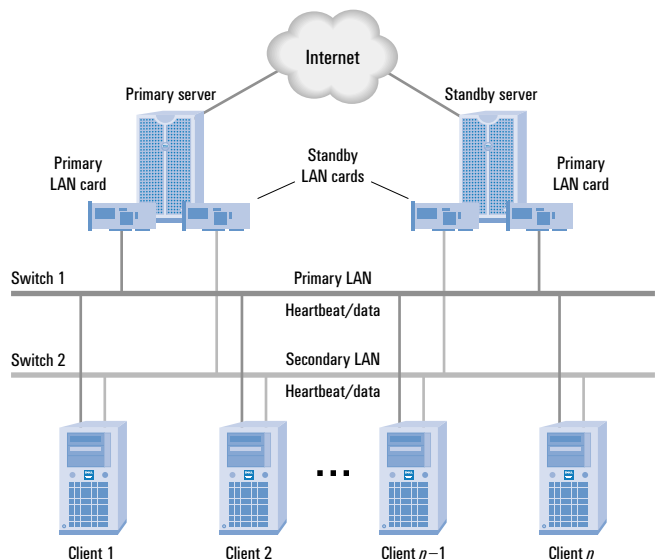



Figure 2. The High-Availability OSCAR cluster architecture facilitates failover

software on a cluster. This reusability and extensibility is a primary goal for the future developments of OSCAR. 

Acknowledgments

Work by ORNL was supported by the U.S. Department of Energy under Contract DE-AC05-00OR22725.

The authors would like to thank the OSCAR community, fellow developers, and users, as well as the organizations that support member contributions to the project.

References

“Abstract Yourself with Modules.” <http://www.usenix.org/publications/library/proceedings/lisa96/pwo.html>.

Clusters for High Availability: A Primer of HP Solutions. 2nd ed. Englewood Cliffs, N.J.: Prentice-Hall, 2001.

“C3 Power Tools.” *Distributed and Parallel Systems: Cluster and Grid Computing*. Boston: Kluwer Academic Publishers, 2002.

“Development, installation and maintenance of Elix-II, a 180-node diskless cluster running thin-OSCAR.” http://hpcs2003.ccs.usherbrooke.ca/papers/desLigneris_03.pdf.

“Open Source Cluster Application Resources (OSCAR): design, implementation and interest for the [computer] scientific community.” http://hpcs2003.ccs.usherbrooke.ca/papers/desLigneris_01.pdf.

“OSCAR Clusters.” *Proceedings of the Ottawa Linux Symposium*. Ottawa, Canada: Linux Symposium, 2003. See also: <http://archive.linuxsymposium.org/ols2003/Proceedings/All-Reprints/Reprint-Scott-OLS2003.pdf>.

“The OSCAR Revolution.” *Linux Journal*, June 2002. See also: <http://www.linuxjournal.com/article.php?sid=5559>.

“The Penguin in the Pail—OSCAR Cluster Installation Tool.” *The 6th World MultiConference on Systemics, Cybernetics and Informatics (SCI 2002)*. Orlando, Fla.: International Institute of Informatics and Systemics, 2002.

“System Installation Suite: Massive Installation for Linux.” *Proceedings of the Ottawa Linux Symposium*. Ottawa, Canada: Linux Symposium, 2002. See also: http://www.linux.org.uk/~ajh/ols2002_proceedings.pdf.gz.

“Thin-OSCAR: Design and future implementation.” http://hpcs2003.ccs.usherbrooke.ca/papers/desLigneris_02.pdf.

Thomas Naughton (naughtont@ornl.gov) is a research associate in the Computer Science and Mathematics Division, ORNL. Thomas has a B.S. in Computer Science and a B.A. in Philosophy from the University of Tennessee—Martin, and an M.S. in Computer Science from Middle Tennessee State University.

Stephen L. Scott, Ph.D. (scottsl@ornl.gov) is a senior research scientist in the Computer Science and Mathematics Division, ORNL. Stephen leads the cluster computing effort at ORNL. He has a B.A. from Thiel College in Greenville, Pennsylvania, and an M.S. and Ph.D. from Kent State University in Kent, Ohio.

Yung-Chin Fang (yung-chin_fang@dell.com) is a member of the Scalable Systems Group at Dell. Yung-Chin has a bachelor’s degree in Computer Science from Tamkang University in Taiwan and a master’s degree in Computer Science from Utah State University. He is currently working on his doctorate degree.

Phil Pfeiffer, Ph.D. (phil@etsu.edu) is a professor of computer science at East Tennessee State University (ETSU) and is currently working with Dr. Scott on an ORNL-sponsored non-instructional leave. Phil has a B.S. in Computer Science from Yale University, and an M.S. and Ph.D. from the University of Wisconsin-Madison.

Benoît des Ligneris, Ph.D. (benoit.des.ligneris@ccs.usherbrooke.ca) is a postdoctoral fellow in the Scientific Computing Center of the Université de Sherbrooke. He has a B.Sc. and M.Sc. from Pierre & Marie Curie University in Paris, and an M.Sc. and Ph.D. from the Université de Sherbrooke.

Chokchai Leangsuksun, Ph.D. (box@latech.edu) is an associate professor of computer science and an affiliate of the Center for Entrepreneurship and Information Technology (CENIT) at Louisiana Tech University. He also is a director of the eXtreme Computing Research (XCR) Group. He has a B.Eng. from Khon Kean University, Thailand, and an M.S. and Ph.D. in Computer Science from Kent State University in Kent, Ohio.

FOR MORE INFORMATION

C3 power tools:

<http://www.csm.ornl.gov/torc/C3>

Open Cluster Group:

<http://www.openclustergroup.org>

OSCAR Cluster User’s Guide:

http://oscar.sourceforge.net/docs/oscar_user_2.3.pdf

OSCAR project:

<http://oscar.sourceforge.net>

System Installation Suite (SIS):

<http://www.sisuite.org>

Thin OSCAR working group:

<http://thin-oscar.ccs.usherbrooke.ca>

AD

pg. 35

DLTtape