Scalable Platform Services on the Intel TFLOPS Supercomputer

Bradley Mitchell, Server Software Technology, Beaverton, OR, Intel Corporation

Index words: DMI, management, scalability

Abstract

This paper describes Scalable Platform Services (SPS)—a collection of software providing the manageability solution for Intel's latest parallel processing supercomputer.

Compared to previous generations of supercomputer management environments, such as that of the Intel Paragon™ Supercomputer, the SPS makes significant strides in feature offerings and overall usability. The SPS consists of distributed, low-level hardware monitoring, and control functions networked to a centralized management station which are in turn exported to administrators through command-line and graphic user interfaces. This software system demonstrates successful application of off-the-shelf standard components, chiefly the Desktop Management Interface (DMI) supported by Intel and the Desktop Management Task Force.

Together with specialized management hardware, the SPS offers a platform management architecture designed for scalability, availability, usability, and high performance.

Introduction

Supercomputers installed at customer sites require good manageability characteristics. In general, high demand exists for machine cycles, and frequent system downtime proves very costly. For the Intel TFLOPS supercomputer in particular, the sheer quantity of densely packaged hardware makes the task of identifying and repairing failed components difficult. (For an overview of the Intel TFLOPS supercomputer, please refer to the paper entitled *An Overview of the Intel TFLOPS Supercomputer* also in this Q1'98 issue of the Intel Technology Journal.)

The manufacturing and system integration of supercomputers also demands fairly comprehensive management support. Assembling the Intel TFLOPS supercomputer involved rigorous hardware configuration and testing activities that could not be completed on schedule without sufficient automation. Facilities for hardware resource sharing became crucial. Even boot and shutdown procedures for the TFLOPS operating systems needed abstraction.

SPS, a distributed software system, addresses both manufacturing and field requirements as outlined above. The SPS architecture focuses on the unique characteristics of the TFLOPS platform yet succeeds in leveraging off-the-shelf software products to meet its delivery time constraints and quality objectives.

One can compare managing the Intel TFLOPS supercomputer with the task of managing a collection of several thousand distributed PC desktops. (This analogy will be revisited later in the paper.) Although both environments contain roughly the same number of "nodes," the Intel TFLOPS supercomputer poses a number of additional challenges from a manageability perspective. These arise from the machine's scale, use of custom hardware components, usage model, and so on.

A significant goal for management environments according to Intel's Wired for Management (WfM) initiative is to reduce the administration cost associated with an installed hardware base. In the PC desktop realm, reducing this cost involves a combination of hardware support, software support, and standards efforts. Although WfM does not address the supercomputer realm explicitly, the principles of WfM remain applicable. By utilizing built-in hardware support, leveraging industry standards such as the Desktop Management Interface, and by providing new examples of management software components, the SPS for TFLOPS serves as a fine example of WfM principles in action.

TFLOPS Platform Management

The SPS focuses on aspects of system management that relate closely to the unique hardware architecture of the

Intel TFLOPS supercomputer. In particular, the TFLOPS machine includes a Monitoring and Recovery Subsystem consisting of hundreds of boards dedicated to hardware instrumentation and low-level control functions. Private Ethernet, as well as a mesh of serial line connections, network these boards together. A considerable fraction of the SPS software exists to provide communication and co-ordination services for utilizing this "platform within the platform."

Scalable Platform Services

The SPS provides the following features to a TFLOPS supercomputer system administrator:

Scripted booting/shutdown. Individually-executing scripts co-ordinate the booting or shutdown of the two distinct operating systems available on the machine.

Fault management. A software agent detects hardware faults as they occur, isolating affected components from the running system when possible, reporting fault information, and initiating automatic recovery operations in some scenarios. This same agent receives notifications of software faults from the running operating systems and takes appropriate corrective actions.

Configuration management. A software agent maintains up-to-date hardware inventory data, supplying this data to clients on demand. This same agent accepts client requests to partition the available hardware resources into independent sub-machines.

Repair services. Individually-executing scripts support board repair, power control, firmware upgrade, and hardware reset operations.

Field diagnostics. Scripts encapsulate diagnostic test scenarios covering many platform hardware components.

Operating system console access. A gateway service provides access to operating system node consoles from remote clients.

Architecture

Based on a centralized Desktop Management Interface

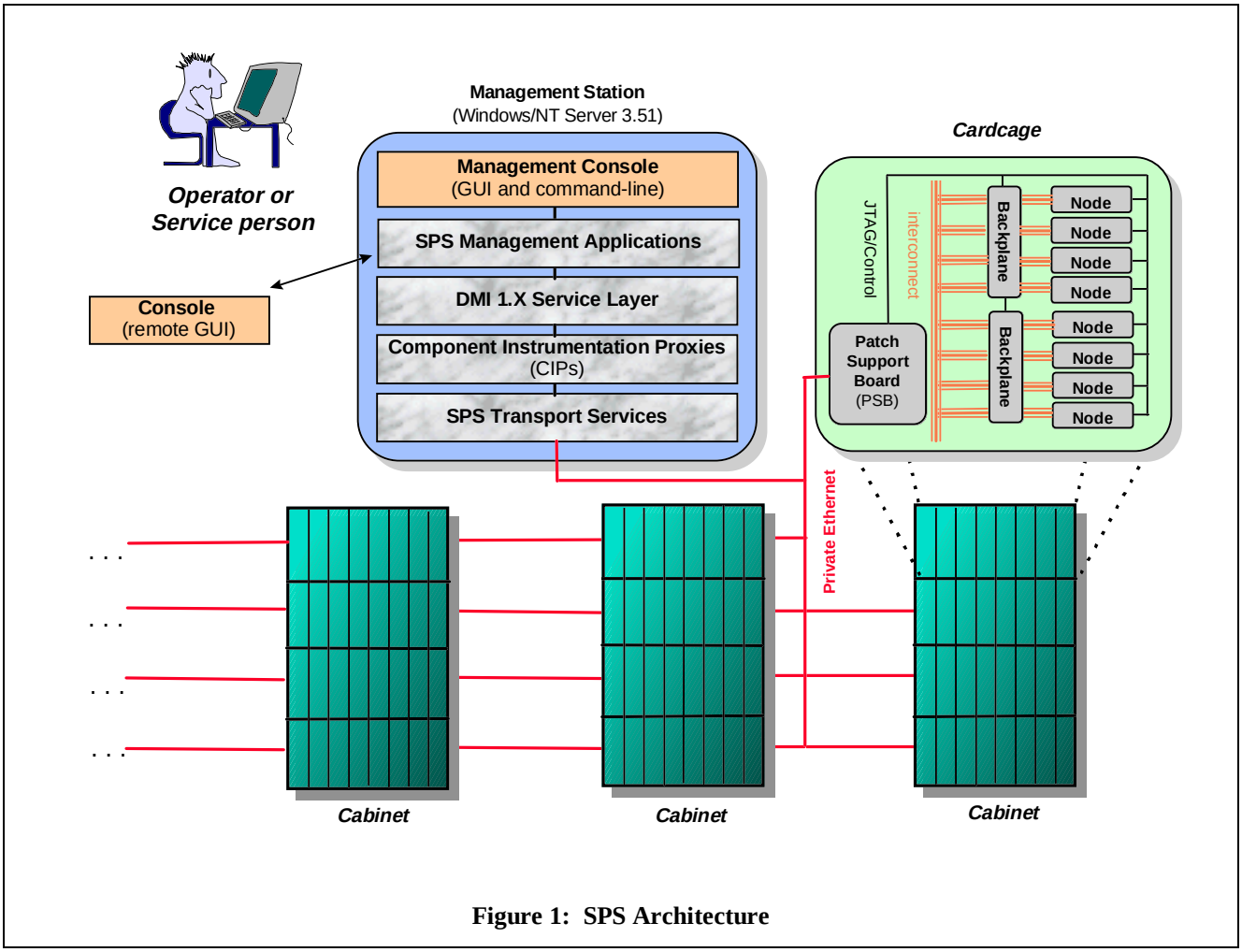

Figure 1: SPS Architecture

(DMI) database, the SPS includes a co-operative set of local management applications (MAs) as well as remote component instrumentation. Five MAs exist corresponding at a high level to the management features supported: *Booting*, *Configuration*, *Diagnostics*, *Fault*, and *Repair*. Unlike many DMI MAs that have a dedicated human interface, the SPS GUI integrates all five MAs into a single coherent presentation. The DMI database and MAs install onto the dedicated "management station" that serves as a central point of administration for the platform

The platform hardware includes Patch Support Boards (PSBs) that oversee each group of eight computational nodes in much the same way the Intel Server Monitor Module oversees a single server system. A fully-populated TFLOPS machine includes roughly 300 PSBs. Each PSB includes an i386™ processor and runs the Wind River Systems VxWorks* real-time operating system. Software agents implemented above the OS, coupled with hardware instrumentation capability built into each PSB, provide a full range of monitoring and control functions for the nodes, backplanes, and other platform components.

Via a transport layer on each end, DMI component instrumentation proxies on the management station communicate with PSB agents to collect instrumentation data for and deliver management requests from a central location. On the station, five such proxies exist broadly corresponding to the types of hardware present in the machine: *Node*, *Backplane*, *PSB*, *Clock*, and *Cabinet*.

An administrator interacts with the SPS through a graphic user interface (GUI) and related command-line interfaces. The GUI implements direct manipulation interfaces for specifying hardware components, individually or in groups, as the target of management operations. It launches management scripts that encapsulate specific booting, diagnostic, or repair operations. Its network map includes an event-driven interface that color-codes individual hardware components based on their real-time state. Finally, it displays in real-time the OS event log entries made by the management applications.

Figure 1 illustrates the SPS software architecture in relation to the hardware platform. Subsequent sections describe the software layers in more detail.

Patch Support Board Software

The PSB software consists of the VxWorks kernel and a set of runtime-loaded SPS modules.

The base VxWorks distribution from Wind River required several source extensions and modifications for operation on a PSB. For example, the SPS team re-wrote its network initialization module—delaying the network initialization process within the OS until platform configuration discovery within SPS modules completes. Additionally, the team added support for reading the on-board EEPROM needed to access, among other things, the PSBs MAC address.

Each SPS object module implements an agent or, alternatively, a library of services shared among agents. Each agent provides management services either for one type of Field Replaceable Unit (FRU)[1] or for one type of communication interface (serial or packet). Libraries implemented include a JTAG[2] scan library providing a standards-based interface to instrumented hardware devices.

In general, the PSB agents support both monitoring and control functions. They receive hardware interrupts and collect instrumentation data on behalf of the management station. As needed, the agents transmit information to the station over the private Ethernet network. The agents also accept requests such as those for power or reset control from the management station and execute the required processing and notification procedures.

Finally, this software implements a communication and keep-alive scheme among the PSBs, taking advantage of the serial line network fabric connecting all PSBs. The management hardware for the Intel TFLOPS supercomputer thus includes two management networks: a secondary network assists in managing the hardware critical to the machine's computational mission, and a tertiary network assists in managing the management hardware itself. Both of these networks exist "out-of-band" from the machine's primary node interconnect.

Node Maintenance Port

One of the two standard serial ports (COM2) on the node board serves a special function as the Node Maintenance Port (NMP). This port connects to the PSB and is used in two ways: the message mode supports reliable datagrams, and the raw mode supports an unstructured, not necessarily reliable, character stream.

The PSB supports multiplexing of data in the two modes and also works as a gateway for the NMP communication

[1] SPS supports the following FRU types: node boards, backplanes, clock boards, power supplies, blower units, and the PSB itself.

[2] JTAG refers to the Joint Test Action Group and IEEE Standard 1149.1 for implementing boundary-scan functionality in hardware devices.

between its eight nodes and the management station. The SPS management applications use the NMP message mode to communicate reliably with the extended node BIOS and with downloaded, off-line node diagnostics. The NMP raw mode supports an administration and debug console to the operating system on the nodes. On the management station, NMP libraries communicate in message or raw mode transparently to any node in the system by hiding the communication to the gateway service on the appropriate PSB.

Management Station Software

All software components resident on the management station operate in a 32-bit Windows NT* environment. Figure 2 illustrates these components, and they are discussed in the following sections.

PSB Transport Service

The PSB Transport runs as a Windows NT* service on the management station. It provides transport services for sending and receiving reliable datagrams between the station and one or more PSBs, multicast group definition, PSB enumeration, and notification when PSBs enter or drop off the network. The inter-process communications mechanism between the transport and its users incorporates local RPC. PSB Transport interface libraries hide the RPC initialization and tear-down details.

Component Instrumentation Proxies

The SPS utilizes DMI as the primary management interface. This choice allows the TFLOPS system to use standards-based management and helps facilitate a single operational view through a single management interface.

The DMI standard defines a *Component Instrumentation* interface that performs the required low-level management operations. However, the actual low-level operation takes place on the PSB. To bridge this gap Component Instrumentation Proxies or CIPs were implemented.

As mentioned earlier, the five proxies provided are *Node*, *Backplane*, *PSB*, *Clock*, and *Cabinet*. Each CIP runs as a Windows NT* service that communicates with a corresponding agent on the PSB to collect instrumentation data and to deliver management

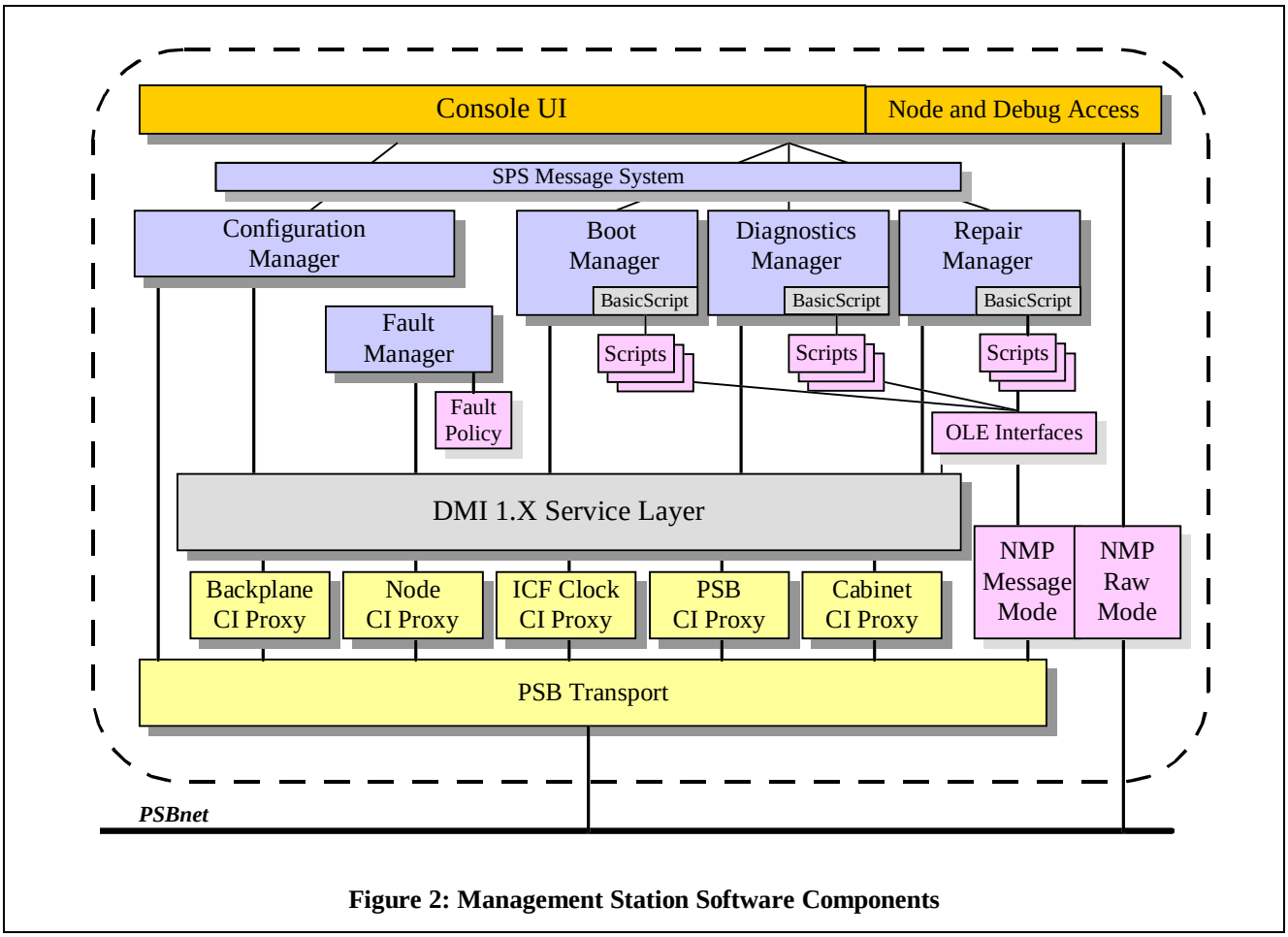

Figure 2: Management Station Software Components

requests. The two primary interfaces for a CIP are with the PSB Transport and the DMI Service Layer.

DMI Service Layer

The SPS utilizes version 1.X of the Desktop Management Interface (DMI). DMI includes a specification written and maintained by the Desktop Management Task Force (DMTF), an industry consortium chartered with the development, support, and maintenance of management standards for PC systems and products. As part of this specification, the DMI service layer provides a standards-based interface between the SPS management agents and component instrumentation responsible for low-level management tasks. It is this interface boundary that SPS takes advantage of to provide the illusion of a single unified system through a single DMI database.

A DMI database generated for the Intel TFLOPS supercomputer contains more than 3,600 component entries. This results in more than 21,500 individual DMI groups and more than 99,000 DMI attributes. Prior to the SPS project, Intel's existing DMI service layer implementation could scale to only 254 component entries. However, Intel made the necessary enhancements to accommodate this SPS scaling requirement.

In addition to scaling the number of components supported, Intel's DMI implementation required an enormous reduction in the time required to generate a large component database. The Intel team reduced this time from two calendar days initially to less than fifteen minutes.

Management Applications

The five SPS management applications are *Boot*, *Configuration*, *Diagnostic*, *Fault* and *Repair*. Each MA runs as a Windows NT* service on the management station.

The Boot, Diagnostic, and Repair Managers advertise a list of available control operations. Each entry in this list corresponds to an executable script. The team implemented these scripts with the BasicScript* environment from Summit* Software, so chosen because the LANDesk* Server Manager product also uses it.

When invoked, each script executes in a separate thread on the management station under control of the BasicScript run-time environment. Multiple scripts may execute concurrently. To allow scripts to access the DMI service layer, the SPS team developed a simple OLE[3] interface library to DMI. (The team prepared a similar OLE library for interface to the NMP.) Most scripts interact with an operator as described in the next section.

The Configuration Manager exists primarily as a server-side agent to the client user interfaces. It uses the SPS message system to converse with instances of the SPS GUI and command-line utilities. It initializes the PSB private network.

The Fault Manager monitors events reported by the PSBs and processes all fault events. It includes a custom inference engine that synthesizes and correlates fault indications from across the platform and initiates automatic corrective action where possible. The Fault Manager includes a `lex`/`yacc` fault grammar.

SPS Message System

The SPS team implemented a simple network transport supporting reliable datagram communications between SPS Managers and (potentially) remote clients. Its main purpose is to support the SPS GUI running on a machine other than the management station.

The Message System uses secure RPC interfaces to support authentication. Additionally, a built-in authorization scheme restricts operator access to management functions according to their membership in Windows NT* security groups.

The Message System provides an API matching that of the Intel LANDesk* Server Manager Message System (server-side) and the Network Transport Server (client-side). Because SPS originally used the LDSM transport, matching the LDSM API allowed the SPS team to replace those components in isolation.

Graphic User Interface and Scripting Environment

Figure 3 illustrates the SPS graphic user interface. This interface runs either on the management station or on a remote Windows NT* console. The GUI utilizes the SPS message system to interface to the SPS managers. The GUI implementation relies heavily on the Microsoft Foundation Class (MFC) library for Win32.

The SPS GUI employs a network map metaphor. The topmost portion of the main window contains a thumbnail sketch representing all cabinets in the machine, rendered in rows and columns just as the actual hardware is installed on the floor. Beneath this sketch, the middle portion of the main window displays a zoom-able view of one or more cabinets within the machine.

As shown in Figure 3, objects on the screen include color codes to match the current state of the corresponding

[3] OLE refers to Microsoft's Object Linking and Embedding technology.

piece of hardware. The color red corresponds to a faulted FRU, green to a healthy FRU, and so on.

An operator launches SPS scripts from the tree included in a dock-able window at the bottom of the GUI main window. The GUI also presents a list of existing partitions in this window. To create partitions, the operator rubber-bands selected cabinets. Likewise, the rubber-banding mechanism allows an operator to zoom the main view in or out.

Figure 3 illustrates a typical configuration for the TFLOPS machine. While the full system includes four rows of 19 cabinets each, operators typically de-couple the rightmost 4x4 cabinets for customer site security purposes. When de-coupled, the SPS GUI will not render these cabinets, although it will account for them in its numbering scheme. When the system transitions to a non-secure operating mode, operators typically re-cable the rightmost 4x4 block and correspondingly de-couple the leftmost 4x4. SPS allows the operators to construct hardware/software "partitions" of the hardware for this and other reasons such as unobtrusive diagnostic testing or hardware resource sharing.

Challenges

The most pervasive challenge faced by the SPS team lies in integrating the diverse collection of off-the-shelf software components.

Another significant challenge, one faced in many high-end platform software development projects, is a lack of available target hardware. The partitioning feature in the SPS mitigated this problem to some degree in the development labs. Yet, the limitation applies particularly now in the sustaining phase of the project when nearly all existing hardware has been shipped to the customer.

A variety of scaling challenges was also encountered. Utilization of the PSB network required careful planning to avoid performance bottlenecks. The DMI implementations required modification to support the large number of installed components. And the design of the SPS GUI required particular emphasis on ease of navigation and clear display of fault notifications.

Another serious development challenge grew out of the need for SPS to support manufacturing test needs. Lacking any other tool support, the manufacturing team required access to incomplete versions of SPS for booting, diagnostic, power control, and other base features. Combined with each of the other challenges mentioned above, this forced the development team to regularly make trade-offs between feature availability, stability, and progress toward the final product. Nor did the feature set in the field always match exactly the functionality needed internally. In this environment of rapid change, it proved very difficult to re-train users of the SPS as features were enhanced over time and the need for special "work-around" procedures was removed.

In a system as large as the Intel TFLOPS supercomputer, the secondary management problem–managing the support hardware and software–can become as difficult as some people's primary management problem. SPS attempted to deal with both problems. However, details

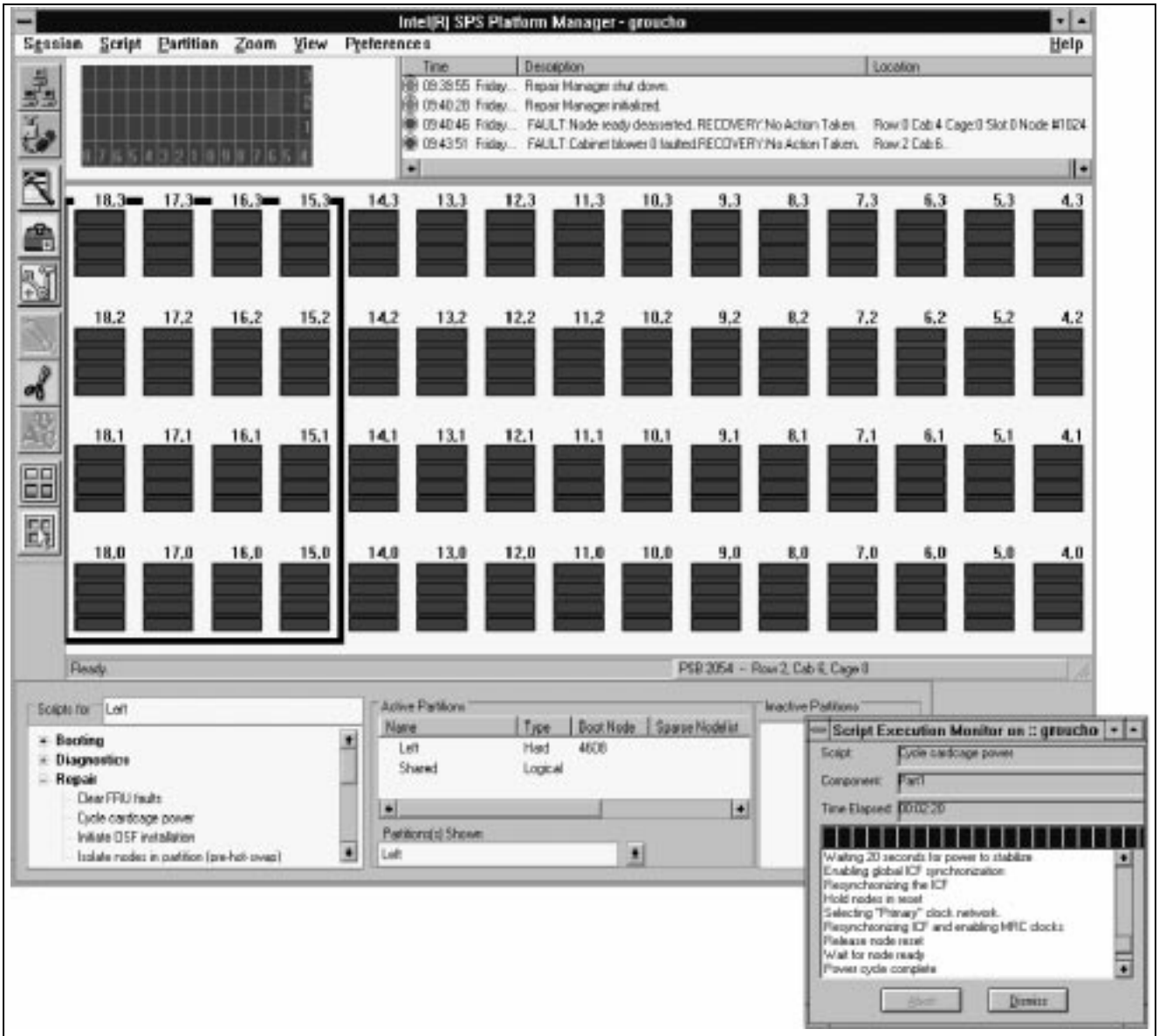

Figure 3: SPS Graphic User Interface

such as software/firmware installation and upgrade of PSBs, or static IP address assignment for the PSB network, proved non-trivial to debug and overcome in practice.

Finally, the SPS overcame a few cultural barriers to acceptance. Traditional supercomputer management environments consist of UNIX* workstations exclusively, and for SPS to be appealing to certain classes of users, UNIX-like command interfaces to particular SPS features were required. By using the TelnetD `telnet(1)` server for Windows NT* from Pragma Systems, Inc., and by tasking Pragma to tailor the product for better interoperability with the TFLOPS OS debugger, these concerns were largely ameliorated.

Results

The SPS played an important role in enabling Intel to win the one teraflop performance race, to meet its contractual commitments, and to raise the bar for supercomputer management software.

In general, the team's experiences in integrating off-the-shelf software components were positive and met or exceeded expectations. In particular, DMI performed nicely and proved to be the single biggest "win" for the project. In most cases, including the DMI implementation, the VxWorks kernel, BasicScript, and the RATP specification, enhancements were required or software defects were encountered that required source modifications. Yet, in every case, these alterations did not derail the project, in part because the changes were small and in part because they were anticipated as part of the development effort.

In only one case, that of the LANDesk Server Manager, did the team replace components of the product in mid-stream. This was due not to any particular technical shortcoming of LDSM, but rather to the fact that the feature offset between it and SPS was too great. Specifically, where LDSM targets heterogeneous workgroups that include servers, SPS targets one instance of one unique type of Intel-architecture based server.

As in many software development projects, the team found room for improvement. In general, more tool support could have been provided for managing SPS itself, including installation, upgrade, and fault recovery features. Features may have been phased into the product in an order better suited to the manufacturing bring-up effort. The customer environment, and the in-house manufacturing environment for that matter, was not so well understood by the SPS development team, causing a few client features to be implemented that proved unnecessary in practice. Finally, the SPS architecture could not accommodate management of the TFLOPS I/O subsystem nor could it completely account for management of the hardware cabling.

Discussion

While there are similarities between managing the TFLOPS system and managing a network of distributed PC desktops, they are certainly not the same. One aspect of the TFLOPS supercomputer that simplifies the management task comparatively is its fixed size and network topology.

Another simplifying factor is its homogeneity: most nodes share identical hardware, firmware, and system software. Unlike distributed PCs, these nodes do not have individual owners who may install arbitrary software packages or otherwise customize the local environment. Only one or a few different parallel applications run across all the machine's nodes at a given time.

Whereas PC network administrators may come from a wide variety of backgrounds and receive varying levels of training, TFLOPS administrators are literally hand-picked and acquire an in-depth knowledge of the machine. This even further simplifies the management burden.

On the other hand, some aspects of system administration become more difficult in the TFLOPS domain. The tightly-integrated, custom hardware design of the Intel TFLOPS supercomputer naturally tends toward more frequent system outages. The system software deployed is not so mainstream as that typically used on large-scale PC networks, leading to even less system stability. TFLOPS nodes lack their own monitors, keyboards, or mice, limiting visibility into the machine's operation. Furthermore, self-administering the management hardware and management software required effort on a scale not typically found in commercial networks.

Nonetheless, many aspects of administration remain clearly similar in both domains. Hardware faults must be detected and reported to any remote location the administrator requires. Firmware upgrade, emergency power-down, and remote reset capabilities are likewise required. The ability to trouble-shoot any single node is extremely useful, and the ability to isolate a node from the overall network is also valuable. Many of the features exported by the SPS live in this "common ground" between supercomputer and PC-based server management.

Conclusion

Compared to previous-generation management environments for Intel supercomputers, such as those for the Intel Paragon™ Supercomputer, SPS makes significant strides in feature offerings and overall usability.

The SPS usage model, based on an intimate knowledge of the server platform's internal hardware constructing and networking topology, has appeal in the scientific supercomputer market where the user base tends toward an "expert" knowledge of underlying implementation details of the system. More mainstream management products, typically built for heterogeneous, distributed commercial environments, often cannot know these details or choose to hide them from the users. Applications such as LANDesk Server Manager employ browsing metaphors in cases where the SPS environment requires more direct interfaces.

In most cases, leveraging vendor products such as the DMI Service Layer afforded clear gains for the project schedule, stability, and/or available features However, some off-the-shelf technologies proved a less-than-ideal fit for the SPS. The process of identifying potential sources of leverage and making build-or-buy decisions can have significant downstream effects on the project schedule and product quality.

It is reasonable to expect that some of the lessons learned from implementing management support for an ultra high-end platform can have cross-over benefits to the more mainstream Intel server product offerings. On one level, the SPS product should prove interesting to those developing DMI-based management software products. On another level, the SPS experience could prove useful to those interested in software systems that leverage off-the-shelf third-party components. The code base of SPS, being too closely tied to the Intel TFLOPS supercomputer's unique hardware characteristics, likely has little direct application in more mainstream environments. However, perhaps some of the SPS design principles, such as scalable communication, user interaction, and configuration management services, can be successfully applied.

Acknowledgments

The author thanks the following members of the development team for their support throughout the project: Ray Anderson, Johannes Bauer, Roy Larsen, Mouli Narayanan, Don Neuhengen, and Rajesh Sankaran.

A special mention goes to Linda Ernst who formulated the original SPS concept and succeeded in convincing people of its feasibility.

References

[1] DMI Specification., http://www.dmtf.org.

[2] VxWorks and Wind River Product Information, http://www.wrs.com/html/vxwks52.html. http://www.wrs.com/html/productindex.html.

[3] Joint Test Action Group IEEE 1149.1 Standard, http://ada.computer.org/tab/tttc/standard/s1149-1/home.html.

[4] Reliable Asynchronous Transfer Protocol RFC 916, http://globecom.net//ietf/rfc/rfc916.shtml.

[5] BasicScript and Summit Software Product Information, http://www.summsoft.com/html/products.htm.

[6] LANDesk Server Manager Product Information, http://www.intel.com/network/server/index.htm.

[7] Pragma Systems and TelnetD Product Information, http://www.pragmasys.com/TelnetD/

Author's Biography

Bradley Mitchell received an S.B. in Mathematics with Computer Science from M.I.T. and an M.C.S. in Computer Science from the University of Illinois, Urbana-Champaign. Before joining Intel in 1993, he was employed by General Dynamics Corporation. Bradley currently works in Oregon as a Senior Software Engineer in Server Software Technology. His e-mail address is bradley_mitchell@ccm.co.intel.com.