A Wireless Architecture For Performance Monitoring And Visualization On PDA Devices

Claudio Zunino Fabrizio Lamberti Andrea Sanna Bartolomeo Montrucchio

Dipartimento di Automatica e Informatica,
Politecnico di Torino, corso Duca degli Abruzzi 24,
I-10129 Torino (Italy)

{c.zunino,lamberti,sanna,montru}@polito.it

ABSTRACT

Information Visualization requires the merging of data visualization methods, computer graphics solutions and graphics interfaces design. In performance monitoring applications, the use of Information Visualization is essential to guarantee an effective and easy consultation of information. The employment of PDA devices could allow an effective monitoring thanks to the use of wireless technologies but their poor resources have limited their application until now. A wireless PDA-based architecture for cluster performance monitoring and visualization is presented. In this paper the issue of efficient data presentation on PDA devices has been addressed and compared to a traditional Web PC-based approach. The effectiveness of the proposed architecture in a real environment is shown.

Keywords: information visualization, PDA devices, wireless architecture, cluster performance monitoring, Web-based application.

1. INTRODUCTION

In a broad spectrum of disciplines a large amount of information has to be visualized. Information Visualization, is an important sub-discipline within the field of scientific visualization and it focuses on visual mechanisms designed to communicate clearly to the user the structure of information and improve the cost of access to large data repositories. Information Visualization enables people to deal with all of this information by taking advantage of visual perception capabilities of the human being. By presenting information more graphically it is possible for the human brain to use more of its perceptual system in initially processing information, rather than immediately relying entirely on the cognitive system. Some of the most important papers in the field are collected in [1].

An attractive field of research in Information Visualization is monitoring and management of parallel systems by means of advanced graphics user interfaces. Moreover, a further challenge is the employment of PDA devices in accomplishing these tasks. The monitoring operation is devoted to obtain a performance analysis; for instance, the user can know in real time the percentage of CPU used at each node, the amount of memory used for a specified application, as well as the network loading, the space available on the hard disk, and many other useful information. All these statistics allow to evaluate the behavior both of a specified application and of the hardware composing the system. On the other hand, management operations can include bootstrap and shutdown of every node as well as more complex commands such as task migration to accomplish load balancing, or security option setting.

Information has to be collected and presented to the user in an intuitive way; a set of suitable icons should be used in order to show CPU usage, memory and disk occupation, number of tasks and of logged users, network traffic, and so on. The monitoring architecture should be easily scalable (it has to be possible adding and removing nodes without affecting the monitoring system structure), network topology independent, and reliable. In this work the performance monitoring task is addressed through a wireless architecture. The proposed architecture has been developed in order to make performance monitoring information visualization possible even on mobile devices such as Personal Digital Assistant (PDA) based systems. PDA devices, that were originally designed as personal organizers, are are today effective pocket computers; in fact, by means of additional software, their functionalities are virtually unlimited. Moreover, a comparison with a traditional Web-based visualization architecture is provided.

Section 2 reviews the main visualization system devoted to cluster monitoring and summarizes nowadays PDA technical characteristics. Section 3 presents the proposed architecture in terms of the infrastructure for performance information collection and of the visualization interface for monitoring data presentation to the user. Section 4 presents the detailed implementation. Section 5 analyzes advantages and drawbacks of the proposed solution.

2. BACKGROUND

In this section, the main works known in the literature aimed to monitor and manage parallel/distributed architectures are reviewed (in subsection Cluster monitoring) and the most important characteristics of PDAs are highlighted (in subsection PDA devices).

Cluster monitoring-information collection and visualization

Performance data visualization techniques have evolved from 2D static to immersive virtual environments (a survey can be found in [2]). This section reviews the main works known in the literature focusing on solutions proposed for cluster monitoring.

A first solution to collect and display information on the status of a Unix-like machine is to use the Rstatd daemon. Rstatd provides information to remote hosts via the SunRPC [3] (Sun Remote Procedure Call protocol); statistics provided by Rstatd are shown in Table 1.

CPU times
Disk transfers
Page in/out
Swap in/out
Interrupts
Packets in/out, errors in/out, collisions
Context switch
Avenrun
Boot time
Current local time

Table 1: Statistics provided by Rstatd.

A remote client program such as Rsysinfo can query Rstatd servers and display information listed in Table 1. Unfortunately, information obtained by Rstatd could be not sufficient (for instance information about memory each node is currently using is not provided) and the efficiency of Rstatd is low as it consumes at least 12% of the CPU.

Supermon [4] attempts to overcome Rstatd drawbacks; Supermon is a tool that allows to simultaneously extract data from all machines in a cluster by a very simple character-oriented command set and response. The main characteristic of Supermon is that the set of commands can be easily used in any type of remote client program (Java, Perl, and so on). In particular, a Perl client is proposed in [4]; the Perl client makes a socket connection to Supermon and receives the set of variables to be analyzed at a specified sampling rate. Another Perl script creates graphics output and examines statistics.

Monitoring and management problems of Beowulf clusters [5] are tackled in [6]. The project SMILE (Scalable Multicomputer Implementation using Low-cost Equipment) allows to gather information from every node in real time, and monitoring and management can be achieved by using a set of APIs developed in C, Java, and Tcl/Tk. SMILE focus on small/medium clusters (between 4 to 64 nodes) and the set of APIs called RMI (Resources Management Interface) allows to monitor CPU and memory load. Similar to SMILE is NOW [7] that gathers and stores data in a relational database. An applet Java provides the graphics interface allowing users to monitor the system from any Internet node.

A monitoring tool entirely developed in Java is PARMON [8]. PARMON allows to monitor the entire cluster, a single node, a component of node, or activities of a single individual component. PARMON architecture consists of the parmon-client and the parmon-server; a server is running on all nodes to be monitored and the communication between the client and a server is performed by message passing. For effective monitoring, the concept of group is supported; a set of nodes forms a group and nodes are selected based on the allocation of resources to various user groups.

A tool specifically devoted to performance analysis of parallel applications is PABLO [9][10]. PABLO consists of two primary components: a portable software instrumentation and a portable performance data analyzer; PABLO can provide both traditional static/dynamic graphics icons and immersive worlds. Through a head-mounted display, three-dimensional sound cues, users can "explore" data concerning the behavior of massively parallel systems. An evolution of PABLO is SvPABLO [11] that supports the analysis of parallel applications written in C, Fortran, and HPF; furthermore, SvPABLO exploits hardware performance counters to capture the interaction between software and hardware.

Another *virtual* analysis toolkit is VIRTUE [12]; VIRTUE provides an immersive, collaborative performance visualization system (based on the SvPABLO instrumentation Toolkit) enabling users to steer and tune parallel and distributed applications during their execution. Mainly, VIRTUE provides a flexible and powerful toolkit for displaying arbitrary three-dimensional graphs: users can define the hierarchical structure, set the appearance, and associate data with graph attributes.

Other special purpose monitoring systems have been also proposed. For instance in [13] a monitoring software for NT machine clusters connected by HPVM (High-Performance Virtual Machine) [14] is proposed. High efficiency is obtained in [15] where monitoring problems related to distributed architectures based on the Myrinet Network Interface [16] are tackled.

PDA devices

The diffusion of Personal Digital Assistants is due to the more powerful devices we can buy today. The typical PDA architecture is composed by a input unit that is a stylus for navigation and data entry via a handwriting recognition system. The typical display size is 160x200 pixels, even if today PDAs reach 240x320 pixels of resolution. More expensive PDAs have a color display while the medium price devices have gray scaled display. Most PDAs have only the beeper for sound, but the newest have speakers for advanced functionality like MP3 players.

Most PDAs include an infrared port for wireless data transfer, while for synchronization with PCs use a serial port.

PDAs have memory in the range between 2 MB to 64 MB, but typically include slots for memory expansion, via Compact-Flash or Mini-Cartridge, or for mini hard disk. The processors have a 32bit architecture and a clock frequency from 50 to 200 MHz. Many PDA devices run on standard alkaline batteries while more expensive models have a rechargeable NiMH or lithium-ion batteries. The monochrome Palm series runs for weeks, while color models, like Compaq, consume more electricity and have 10-15 hours of operational time.

Most PDAs on the market are currently using either PalmOS or Windows CE operating system, but the latest Windows CE supports audio, colors and particular Java Runtime Environments like Jeode [17].

3. PROPOSED ARCHITECTURE

The system overall architecture can be decomposed in two parts:

- A set of middleware applications for performance information collection and storage;
- A Web-based visualization interface for performance monitoring data presentation to the user both on a PDA device and on a Personal Computer.

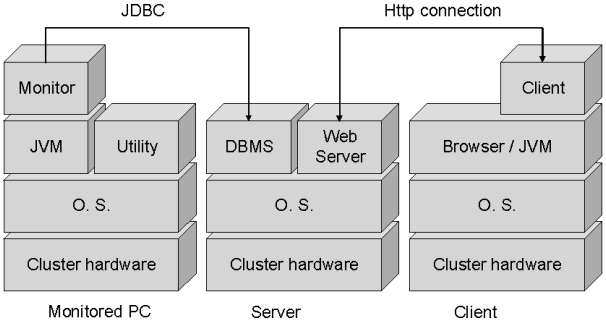

Figure 1: Software system architecture.

The performance statistics collection is performed by a Java application running on each monitored PC as shown in Figure 1, that uses some Linux typical programs (in the figure, the block named Utility). Information are stored in a remote relational database (running on the server). These data are made available by means of the use of a Web server (running on the server) where the HTML page with the code of the visualization interface is stored.
The visualization interface is based on a Java applet virtually connected to the remote database through the Internet, by means of wired and wireless communication links.
In particular, a wireless General Packet Radio Service (GPRS) [18] based communication environment has been designed, in order to allow the remote mobile monitoring through a PDA. The GPRS is a standard recently introduced from the European Telecommunications Standard Institute (ETSI) providing a packet-switched bearer service in the existing circuit-switched GSM network. The PDA devices has been connected to a Motorola t260 GSM/GPRS-enabled phone through its serial port. This allow the PDA to be always connected to the Internet with a maximum bandwidth of 40.2 kbps in download and of 13.4 kbps in upload. Furthermore, because of its true end-to-end IP nature (transparent IP), the GPRS phone provides the PDA with an instant IP packet access on-the-move, since the setup delays due to remote access over circuit switched modem connections are completely removed. The Web-based visualization interface has been evaluated also in a Local Area Network (LAN) and Wide Area Network (WAN) scenario using common PCs. It has to be noticed that the Internet protocols (HTTP, TCP, IP) required by the visualization applet works well both in the LAN/WAN and GPRS environment, requiring only few adjustments to the communication routines when used over GPRS [19].

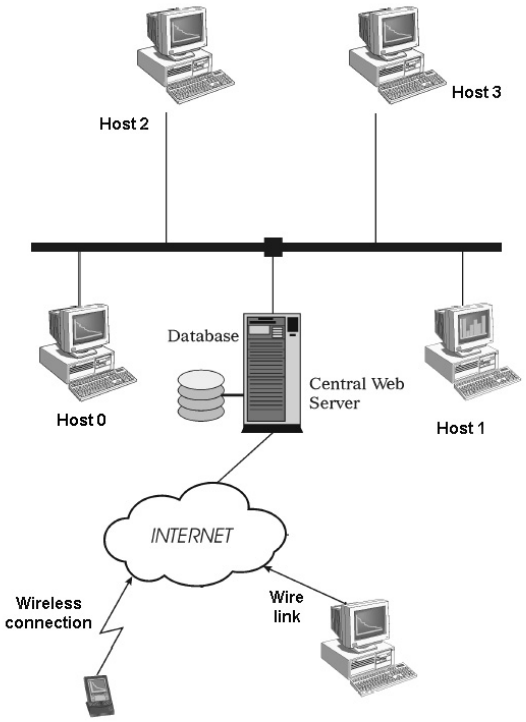

Figure 2: Hardware/network system architecture.

The proposed architecture is independent of the cluster topology. In Figure 2 a typical configuration of the system with four monitored hosts, a server and two clients is shown. The database/Web-server can be one of the cluster PCs.

4. IMPLEMENTATION AND RESULTS

The proposed architecture has been tested on a real network environment. The test environment consists of a cluster of PCs connected over a FastEthernet network through a switch. The collection of performance monitoring data has been developed in Java following a client-server approach. To display in a user-friendly manner the large amount of data being monitored, both the PC and the PDA based graphics user interfaces described in Section 3 have been implemented and tested.
As reference architecture, we have used a cluster composed by 4 Personal Computers with a 400 MHz CPU, 256 MB of RAM and a 6 GB hard disk. The operating system is a Linux RedHat 6.2. The PC connection consists in a FastEthernet LAN based on a 3COM eight-ports switch. The cluster was monitored during the execution of an intensive parallel 3D rendering application.

Statistics collection system

The middleware for performance information collection has been implemented using the Java programming language: this choice is due to the cross-platform property of Java. On each PC, a Java application monitors the machine performances using typical Linux programs: *top* for processes information, *users* for users information and *netstat* for network statistics. The application stores the collected information into a remote database using the JDBC Java APIs. In particular, a Postgresql database server has been used.
The system is able to monitor:

- CPU: load information;
- Memory: the percentage of free and used memory and information about the swap memory;
- Hard disk: File System information about each partition;
- Network: due to the fact that the proposed architecture has been deployed in a cluster environment, network performances are an important parameter for evaluating system behavior;
- Processes: the number of processes and the CPU load for each one;
- Users: information about the number of users connected to the PC.

PC-based performance visualization

In deploying the monitoring interface, a Web-based approach has been used: the visualization interface consists in a Java applet developed using the Java Development Kit 1.1.6 (to guarantee an high level of portability). By means of a Java-enabled Web browser, the user is able to monitor the cluster in a simple way. The interface is implemented using the Java AWT (Abstract Window Toolkit) API for the visualization and the JDBC API for the database connection. There is a thread for each monitored parameter. As shown in Figure 3, there are two main panels on the top of the window: the first is the Settings panel, that allows the user to adjust some visualization parameters (such as the font size and the color of the text) or set some alarms on particular parameters. For example, it is

possible to set a warning on the percentage of free memory: if the free memory is not enough, a warning message is shown to the user.

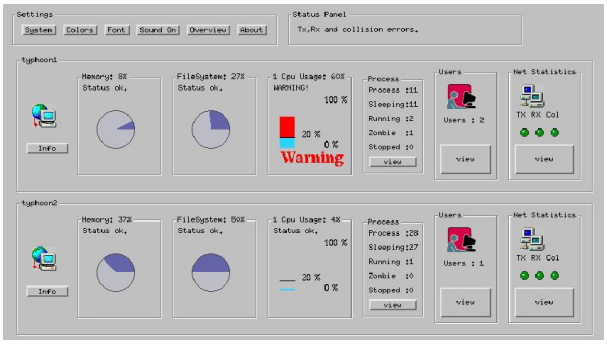

Figure 3: Applet panels.

The status panel shows some information about the status of the application. The cluster information are shown in a variable number of panels (according to the number of the monitored hosts); in the figure, the number of PC under analysis is two. Each panel is made up of seven sub-panels. The first one shows the status of the connection with the server and contains an Info button that, when clicked, opens a window where the PC characteristics are highlighted. The second and the third sub-panels show the percentage of free memory and the file system usage respectively. The fourth sub-panel shows the CPU status; in particular, for the first PC a warning message is displayed. The fifth, sixth and seventh sub-panels show the processes, users and network related information respectively. In these sub-panels, a button allows the user to open a windows where more detailed information are shown: in Figure 4, the processes running on the first PC are shown.

PDA-based visualization interface

The visualization application has been developed and tested also on a Personal Digital Assistant device. The PDA that has been selected is a Compaq iPAQ H3630 equipped with the Microsft PocketPC operating system. Some basic features are: 206 MHz Intel StrongArm CPU, 4096 colors TFT LCD display, 240x320 pixels (2.26 x 3.02 inches) resolution touch screen, 32 MB RAM and 16 MB Flash ROM. The graphics user interface for performance monitoring described in the subsection above has been developed as a Java applet in order to achieve a high degree of portability, even on PDA devices. Nevertheless, the Microsoft PocketPC operating system running on the selected Compaq iPaq PDA does not support Java natively. A third-party Java Virtual Machine (JVM) implementation based on Sun PersonalJava Application Environment (PJAE) specifications [20] has therefore to be installed on the device before running either Java applets or applications. A PJAE is a Java Runtime Environment (JRE) suitable for PDA, smart-phones and set-top boxes that executes software written in the Java programming language. Among the different PJAE implementations available on the market, the Insignia *Jeode Runtime* [17] has been selected for its high adherence to Sun PersonalJava 1.2 specifications and for its advanced performances in executing Java byte-code with respect to other implementations. Jeode Runtime can be used both as Pocket Internet Explorer plug-in to run Java applets from a Web page and as a stand-alone JVM to run Java applications. It has to be remarked that not all the API available in the Sun Java Development Kit (JDK) environment are available to PDA developers through a PJAE platform. This issue had not to be addressed while developing the PDA-based monitoring visualization interface because all the API used in the Web graphics user interface have been selected from those available in the Jeode set. Moreover, while deploying Java code

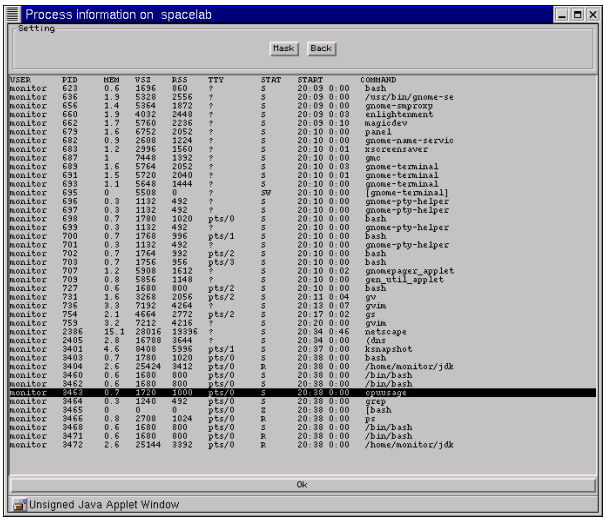

Figure 4: Detailed processes information.

to be executed on a PDA running a PersonalJava JVM, other important issues have to be taken into account:

- PDA devices are characterized by limited computation resources and, as a consequence, execution speed is often much more slow than on a typical personal computer;
- PDA devices usually have a small display with a limited palette;
- handheld devices (and in particular PocketPC based PDAs) allow only a single window to be displayed at one time;
- users have to use a pen over a touch-screen display instead of a traditional mouse pointer.

Therefore, while designing the PDA Java applet for monitoring data visualization, the previous issues have been taken into great consideration. Monitoring information update rate has been adjusted with respect to the PC-based applet in order to reduce the number of screen repaints. The layout of the whole interface has been completely re-designed. Instead of using a large window containing the information related to all the hosts being monitored, a opening window (Figure 5) presenting the cluster network architecture has been introduced. Several buttons allow to change some configuration parameters, such as the alarm levels and the font or color size of the text. When the user tap on one of the icons representing the cluster hosts, a window with the monitoring information for the selected host is displayed (Figure 6).

Figure 5: Screenshot of the main windows for host selection in the PDA-based visualization interface.

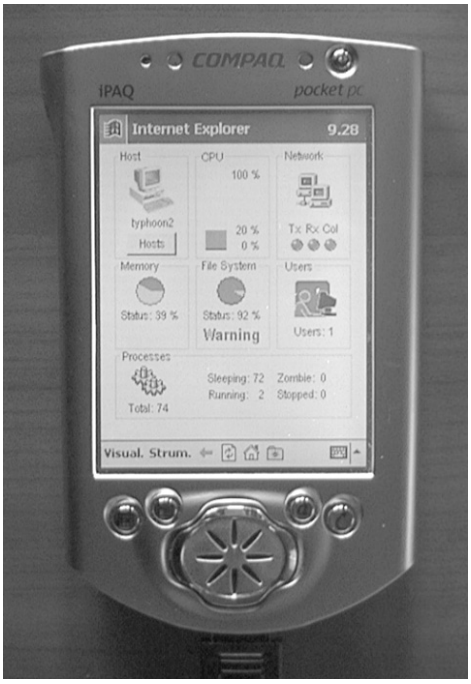

Figure 6: Screenshot of the window for performance monitoring in the PDA-based visualization interface.

The window layout has been adjusted so that all the relevant information are still available and are shown in a compact and user-friendly way. In Figure 7, a zoom on the icons used in the monitoring window is shown. The buttons of the PC-based applet have been removed. By tapping on the icon representing a group of monitoring information, the current window is replaced by another window where the complete set of data is shown. In Figure 8, the window containing the processes-related detailed information is reported.

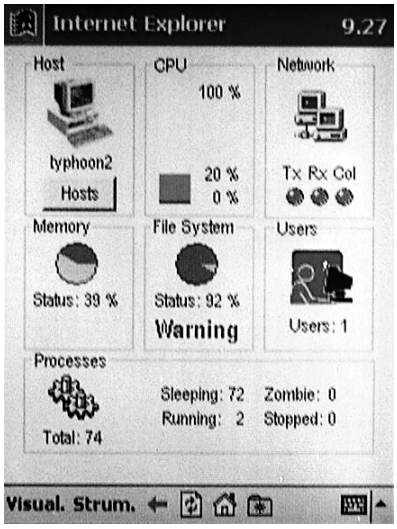

Figure 7: Zoom on the icons used in the performance monitoring window

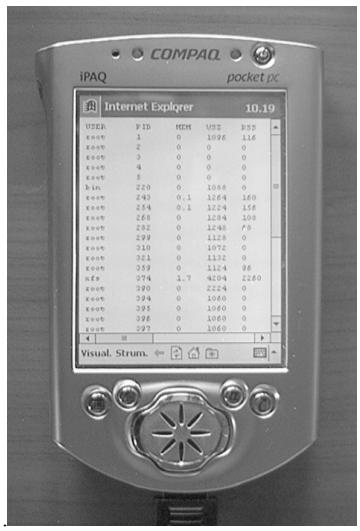

Figure 8: Screenshot of the window for detailed processes-related information visualization in the PDA-based interface.

5. REMARKS

An important characteristic of this proposed architecture is that different applications through different devices could access the information. This concept is the same of typical trading on line or home banking applications where the user can operate on the system via PC, PDA or cellular phone.

The system architecture is scalable and suitable for a medium/small cluster because each PC stores the statistics information in a single shared database. The reliability is the typical of a star architecture: while a crash of a monitored node doesn't modify system behavior, a crash on the database server affects the stability of the whole system. Finally, due to the limited flow of information from the PCs to the database and from the database to the visualization interface, network load is not of great concern in the evaluation of the proposed architecture.

6. CONCLUSION AND FUTURE WORK

In this paper, a wireless PDA-based architecture for cluster performance monitoring and visualization has been presented. The use of a PDA and of a GPRS-enabled phone allows a remote and mobile continuous monitoring by means of a user-friendly graphics interface. The simulation results on a real test environment show the effectiveness of the proposed solution.

Future work will be aimed to further extend the management operations available to the PDA user by introducing shutdown, reboot and task migration functionalities. Other efforts will be devoted to the definition of new and more effective icons for data representation and to the development and evaluation of a more efficient implementation of the visualization application based on native Microsoft Pocket PC API.

The use of a central database server, as explained in Section 5, may be the critical point for the reliability of the system. A possible improvement could be the introduction of a mirror for the database server: if a crash of the main server occurs, an automatic recovering program could activate the Second database.

7. ACKNOWLEDGMENTS

This project is supported by the Ministero dell'Istruzione dell'Università e della Ricerca in the frame of the Cofin 2001 project: elaborazione ad alte prestazioni per applicazioni con requisiti di elevata intensità computazionale e vincoli di tempo reale.
We thank Ing. Alessandro Schillaci for his support and suggestion in the develop of this work.

8. REFERENCES

[1] S.K. Card, J.D. Mackinlay, and B. Shneiderman, *Readings in Information Visualization Using Vision to Think*, Morgan Kaufmann Publishers, 1999.

[2] D.A. Reed, M.J. Gardner, and E. Smirni, Performance visualization: 2D, 3D, and beyond, *Proceedings of the IEEE International Computer Performance and Dependability Symposium*, 1996.

[3] J. Bloomer, *Power Programming with RPC*, O'Reilly, 1992.

[4] Minnich R.G. and Reid K. Supermon: High-Performance Monitoring For Linux Clusters, *The Fifth Annual Linux Showcase and Conference*, 2001.

[5] Sterling T., Becker D.J., Savarese D., Dorband J.E., Ranawake U.A, and Packer C.V. BEOWULF: A Parallel Workstation For Scientific Computation, *Proceedings of the 24th International Conference on Parallel Processing*, 1995.

[6] Uthayopas P., Paisitbenchapol S., and Chongborirux K. Building a Resources Monitoring System for SMILE Beowulf Cluster, *Proceedings of HPCAsia*, 1998.

[7] Anderson E. and Patterson D. Extensible, Scalable Monitoring for Clusters of Computers, *Proceedings of the 11th Systems Administration Conference*, 1997.

[8] Buyya R. PARMON: A Portable and Scalable Monitoring System for Clusters, *International Journal on Software: Practice & Experience*, vol. 30, pages 1-17, 2000.

[9] Reed D.A., Aydt R.A., Noe R.J., Roth P.C., Shields K.A., Schwartz B.W., and Tavera L.F. Scalable Performance Analysis: The Pablo Performance Analysis Environment, *Proceedings of scalable Parallel Libraries Conference*, 1993.

[10] Reed D.A. Experimental Performance Analysis of Parallel Systems: Techniques and Open Problems, *Proceedings of the 7th International Conference on Modeling Techniques and Tools for Computer Performance Evaluation*, pages 25-51, 1994.

[11] De Rose L. and Reed D.A. SvPablo: A Multi-Language Architecture -Independent Performance Analysis System, *Proceedings of the International Conference on Parallel Processing*, 1999.

[12] Shaffer E., Whitmore S., Schaeffer B., and Reed D.A. Virtue: Immersive Performance Visualization of Parallel and Distributed Applications: Immersive Performance Visualization of Parallel and Distributed Applications, *IEEE Computer*, pages 44-51, 1999.

[13] Sampemane G., S. Pakin, and Chien A.A. Performance Monitoring on an HPVM Cluster, *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications*, 2000.

[14] Chien A.A., Pakin S., Lauria M., Buchanan M., Hane K., Giannini L., and Prusakova J. High Performance Virtual Machines (HPVM): Clusters with Supercomputing APIs and Performance, *Proceedings of the Eighth SIAM Conference on Parallel Processing for Scientific Computing*, 1997.

[15] Liao C., Martonosi M., and Clark D.W. Performance Monitoring in a Myrinet-Connected Shrimp Cluster, *Proceedings of the 2nd SIGMETRICS Symposium on Parallel and Distributed Tools*, 1998.

[16] Boden N.J., Cohen D., Felderman R.E, et al. Myrinet: A Gigabit-per-Second Local Area Network, *IEEE Micro*, 1995.

[17] Insignia Jeode Runtime: PDA-based PersonalJava Application Environment. Online WWW: http://www.insignia.com/.

[18] Ferrer C., Oliver M. Overview and capacity of the GPRS (General Packet Radio Service) *Personal, Indoor and Mobile Radio Communications, 1998. The Ninth IEEE International Symposium on* page(s): 106 - 110 vol.1 Sept. 1998

[19] Meyer, M. TCP performance over GPRS *Wireless Communications and Networking Conference*, 1999. WCNC. 1999 IEEE Page(s): 1248 -1252 vol.3

[20] Sun PersonalJava Application Environment (PJAE). Online WWW: http://java.sun.com/products/personaljava/.