Managing a Grid of Computer Laboratories for Educational Purposes

Luis C. E. De Bona Marcos Castilho Fabiano Silva Daniel Weingartner
Luis H. A. Lourenço Bruno C. Ribas

Centro de Computação Científica e Software Livre – C3SL
Universidade Federal do Paraná
E-mail: {bona,castilho,fabiano,danielw,lhal04,ribas}@c3sl.ufpr.br

Abstract

This paper presents a grid-based model for managing hundreds of distributed educational computing laboratories. The aim is to dispense the need of specialised staff in each place whilst maximizing the laboratories uptime. The model was implemented in order to manage more than two thousand laboratories spread across the entire public school network of Paraná State (Brazil). Practical results are also presented validating the model.

1 Introduction

Computer resources and access to the Internet are nowadays considered necessary tools in a student's learning process, as well as a considerable help for teachers to prepare their courses. This fact, associated to the cost reduction of computer hardware, led to the creation and expansion of digital inclusion policies, aiming to provide computing laboratories for every school.

However, the lack of specialized workforce to manage and maintain these laboratories frequently ruins such efforts, because the computers get obsolete, the software doesn't work properly, computers get infected, among other reasons. In public schools this situation is worsened by the lack of funds for frequent updates.

The impossibility of having an expert to manage each school in a huge public network demands an effort in order to define a new administration model. In an ideal situation, educators and students should only worry about the educational use of the system, and not about its technical details. All system configuration and management tasks should be performed either automatically or by experts.

This paper presents a model that allows the administration of thousands of computing laboratories with minimum human intervention. In this model, laboratories are interconnected forming a computational grid [1] that is activelly monitored, with information collected on each node and transmited to a central processing station, and management or maintainance tasks can be automatically assigned to each laboratory in a secure way.

Based on autonomic computing concepts [2], the model intends to reduce the maintenance and management costs, whilst providing a reliable and updated computing environment. The idea is to ensure the self-configuration, self-optimization, self-recuperation and self-protection [3] of all connected laboratories.

The model also foresees two kinds of human interventions for administrative tasks: local execution of simple tasks, by an ordinary user, through user-friendly interfaces; and remote execution of critical or unpredictable tasks, by a team of experts.

This model was implemented by the Paraná Digital Project (PRD) in more than 2100 public school laboratories, and is based on GNU/Linux software, under Free Software licence (GPL), with all code freely available on the internet[1].

The following sections describe the proposed model in more details (Section 2), and present the PRD project as a successfull implementation (Section 3). Section 4 contains concluding remarks.

2 A Model for Maintenance and Management of Computing Laboratories

Computing laboratories are increasingly used in schools and other educational institutions. To keep these laboratories in working conditions is a challenge,

[1] http://www.c3sl.ufpr.br/prd

specially considering the lack of people with the needed expertise.

Therefore, we propose a model that dispenses the use of specialized staff in each laboratory, whilst minimizing downtime due to factors such as: external attacks, failures in software components, bad configuration of the system or its applications and the lack of preventive maintenance of hardware.

The model considers that many geographically distributed computing laboratories can be interconnected forming a computational grid. It also considers that the hardware, operating system and main applications of these laboratories is fairly homogeneous.

Tasks needed to install and manage such a computing laboratory are classified into local and global management tasks. Local management tasks are related to specific aspects of each laboratory (such as user management), and tasks that demand physical access to the hardware. These tasks must be translated into high level, simple decisions, offered by a user-friendly interface, that can be operated by an ordinary user called *local manager*.

The global management tasks are the ones executed to guarantee that the laboratory offers the expected services, with maximum performance. These tasks are determined in a global way and uniformly executed in the grid. Hence, most of these tasks can be automated, so that a small group of specialized system administrators is able to manage hundreds of laboratories.

All critical tasks are considered to be global, and can not be locally modified or executed. Though this imposes strong restrictions, it allows the local manager to be a user with no system expertise, and reduces occurrence of human mistakes that compromise the laboratory's operation.

In order to implement these global management tasks, the proposal is to use the concepts of autonomic computing, which in essence consists of the system's self-management, and can be analyzed in four ways: self-configuration, self-recuperation, self-protection and self-optimization.

2.1 Self-Configuration

The capacity of self-configuration of an autonomic system involves the adaptation of new components or new execution environments without a significant human intervention. Self-configuration is a continuous process aiming to keep the system configured under varying time and environment conditions [4].

The configuration policies are defined for the entire grid by a core management team, who specifies what are the system's desired features, which softwares should be installed, and what should be restricted or not allowed.

In order to reduce hardware and management costs, as well as simplify the implementation of the configuration policies, the use of a computing model based on graphic terminals, so-called X-terminals [5] is proposed. It basically consists several diskless workstations that access one or more processing servers. The software and configuration parameters of these terminals are determined at the server, a computer with large disk, memory and processing capacity, being the only machine that must be configured and maintained.

The initial installation of the server can be done with a CD-ROM or a DVD containing a standard system image. When the installation is completed, the system needs to be updated through the network, in order to ensure that it has the latest version of all software.

The X-terminals must be prepared to automatically recognize and configure peripheral hardware such as printers and removable media. Sharing policies for these peripherals can be defined by the user or by the local manager, depending on their intended use.

The servers' self-configuration is based on modern software package management systems [6], and unified package sources (mirrors). The packaging systems allow automated installation and configuration, but usually present questions during these processes. The grid-based model improves on the classical actualization systems by providing default answers for all questions that might be posed by all packages, eliminating local interactions. In exceptional cases high level, simplified questions can be presented to the local manager.

After the server is up and running it must frequently check for updates at the central mirror, in order to install new software, new versions of existing packages, patches that address security and maintenace issues, configuration changes and other relevant changes. This automated actualization process minimizes service disruption and data loss [7].

2.2 Self-optimization

Modern computing systems allow the configuration of a collection of parameters that significantly impact on their overall performance. Nevertheless, this is an extremely complex tasks and specially demanding for human operators. Autonomic systems constantly monitor such parameters and seek to improve their own performance, identifying and exploiting opportunities to be more efficient [8].

Thus, a monitoring system is a fundamental building block of a self-optimization system. Data provided by monitoring systems can be used by expert managers to optimize system parameters, which then can be applied globally in the grid through the self-configuration system. This turns out to be simpler when the laboratories have similar configurations. Besides, a monitoring system allows the identification of possible failure points, and can be used to determine the cause of failures after they occurred.

The historical and comparative analisys of the load and performance metrics of the laboratories can also be used to optimize the system. This analisys can, for example, help to scale the hardware needed in each laboratory, plan a hardware upgrade given the estimate of load increase or find discrepancy of performance that must be analised by experts.

Last but not least, the monitoring system might keep a local copy of the information it generates, but must regularly send it to the central management core, where data from the whole grid is processed, analysed and made available on a web page to allow public access.

2.3 Self-recuperation and Self-protection

Failures in a system can cost many weeks of a manager's work to diagnose and repare. Autonomic systems automatically detect, diagnose, treat and prevent problems due to bugs or hardware failures, leaving minimal decisions to the manager, and substantially reducing the recovering efforts [9, 10].

The downtime of computing laboratories can be drastically reduced when certain aspects of the hardware and the operating system are tracked. The major aspects to be monitorated are: memory, hard disk and temperature sensors. In case of any abnormal behaviour, automatic alarms should turn on, so that maintenance can be performed.

One aspect that deserves special attention is the hard disk, since it is one of the most failure-prone hardware component. The hard disk's self monitoring facility (SMART) tracks disk parameters such as temperature and number of Hardware ECC Recovered blocks, which are good indicators of failure probability [11]. Data provided by the hard disk's SMART facility might be used to indicate replacement need ahead of any failure. Redundant Array of Inexpensive Disks (RAID) should also be used to easy the replacement of hard disks and to avoid the loss of data in case of an unexpected disk failure.

Downtime can be minimized at the operating system level by verifying the filesystem's integrity. Damaged or corrupted files might compromise the operation of the whole system. So, it is necessary to perform a periodic integrity verification for all the files. Once a corruption is detected, the system must automatically recover the file. This can be done through an automatic reinstallation of the software package that contains the damaged file(s) or by recovering a from a backup.

In case a complete system reinstallation is needed, a recuperation process tries to reinstall the system without loss of user data. Should the reinstallation process stop or fail, the grid-model provides a bootable recovery CDROM for the computing server, that boots the server into an assistance mode, allowing the remote intervention of an expert manager. Every remote access to the server should be done through an encrypted SSH connection.

3 Paraná Digital Project

The Paraná Digital project (PRD) provides a huge testbed for the proposed model. It consists of a partnership between Paraná State Secretary of Education (SEED)[2], the State Computer Company of Paraná (CELEPAR)[3], the energy company COPEL[4] and the Federal University of Paraná (C3SL/UFPR)[5].

The aim of the project is to provide every public school of the Paraná State (from kindergarten to grade 12) with a computing laboratory to access educational tools (such as the Educational Portal[6] and Moodle[7] software). The Paraná State has approximately 1,500,000 students, 57,000 teachers, in 2,100 schools distributed over 399 cities over 199,314 km^2. A more complete survey of the PRD project can be found at [12].

3.1 The PRD architecture

The PRD network is organized as a grid and employs the proposed management model in order to minimize human intervention on schools. Hence, the decision was to model a uniform computer network, managed by one control center located in the city of Curitiba. It is called *the management core* (or just *Core*), and is responsable for the grid's global policies and global management tasks, as described in the previous section.

Each school has a laboratory composed of 20 X-terminals and a special computer, called the *school server*. The latter acts simultaneously as processing and storage unit, gateway to the network, firewall,

[2]http://www.seed.pr.gov.br
[3]http://www.celepar.pr.gov.br
[4]http://www.copel.com
[5]http://www.c3sl.ufpr.br
[6]http://www.diaadiaeducacao.pr.gov.br
[7]http://www.moodle.org

and access point to the Core. It consists of a dual-processed machine, with 2GB of RAM, and two SATA hard disks arranged in a RAID 1. It runs a Debian[8]-based GNU/Linux distribution, and all servers have the same software packages installed.

The network connection from each server to the Core is provided by the electricity company COPEL through a private network (VPN) of optical fibers (70%) and satellite connections (30%). At the Core, a proxy-controlled connection to the internet is provided. Figure 1 illustrates the PRD network architecture.

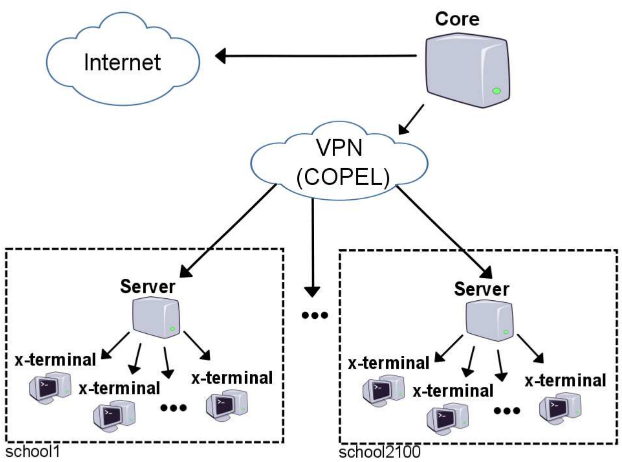

Figure 1: Paraná Digital Network Architecture

The first laboratory was installed in June/2006 and, as of August/2008, 2,126 schools were operational. The management team at the Core is composed of 12 highly trained Unix managers, taking care of all software-related issues from the entire network (approximately 44,000 workstations).

3.2 System Installation

The installation of a new computing laboratory is a straightforward procedure. After the hardware is installed by the reseller, a person called the *local manager* inserts a CD-ROM into the computing server and turns it on. Some few questions are asked in order to identify the school being installed (selected from a predefined list, with predefined IP addresses) and a password is set to controll locally managed tasks. The server then installs the operating system from the CD-ROM. After a reboot, it connects to the Core, performs any needed software upgrade, and is ready to use.

The X-terminals do not need any software to be installed, since they boot over the network, loading all their configuration from the server. The local manager only has to register them by using a user-friendly software. To use the system, a regular user must have an account created by the local manager.

It is worth noting that, despite performing some typical *root* administrative tasks, the local manager does not have root powers (i.e. is not the root user). All the local manager does is execute a management software that writes its output into a predefined directory, which in turn is automatically read by the system and executed. This way, only the Core managers and automatic scripts have root powers, ensuring that critical tasks are globally defined on the grid.

3.2.1 System Recovery

In case of severe system failure, the same CD-ROM used to install the system can be used to recover the installation. The PRD project has conceived two ways of recovering a corrupted local system: the automatic recovery and the remote help. In case of an automatic recovery, the local manager boots up the school server with the installation CD and chooses the recovery option, which formats only the root filesystem, preserving user data and local information. A collection of specially developed scripts saves important files (e.g.. passwords and local configuration), restoring them after system reinstallation.

The remote help is a last resort in case the previous method fails. The school server boots up using the PRD CD-ROM, and connects to the network even in the case of hard disk errors. Then, by remote SSH login, the Core managers can access the server to find (and eventually fix) the problem, and/or try to backup critical data. Only in case of severe disk error is the "start from scratch" method necessary. Even then, the local manager might be able to record an automatically generated CD-ROM image, containing a "snapshot" of the installed system with all its current configurations.

3.3 System Upgrade

Frequent system upgrades are necessary to provide new functionalities, address security problems and propagate new software, tools, or policies from the Core. On a huge grid such as the PRD network, server upgrades must occur automatically, and this can be done in two ways: automatic daily upgrades and triggered upgrades.

The daily automatic upgrade is based on Debian's *apt-get* tools [6]. Every night, each school server looks at the Core Debian mirror for new software packages. Since it is a non-interactive procedure performed by

[8] www.debian.org

the entire grid, the package configuration scripts must be throughly tested and completely sound. This is ensured by the management team at the Core. Besides, the single packages repository ensures that all servers will install exactly the same softwares.

Hence, in a normal situation, it is quite simple to propagate a new tool or configuration over the entire network, by just releasing the new Debian package(s) in the PRD Debian mirror at the Core, and waiting for the automatic upgrade. However, depending on the seriousness of the security breach, the Core can force all school servers to perform an instant upgrade. If some schools are not connected, the Core registeres it, and forces the upgrade as soon as the network link turns on. Such situations occurr when a bug might compromise the server's security, as was the case with the Kernel exploit that allowed an ordinary user to become root [13].

Triggered upgrades might also be used to perform larger changes on the system, such as the upgrade from the oldest stable Debian version called *sarge* to the new one called *etch*, which occurred in February 2008. In this case, the upgrade was sensitive because the server had to reboot, and therefore the Core managers decided to perform it on a more controlled manner, schedulling the upgrade for few school servers at a time, and ensuring it worked properly.

3.4 System Monitoring

Monitoring is an essential feature of autonomic systems, and provides information to allow the system's self-optimization and self-recuperation. Actually, monitoring tools are at the heart of the PRD network, since they reveal the real state of the whole grid. In the PRD model, there are two different systems providing strategic information: the statistics center and the diagnosis system.

3.4.1 Statistic Center

The Statistic Center consists of a web site[9] containing strategic information collected from each school every night. It is used mainly to allow an overview of the network's growth and provide the State managers with data concerning the laboratories usage, allowing to plan further developments of the projet.

Laboratories usage data is automatically saved at the school servers and the Core pulls that data, storing it in the a central database. The visualization of the information is done using Pentaho System[10] facilities. The type of collected data depends on the Paraná State Secretary of Education's demands, but usually includes: uptime, memory use, load average, number of schools, number of users, online time, among others, providing strategic information for decision support.

Figure 2 shows the number of computing laboratories installed since September 2007, as well as the monthly average of existing user accounts and login time (both divided by 1,000 to fit the scale). The graph shows a steady increase in the number of labs and user accounts, while the login time steeply decreases on vacation periods.

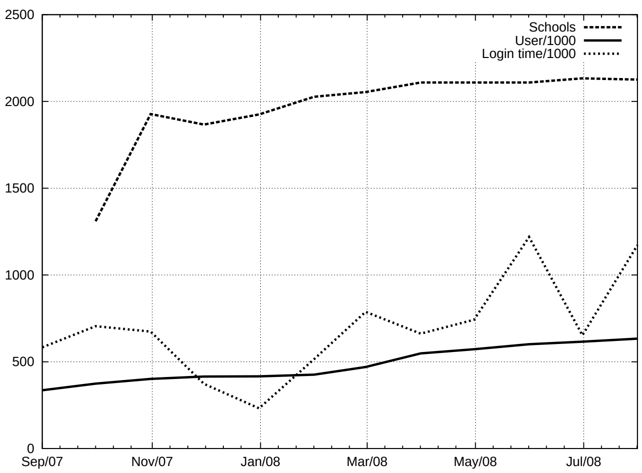

Figure 2: Graph showing number of computing labs, user accounts and login time on the PRD grid.

3.4.2 Instant Diagnosis System

The second monitoring tool is the *Instant Diagnosis System*[11], conceived to provide real time diagnosis of the network's server status. It detects possible faults in the laboratories and triggers alarms to the Core. Currently, it is capable to detect whether the schools are online, identify a system with old software versions, inconsistencies in a server's filesystem, checksum errors on installed software, missconfigured hard disk arrays, status of the server's nobreak, among many others.

This tool, combined with a good collection of scripts, allows the management team at the Core to prevent critical failures, either by acting remotely upon a problem, or by timely providing hardware replacement.

[9] http://www.prdestatistica.seed.pr.gov
[10] http://www.pentaho.com
[11] http://yoda.c3sl.ufpr.br/SDI

3.4.3 Manual System Inspection

Frequently the monitoring tools show that some schools behave abnormally, having a considerable decrease in performance or uncommon errors, whose cause is not easily identified. In such cases, the Core manageres can log remotely via SSH into the servers to manually inspect the server's behavior.

However, if some problem is discovered or an improvement implemented, it should never be manually installed on the server. Since the grid is highly homogeneous, all changes should happen on a global manner, usually by providing new or upgraded software packages to the entire grid.

4 Conclusion

This article presented a model that allows the administration of thousands of computing laboratories with minimum human intervention. In this model, laboratories are interconnected forming a computational grid that is activelly monitored, with information collected on each node and transmited to a central processing station, and management or maintainance tasks can be automatically assigned to each laboratory in a secure way.

A large scale implementation of this model is provided, in which a management team composed of 12 highly trained Unix managers is able to controll all software-related issues from the entire network (approximately 44,000 workstations) on more than 2,100 schools over all the Paraná State.

Acknowlegements

This research was supported by UGF/SETI-PR (CV31/03,CV02/07,CV02/08)[12]. We would also like to express our gratitude to the members of the C3SL Research Group[13] who activelly participated in these projects.

References

[1] I. Foster and C. Kesselman, "Computational grids," pp. 15–51, 1999.

[2] D. M. C. Jeffrey O. Kephart, "The vision of autonomic computing," IEEE, Ed., vol. 1, no. 10, jan 2003, pp. 41–50.

[3] M. R. Nami and K. Bertels, "A survey of autonomic computing systems," in *Proc. 3rd International Conference on Autonomic and Autonomous Systems*. IEEE Computer Society, 2007, p. 26.

[4] G. Tesauro, D. M. Chess, W. E. Walsh, R. Das, A. Segal, I. Whalley, J. O. Kephart, and S. R. White, "A multi-agent systems approach to autonomic computing," in *Proc. 3rd Intl. Joint Conference on Autonomous Agents and Multiagent Systems*. IEEE Computer Society, 2004, pp. 464–471.

[5] A. E. Socarras, "Anatomy of an x terminal," *IEEE Spectrum*, vol. 28, no. 3, pp. 52–55, 1991.

[6] G. N. Silva, "Apt-howto," 2005. [Online]. Available: http://www.debian.org/doc/manual/apt-howto/

[7] S. Potter and J. Nieh, "Reducing downtime due to system maintenance and upgrades," in *LISA '05: Proc. 19th conference on Large Installation System Administration Conference*. USENIX Association, 2005, pp. 6–6.

[8] M. Salehie and L. Tahvildari, "Autonomic computing: emerging trends and open problems," *SIGSOFT Softw. Eng. Notes*, vol. 30, no. 4, pp. 1–7, 2005.

[9] M. Parashar and S. Hariri, "Autonomic computing: An overview," in *Unconventional Programming Paradigms*. Springer, 2005, pp. 247–259.

[10] R. Sterritt and D. Bustard, "Towards an autonomic computing environment," in *Proc. 14th Intl. Workshop on Database and Expert Systems Applications*. IEEE Computer Society, 2003, p. 699.

[11] E. Pinheiro, W.-D. Weber, and L. Barroso, "Failure trends in a large disk drive population," in *Proc. 5th USENIX conference on File and Storage Techonogies*, 2007.

[12] M. Castilho, M. Sunyé, D. Weingaertner, L. Bona, F. Silva, A. Direne, L. Garcia, A. Guedes, and C. Carvalho, *Open Source for Knowledge and Learning Management: strategies beyond tools*. Idea Group Inc., 2007, ch. Making Government Policies for Education Possible by Means of Open Source Technology: a successful case, pp. 343–368, iSBN: 1-59904-117-0.

[13] W. Pitcock, "Linux 2.6 mmap() local root exploit," 2008. [Online]. Available: http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=464953

[12] `http://www.seti.gov.br/ugf`

[13] `http://www.c3sl.ufpr.br`