ENABLING PREOS DESKTOP MANAGEMENT

Tiago Cruz, Paulo Simões
CISUC – Dep. Eng. Informática, University of Coimbra
3030 Coimbra – Portugal
{tjcruz, psimoes}@dei.uc.pt

Abstract: Desktop management is probably the most resource-consuming task for the typical operations and support team, regardless of being frequently overlooked as *not as complex or specialized* as core network operations and management. Nowadays this scenario is even worst, since the increasing number and complexity of desktop systems was not matched by satisfactory management solutions – despite the relative success of products such as Intel's Landesk or Microsoft's SMS.

In order to address this problem, we are exploring a different approach to desktop management, through the design and implementation of the `OpenDMS` management framework. This open source framework differs from available products in several points, such as earlier remote management mechanisms (prior to operating system load), incorporation of existing open standards, a network-centric architecture, operating system neutrality and tighter integration between traditional PCs, Thin Clients and Network PCs.

In this paper we discuss the current status of desktop management solutions and we present an overview of the `OpenDMS` approach, including its most relevant technical foundations and an application scenario.

Key words PC networks, Desktop Management, OSS, Open-Source Tools

1. INTRODUCTION

Processing power was gradually taken off from central mainframes and distributed over each user's desktop, in the form of personal computers (PCs). However, this paradigm shift does not come without a price tag: more PCs to manage; hardware and software with ever-increasing complexity; more users requiring support; and more network requirements. Dropping prices and increasing user-friendliness made desktop PCs ubiquitous. Nevertheless, acquisition prices represent just a small fraction of the total cost of ownership (TCO) of such systems.

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35674-7_66

G. Goldszmidt et al. (eds.), *Integrated Network Management VIII*
© IFIP International Federation for Information Processing 2003

Among other factors, end-user training, helpdesk support and maintenance represent determinant contributions to heavy TCO costs [1-2]. Desktop management probably consumes the main share of human resources of the common IT department, easily outpacing server and network management altogether. Even considering that desktop management tasks are often less complex and specialized (and therefore less expensive) their cost is definitely not negligible.

There were several attempts to reduce these costs, mostly lead by the industry: commercial desktop management suites [3-4]; the DMTF initiative [5]; Thin Clients [6-7]; Microsoft's *Zero Administration for Windows* and Intel's *Wired for Management* [8]. However, despite these efforts, the current practice of desktop management is still not satisfactory.

Desktop management is usually based on some kind of *runtime agent* that supports remote operations such as monitoring, configuration and inventory management. This agent is then complemented with a centralized console used by the operator to access and manage several PCs.

One problem with this model is that only relatively healthy PCs can be managed. In the case of hardware, file system or operating system (OS) boot failure the agent will not load and local intervention will be required.

Another problem is related with the lack of integration. Management suites tend to be designed for specific environments (generally complete Windows PCs connected to local Windows Domain Controllers) and their functionality for alternative configurations is either reduced or non-existent at all.

Based on our own experience in the management of small PC networks with tenths or hundreds of PCs per location, we feel the current model might cover the basic needs in the management of groups of healthy and homogeneous Windows PCs. However, it provides no recovery solutions for problems prior to OS load and it lacks flexibility in the management of more heterogeneous environments, such as diskless PCs, Unix desktops, Windows Thin Clients and terminals based on Citrix Metaframe [9] or X-Windows.

In the `OpenDMS` project [10] we are exploring a different approach: the desktop becomes remotely manageable right after the initial Power On Self Test (POST) procedures; lighter PC configurations, such as Thin Clients, are explicitly supported and integrated in the management model; and mainstream standards and open tools are used to build an integrated network environment for the managed desktops.

The rest of this paper is organized as follows. The current status of desktop management is discussed in Section 2. The general architecture of the `OpenDMS` project is presented in Section 3. The next two Sections detail two of the most distinctive features of `OpenDMS`: PreOS management (Section 4) and the support for Thin Clients (Section 5). Section 6 presents a deployment scenario and Section 7 concludes the paper.

2. DESKTOP MANAGEMENT

Unlike other computer systems, for traditional PCs – "the most crash-prone computers ever built" [11] – reliability is less important than cost, performance or functionality. This approach opened the way for the *PC in everyone's desk* era and accelerated the spread of paradigms like client/server, distributed and workgroup computing. However, it is also truth that replacing mainframe-based computing by

the current decentralized architecture, where the PC is the dominant specie, failed to fully deliver some of its promises, such as a dramatic reduction of TCO costs [1-2].

Over the years we have witnessed several industry-lead attempts to physically redesign PCs, to standardize PC management services and even to replace the classical full-blown PC desktop by new desktop paradigms, such as Thin Clients and Network PCs.

TCO and ecological concerns motivated several attempts to physically redesign the PC, with initiatives like the IBM PS/2 Energy Desktop [12]; Intel's Micro ATX and Flex ATX [13-16], and VIA Technologies' ITX and Mini-ITX [17-19] reference design prototypes. All these designs, as well as the PC Design Guides [20], from Intel and Microsoft, share the same core ideas: simpler PCs with reduced power consumption, enhanced ergonomics, flexibility, legacy-free design, small size and aesthetics.

Thin Clients and Network PCs were also proposed as a solution for the TCO problem. Reducing user freedom and moving data and programs back to a central system seemed the logical solution, but the paradigm failed to reach critical mass. In our opinion this was probably due to bandwidth limitations, the need to rewrite a large number of applications and the "proprietary/open-disguised" vendor approach in which software and hardware were actually closed and very platform-specific. This happened to projects like the X-Terminal, Microsoft's NetPC [6] and Oracle's NC [7]. Ironically, now that the hype has vanished, increasing network bandwidth and emerging computing technologies are finally creating the conditions for successful widespread deployment of truly open Thin Clients.

The Distributed Management Task Force (DMTF) [5] took the leading role in determinant initiatives for desktop management standardization, such as the desktop management interface (DMI [21]), the system management BIOS (SMBIOS [22]) and the Common Information Model (CIM [23]). DMTF standards provided ground support for initiatives like Microsoft's Zero Administration for Windows and Intel's Wired for Management, which tried to create a set of tools and technology resources to help systems administrators [8].

As already mentioned, desktop management products are based on OS-dependent management agents that directly support a predefined set of management operations, as well as direct remote desktop access for more unusual operations. Some of these tools also provide some level of integration with Windows management mechanisms, allowing integrated software distribution and configuration management for Windows domains. However, interoperability between products of different vendors is still reduced. Solutions like Intel's Landesk suite [4] and Microsoft's SMS [3] are effective only when managing Microsoft-only networks and following very product-specific management guidelines that difficult interoperability.

3. THE OPENDMS APPROACH

`OpenDMS` is a project where we try to complement the traditional desktop management model, addressing issues not covered by current practices. Two of the most relevant issues are the remote desktop management in the PreOS stage and the integrated support of alternative desktop models, such as thin client configurations and UNIX workstations.

To better understand how those issues interact with classical management technologies, the project also comprises a complete management framework (built using open standards and already available tools) that includes, for instance, a *runtime management agent* with many similarities with typical desktop management agents. However, it should be stressed that `OpenDMS` is not competing with tools like SMS or Landesk. In fact, in some situations it is possible and worthwhile to complement the `OpenDMS` framework with more sophisticated management agents from commercial suites.

3.1 PreOS Management and Platform Neutrality

The `OpenDMS` target platform is the common PC built with of-the-shelf hardware components. With very few exceptions (e.g. Apple machines or computers with very exotic hardware) this practically covers every PC sold nowadays.

Minimizing OS-dependency was also possible because many procedures are performed in an OS absent state: the so-called PreOS instant of the PC initialization sequence (see Figure 1). This opposes to typical desktop management solutions whose pre-boot capabilities are non-existent or, at most, limited to rudimentary OS installation or image distribution.

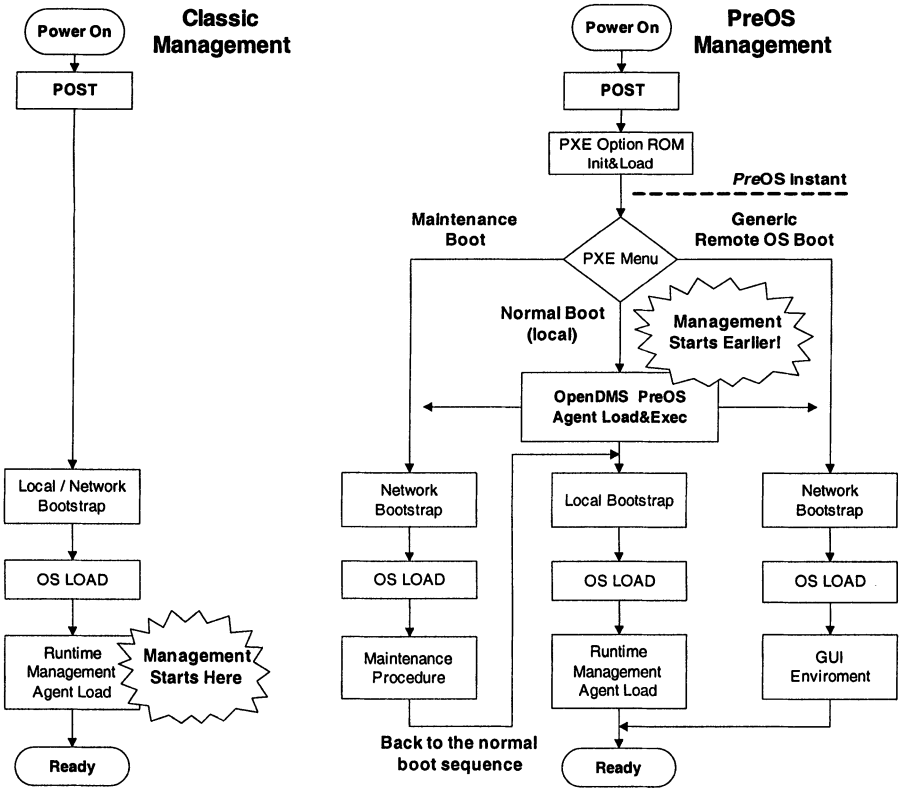

Figure 1. PreOS Management vs. Classic Desktop Management

With OpenDMS a PC becomes remotely manageable right after the initial POST procedures. Making a remotely controlled detour in this precise instant, in order to download and execute a special PreOS Agent or to boot an OS over the network, it is possible to manage a PC without a working OS. It is enough to have a network connection, a core set of operational hardware components (power supply, motherboard, memory, processor, network adapter) and standard remote boot firmware extensions.

OS-present operations are contemplated through the use of an additional management agent which is similar to those normally found in classical management suites, although somewhat different because several tasks normally executed at OS runtime are now performed by the PreOS Agent. OS load time and runtime load are considerable reduced, since the management agent imposes little load on the client system: there are no integrated web-servers or hardware asset management tools, with the only exception being the support for hot-pluggable devices.

The OS runtime component of OpenDMS was also designed to be as platform neutral as possible. In order to take advantage of specific OS features, the management agents are obviously OS-dependent. However, their management service interfaces follow a common specification. Furthermore, the whole environment – distributed file systems, remote terminal services, thin-client support services, etc. – is based on widespread standards and open tools available in a wide number of multi-platform versions, including at least the Windows family, FreeBSD and the most popular Linux variations.

3.2 Architecture

Figure 2 shows the OpenDMS client architecture. The first layer consists of PC hardware (of-the-shelf PC components) and related resources, such as Wake-On-LAN [24] (for remote off-hours maintenance tasks like backup and virus scanning) and system instrumentation.

The second layer consists of native firmware and standard extensions, including Preboot Execution Environment Boot ROM (PXE [25]), PC BIOS and DMI/SMBIOS. All these components are already available in current PCs, and their role in PreOS management will be discussed in Section 4.

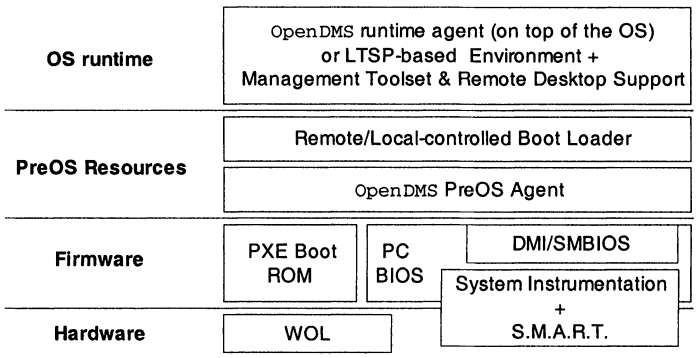

Figure 2. OpenDMS Architecture (Client Side)

The PreOS Agent is the key component of the third layer. This agent is dynamically downloaded from the OpenDMS server and performs several management tasks in the absence of an operating system. After that, the PreOS agent passes control to the standard Boot Loader, which goes on with the boot of the operating system (either from the network or from local storage devices).

After OS load there are two possible configurations. If we have a traditional operating system, remote management is assisted by a classic runtime agent controlled by the management server. On the other way, it is also possible to build a thin-client configuration using additional resources provided by the OpenDMS framework. These resources are organized in the form of a modular thin-client platform specification built from commodity PC hardware, with support for network file systems and multiplatform connection capabilities that include Microsoft RDP-based systems [26], X/Windows servers, Citrix Metaframe [9] (client not available under GPL) and character-based protocols. Supporting access to local multimedia and storage, if needed, this platform is flexible enough to perform adequately in many usage scenarios, from desktop hybrid NCs-PCs to multimedia kiosks. Thin Client support will be discussed in Section 5.

On the server side one there are two distinct servers (Figure 3). The first is the core desktop management server. This server – built on top of GNU/Linux – uses PXE and either the classic TFTP (Trivial File Transfer Protocol) or its multicast-enhanced version (MTFTP) to upload the PreOS agent to the managed desktop. It remotely controls the execution of this PreOS agent (e.g. validating user authentication), as well as the runtime management agent. The systems manager uses a Web-based user interface and processed management information is organized using OpenLDAP [27].

The second OpenDMS server is dedicated to thin-client support. This server exports to the managed desktop a Linux-based thin-client environment built on top of distributed file systems (NFS [28], Samba/SMB [29] and NBD [30]), eventually existing local file systems and remote desktop clients (X-Windows, character-based protocols, Windows Terminal Server/RDP [26], VNC [31] and Citrix Metaframe). PXE and TFTP (or MTFTP) provide support to remotely boot an OS image. This set of services is either provided by a single machine or spread across several servers. In the specific case of Windows Terminal Server and Citrix Metaframe the service has to be provided by a native Windows server. To configure and maintain this second OpenDMS server the systems manager also uses OpenLDAP and a Web interface.

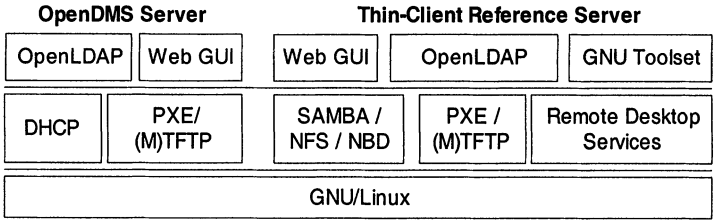

Figure 3. OpenDMS Architecture (Server Side)

4. BUILDING PREOS MANAGEMENT

The PreOS stage of the PC initialization process still remains a sensitive area for the great majority of available desktop management tools, probably because until a few years ago there was no built-in firmware support for PreOS management mechanisms. However, modern PCs support a specific kind of special-purpose firmware extensions through the use of option ROMs, normally embedded in hardware such as Network Boot ROMs. Those extensions were originally designed just for remote deployment of operating systems over the network, but there is no reason not to use them to force the download of more specific software, such as a PreOS Management Agent.

Ordinary Boot ROMs are incapable of dealing with the needs of PreOS management due to design limitations. Standard TCP/IP Boot ROMs are built around unicast-based protocols with remote OS load from a single point in mind. Without any kind of load-balancing or fault-tolerant features, they are too unreliable to use as a PreOS management tool. Furthermore, implementations from different sources tend to use non-standard downloadable OS image formats, leading to incompatibilities. These issues were solved by PXE Boot ROMs, designed with interoperability and reliability concerns in mind. They extend the basic remote boot functionality, providing a basic OS-absent program execution environment.

PXE Boot ROMs were created in the context of the Intel Boot Initiative [32], circa 1998, and meanwhile they made its way into almost every kind of PC network interface currently produced. At the time of this writing Intel is working in the specification of an Extensible Firmware Interface (EFI), as a replacement for the currently available BIOS architecture for IA-32 systems (already available for IA 64 systems), that will include a PXE-compliant facility [33]. PXE is the standard for remote boot ROMs.

Figure 4 presents the modular structure of the PXE specification. Integrated in the system's firmware as an option ROM, the PXE BIOS extension provides four APIs designed to be used by any kind of Network Boot Program:

- PreBoot API. This API provides the means to control the entire PXE environment, from deactivation/activation of the embedded TCP/IP stack to access to DHCP packet data received during the boot negotiation process.
- (M)TFTP API. This API supports TFTP or MTFTP-based file transfer functions.
- UDP API. This API provides basic UDP-supported I/O functions.
- UNDI (Universal Network Device Interface). This API is an abstraction layer that works as a universal device driver, allowing the other APIs to work independently of the available network interface hardware. This layer also allows the creation of universal device drivers partially supported by the PXE UNDI API.

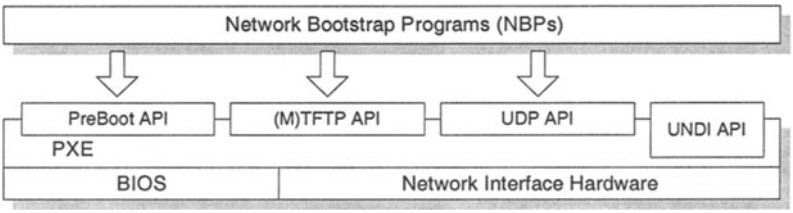

Figure 4. The PXE Boot ROM

It should also be stressed that PXE provides fault-tolerance and load-balancing mechanisms: there are several boot servers, allowing the client to boot from one of them according to several load-balancing techniques.

Once the system is turned on and the POST routines are executed, the system's BIOS begins searching and initializing all option ROMs. The PXE code will then force the download and execution of the PreOS Agent.

Figure 5 shows the key execution steps of the PreOS Agent. Once initiated, it tries to establish communication with the management server in order to receive enough information to proceed with the remaining tasks. Next, if the server informs the agent that the workstation is allowed to boot, it will make a system integrity check and will report the status together with the hardware configuration detected (using DMI/SMBIOS or direct access methods). If the system status is satisfactory and the hardware inventory list is coherent with what the server has in its own database, the agent will be authorized to proceed to the next step. Otherwise the system may lock the boot process (avoiding further progress and shutting down the system, if needed). After an optional user authentication step, the PreOS Agent will pass control to the bootstrap loader, allowing the system to proceed with its normal boot sequence. Since the specific OS image to be loaded may be selected by the `OpenDMS` server, it becomes possible to have a PC with multiple personalities (Windows desktop, Unix workstation, thin-client, etc.) according to the user and the circumstances. This feature can also be used to load an OS-environment specifically designed for recovery operations.

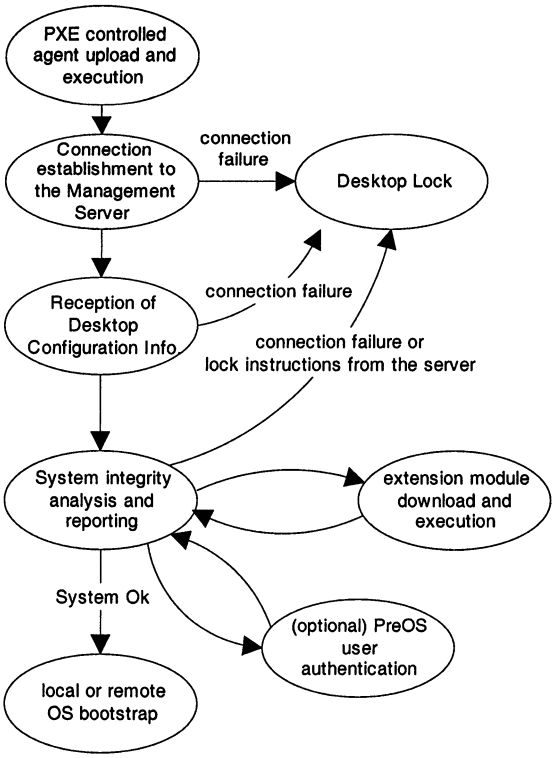

Figure 5. PreOS Agent Execution

Thanks to the PreOS Agent, common tasks like user authentication and hardware inventory can be performed in the PreOS stage of the PC boot sequence, without depending on a runtime OS environment. Furthermore, the PreOS Agent is developed around a modular approach in which functionality can be extended by adding third-part downloadable modules. Two of the most interesting modules already developed are *memtest86* and *S.M.A.R.T.* (System Monitoring, Analysis and Report Technology), which provide extensive memory and hard drive diagnostic features accessible to the PreOS management toolset.

If the normal boot sequence is followed and an OS is loaded, the `OpenDMS` runtime agent (or eventually a more feature-rich runtime agent from a commercial third-party provider) will ensure that the system is still remotely manageable during normal operation.

5. THE OPENDMS THIN CLIENT APPROACH

As already mentioned, Thin Clients and Network PCs were once seen as an interesting paradigm to reduce the complexity of the user desktop and move data and programs back to a central system. However, the proprietary approach followed by products like the X-Terminals, Microsoft's NetPC and Oracle's NC, as well as technological limitations, prevented the paradigm from reaching critical mass.

But nowadays such weaknesses are addressable using a different approach: why not build a Thin Client based on common PC-hardware with the same capabilities of its proprietary counterparts? Bandwidth is now available and current PC hardware is far more sophisticated than the one found in most proprietary solutions. PXE also fits these purposes: despite being used mostly for OS deployment (booting a small mini-environment that performs OS-deployment procedures), PXE supports the direct boot of an OS through the network. Therefore, its role is not limited to the execution of the PreOS Agent (which is also important in the selection of the correct "desktop personality"), since it is possible to use PXE and TFTP to load a remotely-provided Thin Client environment.

`OpenDMS` includes several modules – most of them adapted from the LTSP Project [34] – that can be combined to build custom-tailored Thin Client configurations. Four distinct operations modes were considered:

- Diskless GUI-based Thin Clients, in which the system does not need any kind of local fixed mass-storage device, booting from a cluster server, accessing a remote file system using NFS, and using a small amount of local memory as a *ramdrive* to cache frequently used data. It uses a locally executed X-server to provide display facilities to applications ran on the remote cluster server or to provide a graphic environment in which an RDP or Citrix client can be used to access a Windows Terminal Server system. Access to local audio, serial, parallel and removable storage is possible on such configurations, thanks to the use of smart redirector methods (as the network block device protocol). This mode can also be use to build intelligent, remotely managed multimedia kiosks or point-of-information terminals. Normal PCs may also benefit from this mode of operation, allowing them to behave like they had an alternate Thin Client personality, depending on a user-selectable boot option in a PXE-provided menu. Additionally, this operation mode provides a special remote helpdesk

mode in which a normal PC can boot directly into a browser window pointing to a helpdesk request service page.
- Network Computer, a natural extension of the Thin Client mode in which the system is allowed to possess local fixed mass-storage facilities used to store frequently used programs and configuration data.
- Network Appliance. In this mode, a PC boots from the network with a special purpose, self-contained mini-Linux distribution in order to transform a simple PC in a network appliance, such as a print server with no local storage facilities, a remotely manageable Network Attached Storage system or a basic Router/Firewall.
- Recovery Mode, a special operation mode devised to allow the remotely initiated recovery of a PC with a seriously damaged file system. A normal PC can boot into this mode in order to download and restore a previously-uploaded file system image from a cluster server. This procedure depends on a set of GPL licensed tools (like the parted partition manipulation tool) executed under the control of automated scripts on a minimal self-contained Linux environment booted from the network.

Thin-clients do not require top-notch hardware. Even a fifth generation *x86* system (e.g. systems based on the old 90 MHz Intel Pentium, with a PCI bus, a PXE-capable network interface card and 32MByte of RAM) is able to provide a reasonable hardware platform to build an `OpenDMS` Thin Client. This way hardware lifetime is significantly extended.

6. DEPLOYMENT SCENARIO

Figure 6 shows an example of an `OpenDMS` managed environment. In this environment there are Linux Workstations and Windows Desktops with a "normal" configuration based on a local OS. There are PCs that have multiple personality, dynamically selected according to the circumstances. There are also diskless PCs, used as thin clients, and one managed network appliance (a small router/firewall). There is one `OpenDMS` server dedicated to the management tasks and another server for Thin Client support.

For PCs with local OS and fixed configuration (Windows and Linux desktops) the PreOS Agent is primarily used to diagnostic hardware or file system failures. In the case of severe hardware failure the system may be locked waiting for local intervention. In the case of file system or OS failure the system might boot a special recovery environment. This special mode is a lightweight Linux installation, provided by the "Thin Client Server", with recovery tools from the GNU Toolset and means to allow the user to perform helpdesk request operations from the damaged PC itself, even in the case of hard drive failure or corruption. It is also possible to upload and locally install a previously prepared OS image of the system stored in the "OS Images Database" (eventually overwriting the corrupted OS). After successful OS load, the runtime management agents provide the means to perform remote management on these desktops.

Users of multiple-personality desktops are authenticated by the PreOS Agent. Based on the user ID and the adopted policies, the `OpenDMS` server enforces one of several available configurations: a complete Windows/Linux machine booting from a local OS; a Network PC booting from the network but still using local storage

devices; or a Thin Client working like a diskless system in GUI mode and allowing access to the Windows Terminal Server or a more generic Application Server. Remotely loaded OS images are provided by the Thin Client Server, and runtime network file systems might be provided by an additional Network Attached Storage device using SAMBA or NFS.

Diskless PCs are a simplification of the multiple-personality desktops: although it is also possible to choose from several available OS images, only diskless configurations are possible (e.g. web browsers, X terminals or Windows diskless terminals). Nevertheless, using NBD [30], it is still possible to use local devices such as CD drives, microphones or speakers.

The network appliance loads a task-specific remote OS image each time it is booted. There are no local file systems to maintain, and router hardware failures are easily solved: get a new computer (or replace the faulty hardware), change the appliance ID in the `OpenDMS` servers and turn on the new computer. It will automatically load the router software and start working.

The `OpenDMS` server includes the DHCP server – to manage the IP pool reserved to the client systems and to provide PXE discovery services – the PXE server and a TFTP service to upload the PreOS Agent. This server controls both the PreOS Agent and the runtime management agents.

The Thin Client Server is used just to maintain OS images that may correspond to "local OS images" (i.e. uploaded to the desktop and locally booted) or operating systems directly booted from the network: Thin Client configurations, special recovery environments and network appliances designed for specific tasks.

Network file systems and remote desktop services are conceptually part of the Thin Client Server (see Figure 3). However, in the presented scenario they are provided by three distinct specialized servers: a file server using Samba and/or NFS, a generic application server (based, for instance, on X-Windows or Web interfaces) and a Windows Terminal Server or a Citrix Metaframe Server to run Windows based applications (e.g. Microsoft Office) from thin clients.

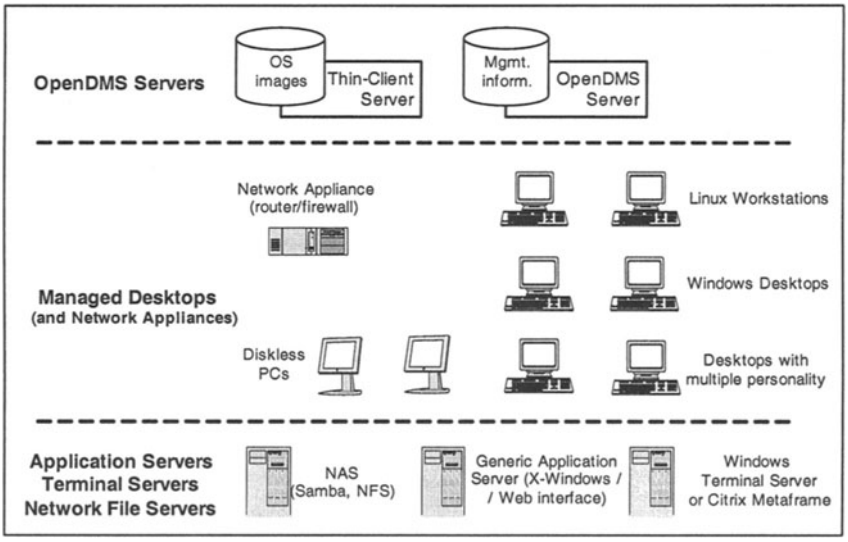

Figure 6. Example of an *OpenDMS* Managed Environment

There are several relevant issues not explicitly exposed by the presented scenario: the total amount of servers required to build an operational environment; the potential usage of load-balancing mechanisms; and integration with commercial desktop management suites.

Figure 6 presents two `OpenDMS` servers and three additional servers providing network file systems and remote desktop services. Five servers might be overkill for small networks and not enough for larger infrastructures. Our experience shows that it is possible to concentrate the support of 10 to 20 thin-clients in one desktop management server and one thin-client server. Even using just these two servers (two basic Celeron 400 MHz/256 MByte machines, in our specific testbed) it is possible to provide a basic degree of resilience and load-balancing, since several services (e.g. NFS, PXE, TFTP, MTFTP and recovery support) are easily and naturally replicable in both machines. In larger systems this natural resilience might be combined with the distribution of services across specialized servers located in strategic network locations, increasing the number of servers to manage but achieving a higher level of performance, availability and scalability.

So far we have not invested much effort in the potential integration with already existing desktop management platforms. Nevertheless, PreOS management and specific support for thin-clients are orthogonal to current approaches, and therefore it is possible to use `OpenDMS` and other platforms at the same time without overlapping each other (except for the runtime management agents). High-level integration could eventually be achieved using the LDAP-organized management information and policies kept by the `OpenDMS` server. Tighter integration would require adapting current platforms in order to control the PreOS Agent, whose interface is simple and well documented. Extending this agent is also simple, since it comprises an interface for additional software modules.

7. CONCLUSIONS AND FUTURE WORK

PC ubiquity, as well as PC increasing complexity, keeps desktop management as one of the most resource-consuming areas of operations and support. Despite several industry-lead initiatives the truth is that classical desktop tools target just a fraction of the whole problem.

In the `OpenDMS` framework we are searching for alternative solutions for this problem. Rather than directly competing with the best features of commercial products – such as their suitability for integrated management of healthy Windows domains – we are more interested in complementing these tools with less common approaches.

One of these approaches is PreOS management. PreOS management allows more complete control of user-available computing resources. With PreOS management it becomes possible, for instance, to perform early hardware checkup, user-authentication procedures and basic file system verification. Furthermore, PreOS makes possible to flexibly select the PC personality (full-blown PC, thin client, network PC, network appliance, etc.) according to the identity of the user and the status of the system. Relatively recent industry standards, such as PXE, DMI BIOS and SMBIOS finally made possible the development of capable PreOS management agents. However, to the best of our knowledge, `OpenDMS` is the first tool that provides such functionality.

The thin client approach also differentiates OpenDMS management. This approach corresponds to the notion that thin clients are indeed the right answer to many situations, with considerable advantages over traditional PCs: increased control, increased reliability and less demanding hardware requirements. Although this might sound as *déjà vu*, considering the vanishing hype around previous thin client proposals, our approach to the subject does bring some new elements. First, we believe that some of the problems that affected previous approaches are now less constraining than before. LAN bandwidth is becoming cheaper and cheaper, for many uses a modern Web Browser in sufficient and even remote desktop technology has been remarkable improved in the last few years. Second, we are not going after a proprietary or "100% thin client" approach. Instead, we are using regular PC hardware and open source software to build flexible thin clients that may fit many configurations, from diskless computers to thin clients or even network appliances. In fact, it becomes even possible to select the "desktop personality" at boot time, according to the intended use.

OpenDMS is not limited to these two aspects, and its architecture also embraces more classical management mechanisms, such as runtime management agents. However, at least for now, we just use those runtime management mechanisms to better understand the potential implications of PreOS management in classic management practices. Implemented runtime agents (for Windows and Linux) just provide remote desktop access (built upon VNC), basic health monitoring, software inventory, troubleshooting and log facilities. Systems administrators needing additional features can use OpenDMS just for PreOS management and thin client deployment, while keeping commercial products for runtime management of Windows PCs.

OpenDMS already includes fully functional implementations of the PreOS Agent and the reference Thin Client environment. The *proof-of-concept* server is also developed but still needs to be productized (e.g. with simpler installation procedures and user friendlier interfaces). We are currently testing the platform using a few dozens of classroom PCs, in order to study scalability, robustness, and hardware and bandwidth requirements. Further work on security is also planned: complying with the PXE specification, the current implementation uses potentially dangerous protocols such as TFTP, requiring either the usage of more careful network configurations or the evolution for more secure network services.

REFERENCES

[1] Gartner Consulting Group, "TCO Analyst: A White Paper on GartnerGroup's Next Generation Total Cost of Ownership Methodology", 1997, pp. 19-20

[2] S. Heilbronner, R. Wies, "Managing PC Networks", IEEE Communications Magazine, October 1997

[3] Microsoft Systems Management Server (SMS) Homepage, http://www.microsoft.com/smserver/

[4] Intel Landesk Homepage, http://www.intel.com/network/products/landesk/

[5] DMTF, Distributed Management Task Force Homepage, http://www.dmtf.org

[6] Microsoft Corporation, "Network PC System Design Guidelines v1.0b", 1997 (available at http://www.eu.microsoft.com/hwdev/archive/netpc.asp)

[7] Willian Blundon, "Deciphering the NC World", JavaWorld, March 1997 (available at http://www.javaworld.com/javaworld/jw-03-1997/jw-03-blundon.html)
[8] L. Weller, "Reducing TCO with Intel WFM and Microsoft ZAW initiatives", Intel Developer Update Magazine Issue 17, February 1999
[9] Citrix, Citrix Metaframe Homepage, http://www.citrix.com/
[10] Tiago Cruz, Paulo Simões, "Rethinking Desktop Management", Proceedings of APNOMS'2002, Jeju, South Korea, September 2002
[11] Tom Halfhill, "Here's why today's PC's are the most crash-prone computers ever built – and how you can make yours more reliable", Byte Magazine, April 1998
[12] IBM Corporation, "IBM PS/2 E product information", IBM DC 00210-00, 1993
[13] N. Sumrall, "High Concept Comes to the PC: Form, Function and Fashion", Report from Intel Developer Forum, fall'99, Intel Developer Update Newsletter
[14] Intel Corporation, "MicroATX Motherboard Interface Specification", 1997
[15] Intel Corporation, "FlexATX Addendum V 1.0 to the microATX Specification", 1999
[16] Intel Corporation, "Ease of Use Initiative – Concept PC", 2002
[17] VIA Technologies, "ITX Mainboard Specification White Paper", 2001
[18] VIA Technologies, "Mini-ITX Mainboard Specification White Paper", 2002
[19] VIA Technologies, "Information PC reference design", 2001
[20] Intel Corp, Microsoft Corp, "PC Design Guide Website", http://www.pcdesguide.org/
[21] DMTF, "Desktop Management Interface (DMI) v2.0s", 1998
[22] DMTF, "System Management BIOS Reference Specification v2.3.2", 2001
[23] DMTF, "Common Information Model (CIM) Specification v2.2", 1999
[24] AMD Corp., "Magic Packet Technical White Paper", 1995
[25] Intel Corp, "Preboot Execution Environment (PXE) specification version 2.0", 2000
[26] Mark Chapman, "Rdesktop project", http://rdesktop.sourceforge.net
[27] OpenLDAP, http://www.openldap.org
[28] Hal Stern, "Managing NFS and NIS", O'Reilly & Associates, 1992
[29] Samba Project, http://www.samba.org
[30] NBD Project, http://nbd.sourceforge.net
[31] AT&T Cambridge Labs, "Virtual Network Computing", http://www.uk.research.att.com/vnc/
[32] Intel Corp, "Intel Boot Initiative program", 1998
[33] Mike Henry, "PXE Manageability Technology for EFI", Intel Developer Update Magazine, October 2000
[34] LTSP, "Linux Terminal Server Project", http://www.ltsp.org