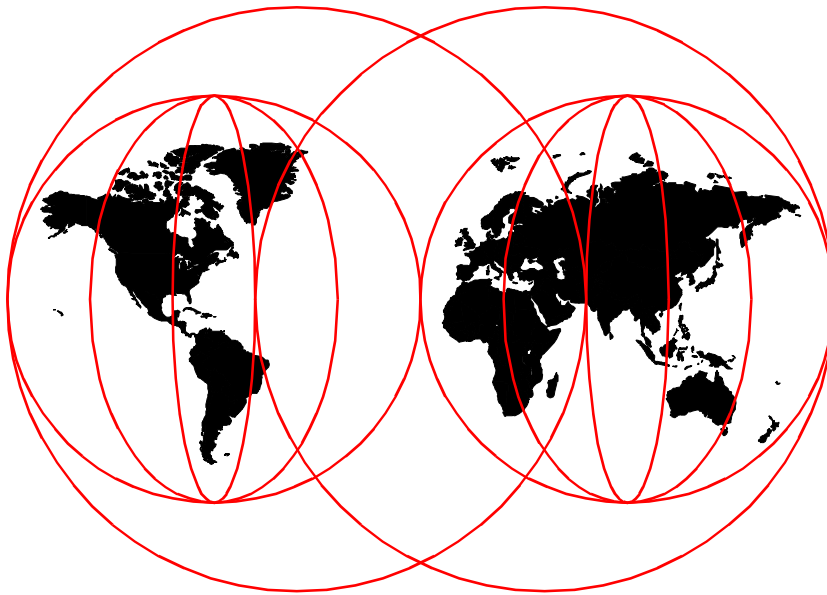


# **WorkSpace On-Demand 2.0 Feature for Windows Clients**

*Axel Buecker, Joao Miguel Galvao, Khathutshela Muvenda, Indran Naick*



**International Technical Support Organization**

[www.redbooks.ibm.com](http://www.redbooks.ibm.com)

SG24-5396-00





International Technical Support Organization

**WorkSpace On-Demand 2.0 Feature  
for Windows Clients**

August 1999

**Take Note!**

Before using this information and the product it supports, be sure to read the general information in Appendix D, "Special notices" on page 235.

**First Edition (August 1999)**

This edition applies to IBM WorkSpace On-Demand 2.0 Feature for Windows Clients for use with the IBM WorkSpace On-Demand 2.0 and OS/2 Warp Server.

Comments may be addressed to:  
IBM Corporation, International Technical Support Organization  
Dept. DHHB Building 003 Internal Zip 2834  
11400 Burnet Road  
Austin, Texas 78758-3493

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

**© Copyright International Business Machines Corporation 1999. All rights reserved.**

Note to U.S Government Users – Documentation related to restricted rights – Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

---

## Contents

<b>Figures</b> .....	ix
<b>Tables</b> .....	xi
<b>Preface</b> .....	xiii
The team that wrote this redbook .....	xiii
Comments welcome .....	xvii
<b>Chapter 1. Introduction</b> .....	1
1.1 WorkSpace On-Demand 2.0 overview .....	1
1.2 The Feature for Windows Clients .....	3
1.2.1 Components .....	4
1.2.2 Server platforms supported .....	7
1.2.3 Client platforms supported .....	7
1.3 Client hardware considerations .....	8
1.3.1 Remote boot capability .....	8
1.3.2 Windows 95 and Windows 98 hardware .....	9
1.3.3 Windows NT 4.0 hardware .....	9
1.3.4 Device support .....	10
1.4 Licensing and packaging .....	10
<b>Chapter 2. Feature for Windows Clients architecture</b> .....	13
2.1 Overview .....	13
2.1.1 Client-side architecture .....	14
2.1.2 Server-side changes .....	16
2.2 The boot process .....	18
2.3 The installation boot sequence .....	19
2.3.1 Operational boot sequence .....	24
2.4 RPL control files .....	26
2.4.1 RPL.MAP .....	26
2.4.2 The STATE files .....	39
2.4.3 A sandbox .....	42
2.4.4 Additional boot process options .....	43
2.5 The user environment .....	43
2.5.1 Building the desktop .....	44
2.5.2 The Feature for Windows Clients Networks Logon Client .....	50
2.5.3 The logon process .....	51
2.5.4 User home directory usage .....	53
2.5.5 User management .....	53

<b>Chapter 3. Installation and planning</b> . . . . .	55
3.1 Install environment . . . . .	55
3.1.1 Single-server environments . . . . .	55
3.1.2 Multiple-server environments . . . . .	56
3.1.3 Windows NT integration . . . . .	57
3.1.4 File systems . . . . .	57
3.2 Hardware prerequisites . . . . .	58
3.2.1 Processor . . . . .	58
3.2.2 Disk space . . . . .	58
3.2.3 Memory . . . . .	59
3.3 Software prerequisites . . . . .	59
3.4 Installation roadmap . . . . .	61
3.5 Install and client operating system setup . . . . .	62
3.5.1 Attended installation . . . . .	62
3.5.2 Lightly attended installation (CID) . . . . .	67
3.5.3 Unattended installation . . . . .	69
3.5.4 Reinstalling the Feature for Windows Clients . . . . .	69
3.5.5 Post-installation procedures . . . . .	71
3.5.6 Uninstalling the Feature for Windows Clients . . . . .	72
3.5.7 Adding the client operating system . . . . .	72
3.6 Year 2000 readiness . . . . .	73
3.6.1 OS/2 Warp Server and WorkSpace On-Demand 2.0 . . . . .	73
3.6.2 Windows client operating system . . . . .	73
3.7 Initial testing . . . . .	76
3.8 Tivoli Management Agent (TMA) and Java Virtual Machine (JVM) . . . . .	78
3.8.1 Installing the TMA . . . . .	78
3.8.2 To exclude the Java Virtual Machine (JVM) . . . . .	79
3.9 Client performance . . . . .	80
3.9.1 File system considerations . . . . .	82
<b>Chapter 4. About application creation and management</b> . . . . .	83
4.1 Application processing . . . . .	83
4.1.1 Overview: Capturing and creating an application . . . . .	85
4.1.2 Overview: Distributing applications . . . . .	86
4.2 Application type . . . . .	88
4.2.1 Well-behaved applications . . . . .	88
4.2.2 Server-based applications . . . . .	89
4.2.3 Client-based applications . . . . .	89
4.2.4 Selecting applications . . . . .	89
4.3 Capturing and creating an application package . . . . .	90
4.3.1 Define and set up the sandbox machine . . . . .	91
4.3.2 Define a sandbox user account . . . . .	92
4.3.3 Create application directories . . . . .	93

4.3.4	Create the application package . . . . .	95
4.3.5	Define the application package . . . . .	103
4.4	Distributing the application files to client machines . . . . .	104
4.4.1	Using commercial software distribution products . . . . .	104
4.4.2	Distributing files within the client image . . . . .	104
4.4.3	Updating the Windows NT client image . . . . .	106
4.4.4	Updating the Windows 9X client image . . . . .	110
4.4.5	Renaming files that are in use on Windows NT . . . . .	114
4.4.6	Renaming files that are in use on Windows 9x . . . . .	115
4.5	Assigning and deleting applications for users . . . . .	115
4.5.1	Assigning an application to a user . . . . .	115
4.5.2	Deleting an application from a user . . . . .	117
4.5.3	Enumerating an application for a user . . . . .	119
4.6	Application management commands . . . . .	120
4.6.1	Adding an application package . . . . .	121
4.6.2	Deleting an application package . . . . .	122
4.6.3	Enumerating an application package . . . . .	122
4.6.4	Querying an application package . . . . .	123
4.6.5	Modifying an application package . . . . .	123
4.7	Application commands . . . . .	124
4.7.1	The CRTPKG command . . . . .	124
4.7.2	The NETWIN APP command . . . . .	126
4.8	Application control files and data stores . . . . .	128
<b>Chapter 5. Workstation customization and administration . . . . .</b>		<b>133</b>
5.1	About defining workstations . . . . .	133
5.1.1	Defining workstations individually . . . . .	138
5.1.2	Defining multiple workstations . . . . .	141
5.2	The install scripts . . . . .	142
5.2.1	Windows NT Workstation install files . . . . .	143
5.2.2	The \$\$Rename.txt files . . . . .	149
5.2.3	Windows 9x installation flow . . . . .	150
5.2.4	Operating system files on the client . . . . .	155
5.3	Summary chart . . . . .	156
5.4	Machine classes and response files . . . . .	157
5.4.1	IBM PC 300GL (Model 6561-54U) . . . . .	157
5.4.2	IBM PC: 350, 730, and 750 . . . . .	157
5.4.3	Creating your own machine classes . . . . .	158
5.5	Video adapters . . . . .	159
5.5.1	Installing support for video adapters on Windows NT . . . . .	159
5.5.2	Installing support for video adapters on Windows 9x . . . . .	160
5.6	Enabling network adapter support . . . . .	161
5.6.1	Enabling network adapters on Windows NT . . . . .	161

5.6.2	Enabling network adapters on Windows 9x . . . . .	162
5.6.3	IBM Auto, Turbo, and 3COM . . . . .	167
5.7	Printing . . . . .	167
5.7.1	Windows 9x printing . . . . .	168
5.7.2	Windows NT printing . . . . .	169
5.8	Network protocol support . . . . .	171
5.9	The Microsoft ScriptIt utility . . . . .	172
5.9.1	Using ScriptIt . . . . .	173
5.9.2	Multimedia support using ScriptIt . . . . .	175
5.10	Other client commands . . . . .	176
5.10.1	Changing the client operating system . . . . .	176
5.10.2	Resetting a client . . . . .	177
5.10.3	Redefining a client . . . . .	177
5.10.4	Deleting a client . . . . .	177
5.10.5	Querying client information . . . . .	178
<b>Chapter 6.</b>	<b>User and desktop administration . . . . .</b>	<b>179</b>
6.1	Windows Clients . . . . .	179
6.1.1	The NETWIN USER command . . . . .	179
6.2	User profiles and system policy files review . . . . .	181
6.3	Customizing the desktop or user profile . . . . .	181
6.3.1	Planning your desktop . . . . .	182
6.3.2	Defining a new desktop . . . . .	183
6.3.3	Changing the appearance of an existing desktop . . . . .	190
6.3.4	Deleting the Connect to the Internet icon . . . . .	191
6.4	System policies . . . . .	193
6.4.1	The System Policy Editor . . . . .	195
6.4.2	System policy file considerations . . . . .	197
6.4.3	System policy templates . . . . .	197
6.4.4	Configuring policy settings . . . . .	198
6.4.5	WorkSpace On-Demand .adm files . . . . .	200
6.4.6	Changing the Windows NT system policy file . . . . .	202
6.4.7	Changing the Windows 95 and Windows 98 system policy file . . . . .	203
6.4.8	Hiding the desktop . . . . .	204
6.4.9	The Start menu . . . . .	205
<b>Chapter 7.</b>	<b>The Tivoli TMA client . . . . .</b>	<b>207</b>
7.1	About Tivoli as a company . . . . .	207
7.2	The Tivoli Management Framework . . . . .	208
7.3	Introducing the TMA . . . . .	208
7.3.1	The endpoint (TMA) . . . . .	209
7.3.2	The endpoint gateway . . . . .	209
7.3.3	The endpoint manager . . . . .	209



7.4	The role of the TMA . . . . .	210
7.5	TMA installation prerequisites . . . . .	210
7.5.1	Enabling the TMA . . . . .	211
7.5.2	Installing the TMA . . . . .	213
7.5.3	Configuration of the clients . . . . .	215
7.5.4	Stopping and starting the TMA . . . . .	215
<b>Appendix A. Comparing the Feature for Windows Clients . . . . .</b>		<b>217</b>
A.1	Introducing the alternatives . . . . .	217
A.1.1	Microsoft Windows Terminal Server . . . . .	217
A.1.2	Citrix MetaFrame . . . . .	219
A.1.3	The MetaFrame architecture . . . . .	221
A.2	Computing architectures, a quick comparison . . . . .	223
A.2.1	Why bother? . . . . .	224
A.3	The total cost of ownership . . . . .	224
A.3.1	Centralize as many resources as possible . . . . .	224
A.3.2	Reduce end user complexity . . . . .	226
A.3.3	Exploit current resources . . . . .	226
A.4	Application architecture models . . . . .	226
A.5	Summary . . . . .	227
<b>Appendix B. Supported network adapter list . . . . .</b>		<b>229</b>
B.1	Windows NT Workstation supported adapter list . . . . .	229
B.2	Windows 95 supported adapter list . . . . .	230
B.3	Windows 98 supported adapter list . . . . .	231
<b>Appendix C. IBM Object REXX interpreter . . . . .</b>		<b>233</b>
C.1	Features . . . . .	233
C.1.1	Object-oriented programming . . . . .	233
C.1.2	An English-like language . . . . .	233
C.1.3	Fewer rules . . . . .	234
C.1.4	Interpreted, not compiled . . . . .	234
C.1.5	Built-in functions and methods . . . . .	234
C.1.6	Typeless variables . . . . .	234
C.1.7	String handling . . . . .	234
C.1.8	Clear error messages and powerful debugging . . . . .	234
C.2	Installation . . . . .	235
<b>Appendix D. Special notices . . . . .</b>		<b>237</b>
<b>Appendix E. Related publications . . . . .</b>		<b>241</b>
E.1	International Technical Support Organization publications . . . . .	241
E.2	Redbooks on CD-ROMs . . . . .	241
E.3	Other publications . . . . .	241

<b>How to get ITSO redbooks</b> .....	243
IBM Redbook Fax order form .....	244
<b>Glossary</b> .....	245
<b>List of abbreviations</b> .....	249
<b>Index</b> .....	251
<b>ITSO redbook evaluation</b> .....	253

---

## Figures

1. WorkSpace On-Demand 2.0 components . . . . .	2
2. Client licensing. . . . .	11
3. Feature for Windows Clients, architecture . . . . .	14
4. Install boot sequence . . . . .	19
5. The install process . . . . .	23
6. The operational boot process . . . . .	25
7. Sample RPL.MAP file . . . . .	26
8. Going from the different server records . . . . .	34
9. Standard IBM Token Ring, boot block definition file . . . . .	36
10. Boot from hard disk, boot block definition file . . . . .	36
11. User's desktop creation . . . . .	44
12. Default desktop locations. . . . .	46
13. System policy files location . . . . .	48
14. User application management structure . . . . .	54
15. Accessible services . . . . .	57
16. Introductory panel . . . . .	63
17. Feature for Windows Clients components . . . . .	64
18. Installation parameters. . . . .	65
19. Ready to install panel. . . . .	66
20. Installation complete panel . . . . .	67
21. The lab environment . . . . .	77
22. Server I/O and client requirements . . . . .	81
23. Application process overview. . . . .	84
24. Creating an application package . . . . .	86
25. Application distribution, components . . . . .	87
26. The lab environment . . . . .	90
27. Creating a sandbox user account . . . . .	93
28. Creating an application directory . . . . .	95
29. Capturing the application . . . . .	96
30. Directory tree for an application package . . . . .	101
31. File locations . . . . .	131
32. Installation and configuration process for Windows client setup. . . . .	143
33. Script file, audio.scr . . . . .	175
34. Updated logonclt.bat file adding support for the Crystal audio drivers. . .	176
35. File list for the Crystal drivers. . . . .	176
36. Windows 95, Control panel . . . . .	183
37. Windows 95, Password Properties . . . . .	184
38. Windows 95 Display Properties, Background tab . . . . .	185
39. Windows 95 Display Properties, Appearance tab . . . . .	186
40. Windows 95 Display Properties, Screen Saver tab . . . . .	186

41. Windows 95 Program menu, before customization . . . . .	187
42. File listing for the Start menu, Programs tab . . . . .	188
43. File listing for the Start menu, Programs tab, after modification . . . . .	188
44. Windows 95 Program menu, after customization . . . . .	189
45. Implementing the system policy . . . . .	194
46. System Policy Editor . . . . .	195
47. Policy template options . . . . .	198
48. Editing the policy file . . . . .	199
49. Setting up a custom program folder. . . . .	206
50. Steps to enable the TMA . . . . .	211
51. The endpoint login process . . . . .	212
52. Data flow in the Terminal Server . . . . .	219
53. ICA protocol bandwidth . . . . .	220
54. Citrix MetaFrame architecture . . . . .	222

---

## Tables

1. The RPL.MAP file - Workstation record field descriptions . . . . .	28
2. The RPL.MAP file - Server record field descriptions . . . . .	30
3. The boot block definition file - Field descriptions . . . . .	37
4. Values for the State variable in STATE.FIL . . . . .	40
5. Fields in the STATE file . . . . .	40
6. Disk space requirements for the Feature for Windows Clients . . . . .	59
7. Disk space requirements for Window Clients Feature clients . . . . .	59
8. Feature for Windows Clients, installation steps . . . . .	61
9. Utilities to install Windows operating systems on the server . . . . .	72
10. Application files on the primary domain controller hard disk . . . . .	105
11. Application management command summary . . . . .	120
12. Directory path for default response file . . . . .	138
13. The winnt parameters . . . . .	144
14. The rename.txt file description . . . . .	150
15. Windows 9x setup switches . . . . .	150
16. Summary table of steps . . . . .	156
17. ScriptIt file options . . . . .	174
18. Policy status . . . . .	199
19. worksnt.adm file options . . . . .	200
20. worksnt.adm file options . . . . .	201
21. Comparing application model architectures . . . . .	223



---

## Preface

This redbook will help you add the IBM WorkSpace On-Demand 2.0 Feature for Windows Clients to your current WorkSpace environment. It is the result of a residency conducted at the International Technical Support Organization, Austin Center, during the final development of the product. The redbook will help you to:

- Understand the IBM WorkSpace On-Demand 2.0 Feature for Windows Clients product and the situations in which it is most appropriate to deploy.
- Understand the remote IPL and software distribution concepts behind the IBM WorkSpace On-Demand 2.0 Feature for Windows Clients and the implementation of this product.
- Plan and install IBM WorkSpace On-Demand 2.0 Feature for Windows Clients in your enterprise.
- Define client workstations on your WorkSpace On-Demand server, and boot and install these client workstations remotely over the network using a Windows 9x or Windows NT workstation.
- Add applications to a managed Windows environment.
- Add support for additional network adapters and other types of hardware to expand the hardware support provided by the IBM WorkSpace On-Demand 2.0 Feature for Windows Clients.

This redbook also compares the IBM WorkSpace On-Demand 2.0 Feature for Windows Clients against similar technologies available from other vendors, allowing you to make an informed decision.

---

### The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, Austin Center.

**Axel Buecker** is a Software IT Specialist in IBM Germany. He holds a degree in computer science from the University of Bremen. He has 12 years of experience in a variety of areas related to OS/2 and Network Computing, including design and implementation of complex networks in large organizations, utilizing different aspects of OS/2, LAN Server, Warp Server environments, system management, and software distribution.

**Khathutshela Muvenda** is an IT Specialist in IBM South Africa. He has four years of experience in OS/2 and LAN Servers. He holds a degree in

computer science from the University of the North. His areas of expertise include OS/2, LAN Server, WorkSpace on Demand, and Warp Server.

**Joao Miguel Galvao** is a computer technician and manager in Portugal. He has 10 years of experience in operating systems and networking. He has been an IBM BesTeam member for the last five years. His areas of expertise include hardware, software, OS/2, Linux, and Windows.

**Indran Naick** is a Senior IT Specialist at the International Technical Support Organization, Austin Center. He holds a degree in computer science from the University of the Witwatersrand. He has 10 years of experience with IBM and writes extensively on OS/2, Warp Server, and WorkSpace On-Demand. Before joining the ITSO in 1999, Indran worked in IBM South Africa as a Software Solutions Architect.

Thanks to the following people for their invaluable contributions to this project:

Randy George  
Architect, IBM Austin

Ron Aguirre, Tsutomu Oya, Oscar Cepeda, Mike Foster, Milos Radosavljevic,  
and Temi Rose  
International Technical Support Organization, Austin Center

Remi Trombetta  
IBM France

Drago Belak  
IBM Slovenia

Jose Carlos Faisca  
IBM Business Partner

David Dutcher  
IBM Austin

Khoa Huynh  
IBM Austin

Marc-Arthur Pierre-Louis  
IBM Austin

Kerry Fehlis  
IBM Austin



Paul Craton  
IBM U.K

Angel Hernandez Bravo  
IBM Spain

Norm Mattson  
Golden Code Development Corporation

Reinaldo de Medeiros  
IBM Brazil

Charlie Rouh  
IBM U.S.

Neil Stokes  
IBM Australia

Antonio Arias  
IBM U.S.

Mala Anand  
IBM Network Computing Systems Division, Austin

Martin Bense  
GAD Gesellschaft fuer automatische Datenverarbeitung eG

John Case  
IBM U.S.

Martin Dippold  
IBM Global Services, Germany

Andy Erhenzeller  
IBM U.S.

Ivan Faisal  
IBM Indonesia

Steve French  
IBM Network Computing Systems Division, Austin

Ann Gleason  
IBM Network Computing Systems Division, Austin

Aidon Jennery  
IBM Network Computing Systems Division, Austin

Scott Jonston  
IBM Canada Ltd

Larry D. Jones  
Innova Solutions Inc.

Brian Howe  
IBM Japan

Marcello Savio  
IBM Brazil

Tim Sennitt  
IBM U.K.

Peter Greulich  
IBM U.K.

Toshi Shimizu  
IBM Network Computing Systems Division, Austin

Timothy Sipples  
IBM U.S.

Oliver Stein  
IBM Germany

Larry Sullenger  
IBM Network Computing Systems Division, Austin

Keiichi Togashi  
IBM Japan

Robert Rose  
IBM Network Computing Systems Division, Austin

Dick Reardon  
IBM Network Computing Systems Division, Austin

Dan Wiggins  
IBM Network Computing Systems Division, Austin

Uwe Zimmermann  
International Technical Support Organization, Austin Center

Starwalker Jj  
Graphic Illustrations

---

## **Comments welcome**

### **Your comments are important to us!**

We want our redbooks to be as helpful as possible. Please send us your comments about this or other redbooks in one of the following ways:

- Fax the evaluation form found in “ITSO redbook evaluation” on page 255 to the fax number shown on the form.
- Use the online evaluation form found at <http://www.redbooks.ibm.com/>
- Send your comments in an Internet note to [redbook@us.ibm.com](mailto:redbook@us.ibm.com)



---

## Chapter 1. Introduction

WorkSpace On-Demand is IBM's Intel-based network operating system that is optimized for network computing. It is a simple, economical software alternative to a traditional client/server environment that extends the use of your current investment in personal computer hardware and software, and it allows you to add new hardware and software while lowering your total cost of ownership.

The Feature for Windows Clients enhances your WorkSpace On-Demand 2.0 environment by giving you the option to integrate Windows NT or Windows 95/98 based clients in a similar way as your current WorkSpace On-Demand clients.

This chapter provides a brief introduction to the Feature for Windows Clients. It covers functional descriptions of the components as well as the hardware and software prerequisites. The licensing and packaging of the product is also covered here.

---

### 1.1 WorkSpace On-Demand 2.0 overview

WorkSpace On-Demand Version 2 implements a server-managed client environment using Intel-based client workstations and servers as well as RISC based network computers. As such, it provides the benefits of a server-managed client environment, while adding some specific benefits of its own:

- Effective software and data management
- Easier end user support
- Enhanced security
- Broad application support
- End user mobility
- A migration path to full network computing
- Investment protection for current hardware

A detailed description of these benefits can be found in the redbook, *WorkSpace On-Demand Handbook Release 2.0*, SG24-5117.

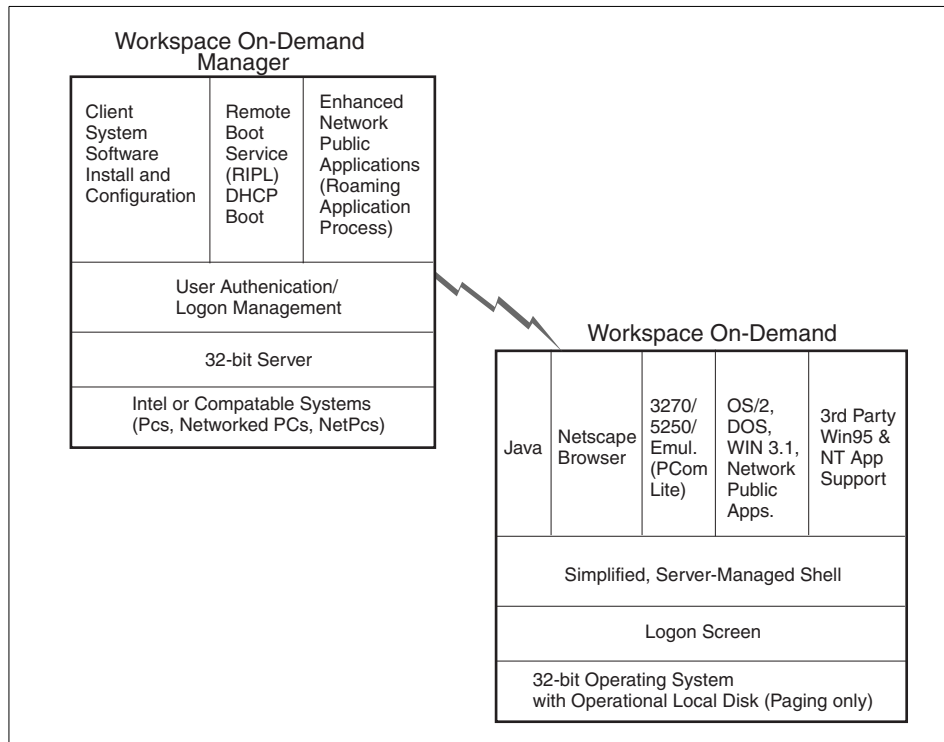


Figure 1. WorkSpace On-Demand 2.0 components

WorkSpace On-Demand consists of two basic components, shown in the figure above:

- The client component, known simply as WorkSpace On-Demand.
- The server component, known as WorkSpace On-Demand Manager, runs on an OS/2 Warp Server system. The WorkSpace On-Demand Manager is a set of server utilities and is used to install, configure, and maintain the network client hardware and software.

Note that both these components are physically installed on an OS/2 Warp Server system, which is then known as a WorkSpace On-Demand server.

WorkSpace On-Demand was designed to co-exist with other client environments in your environment. Together with OS/2 Warp Server, WorkSpace On-Demand supports the following client environments:

- DOS

This lets you boot your client with a DOS interface, for example, the Year 2000 ready solution IBM DOS 2000.

- DOS / Windows 3.xx

This adds a Windows 3.xx graphical user interface to your DOS operating system.

- WorkSpace On-Demand

The traditional OS/2 based WorkSpace On-Demand operating system lets you run all of your OS/2, DOS, DOS/Windows 3.xx, and JAVA-based applications.

- RISC-based Network Stations

The Network Station Manager 3.0 that comes as part of WorkSpace On-Demand 2.0 gives you the capability to include RISC-based network stations in your client environment.

---

## 1.2 The Feature for Windows Clients

Today, most customers need to deploy applications based on the Windows application model. These applications cannot be executed on machines other than Windows NT or Windows 95/98 based PCs.

With WorkSpace On-Demand 2.0, if you wanted to integrate Windows 32-bit software into your current WorkSpace On-Demand environment, you had to introduce a third party multi-user Windows application server to provide access to those applications. This additional effort can be worthwhile when you have to deploy a few specific applications to a few people in your enterprise. But when you start planning for a large number of branch offices distributed over a large area, this solution quickly becomes complicated and expensive. See Appendix A, "Comparing the Feature for Windows Clients" on page 215 for a more detailed overview of the Windows application server architecture.

The Windows Client Feature for WorkSpace On-Demand 2.0 extends WorkSpace On-Demand to provide management capability to Microsoft Windows operating systems and applications.

In addition to the list of clients supported by WorkSpace On-Demand 2.0, the Feature for Windows Clients adds support for the following client environments:

- Windows NT Workstation 4.0

A secure, enterprise-targeted operating system platform is now included into your choice of desktop environments with the Feature for Windows Clients.

- Windows 95/98

Even Windows 95/98 with its broad support on available hardware can be deployed as the second new choice in the WorkSpace On-Demand installation.

All of the supported client operating systems can be serviced from the same server in your network. For smaller branch offices, this may impact savings in costs as well as maintenance. The administrator only needs to keep one server updated and running. There is no longer a need to deploy different types of servers for different client and application requirements.

#### **Naming Conventions**

To make the naming of all these different flavors a little more convenient for you, we will use the following terms throughout the book.

- *WorkSpace On-Demand Client*

To refer to the OS/2-based client

- *WorkSpace On-Demand Manager*

To refer to the WorkSpace On-Demand server installation

- *Feature for Windows Clients*

To refer to the Workspace On-Demand Feature for Windows Clients

- *Windows 95/98 Client Feature*

To refer to the Windows 95/98 client

- *Windows NT Client Feature*

To refer to the Windows NT client

- *Windows applications*

To refer to Windows 32-bit applications

### **1.2.1 Components**

The following components have been added or updated on the WorkSpace On-Demand Manager to provide support for Windows Clients. There is currently no graphical interface available for the Feature for Windows Clients.



- Workstation Management

In addition to being able to define all the clients supported by WorkSpace On-Demand 2.0, a new command line function now allows you to define Windows 9x and Windows NT clients.

A state utility is also provided that allows an administrator to query the current state of a Windows client.

- Desktop Management

An administrator is able to create a number of desktops and assign them to existing users.

- Application Management

This function has been extended to support Windows applications. A number of utilities/commands have been added that enable an administrator to define, delete, and query Windows applications on the server.

Once the administrator has defined the users and applications on the server, the command line functions are available that allow the administrator to assign, unassign, and query applications to a user.

The architecture on the client and the components are described briefly below:

- Feature for Windows Clients

While WorkSpace On-Demand 1.0 and WorkSpace On-Demand 2.0 provided server-based management of WorkSpace On-Demand machine images using Remote Program Load (RPL) or Dynamic Host Configuration Protocol (DHCP) boot for operations, the Feature for Windows Clients support is implemented differently. Since these operating systems do not remote boot from the Warp Server, a remote-install paradigm is used.

The client systems are set up to first boot from the network. DOS RPL is used to initiate an unattended remote install to the local client hard disk of the target system.

**Hard Disk Required**

Using Windows NT, 95, or 98 operating systems in your WorkSpace On-Demand environment requires a local hard drive to be installed in the desktop machine.

The install process uses input parameters from a response file to drive the configuration of the client system. Each client (each unique MAC address)

will have its own response file containing information about its configuration (for example, IP address and hostname).

Once the install process is completed, during subsequent reboots, the client system still sends a RPL boot request to the server. After a brief handshake with the server, the client continues the boot process from the operating system image on the local disk. The boot process completes with the client machine booting up to the WorkSpace On-Demand logon panel.

The boot process will be discussed in more detail in Section 2.2, “The boot process” on page 18.

- Network Logon Client

To incorporate the changes necessary for the secure Feature for Windows Clients environment, updated versions of the IBM Networks Primary Logon Client for Windows NT and the IBM Networks Client for Windows 95 are shipped with this product. They differ from the versions distributed through IBM Software Choice and they are not available separately.

- Desktops

To adjust the look and feel of the Feature for Windows Clients, a standard WorkSpace On-Demand desktop is available. Administrators can also create their own desktop images and logon bitmaps.

When users change the appearance of their desktops or alter application-specific settings, this information is stored in the *user profiles*. There is a method provided with the Feature for Windows Clients where an administrator is able to define all the basic settings and then deploy the user profile on the primary domain controller.

When users are logging on to the primary domain controller using the Feature for Windows Clients, they get their local user profiles updated by the ones on the server side. Temporary changes now made by a user are not saved back to the server. Each time a user logs on she or he will find the same corporate desktop setup.

To restrict the users' access to their desktops and several system functions, you use the Windows system policies. These files are controlled by the administrator using the System Policy Editors for Windows 95/98 or Windows NT and by deploying them in the right places on the primary domain controller that every user has access to. There are two preconfigured, restricted policy files shipped with the product, one for Windows 9x and one for Windows NT environments.

The combination of using system policy files along with user profiles lets you create a restricted client desktop environment like the one in WorkSpace On-Demand.

- **Java Virtual Machine (JVM)**

The IBM Win32 Runtime Environment, Java Technology Edition, Version 1.1.7 is included with the product. This enables customers to run and develop applications that are 100 percent pure Java.

Customers who have implemented 100 percent Java Applications are able to run them on their Windows 95/98 and Windows NT 4.0 clients that are being managed by WorkSpace On-Demand. They are still able to use Microsoft's JVM from Internet Explorer as well. Explorer uses its own JVM, instead of the system's JVM.

- **Tivoli Management Agent (TMA)**

Installing the Tivoli management agent on your Windows NT clients is another option you may want to consider. This agent enables your clients for all Tivoli-based systems management disciplines, such as software distribution, remote monitoring, and remote console, without the need to revisit these systems again.

## **1.2.2 Server platforms supported**

The Feature for Windows Clients is a feature for WorkSpace On-Demand 2.0 and, thus, requires that WorkSpace On-Demand 2.0 be installed on the server. The supported server environments are (with the latest fixpacks):

- IBM OS/2 WARP Server SMP
- IBM OS/2 WARP Server 4.0 Entry
- IBM OS/2 WARP Server 4.0 Advanced
- IBM OS/2 WARP Server for e-business

## **1.2.3 Client platforms supported**

In addition to the standard OS/2-based platforms, the following sections describe the exact versions of the supported operating systems.

### **1.2.3.1 Windows 95**

There are multiple versions of Windows 95 currently available. The version of Windows 95 supported in this environment is:

- Windows 95 OSR2 OEM Version which is available only through the OEM preload channel.

Other versions of Windows 95 are not supported. You need to have the original CD to have it installed on your WorkSpace On-Demand boot server.

#### **1.2.3.2 Windows 98**

The Windows 98 retail version is the version supported in this environment. You need to have an original CD to install it on your WorkSpace On-Demand boot server.

#### **1.2.3.3 Windows NT Workstation 4.0**

Windows NT Workstation 4.0 is the version supported in this environment. You need to have an original CD to install it on your WorkSpace On-Demand boot server. There is a mechanism to deploy the latest fixpacks with the base code.

---

### **1.3 Client hardware considerations**

Hardware requirements differ significantly between environments. Usually, you need to take into account what applications are required, how often they are used, how many are used concurrently, and so on. There are some basic hardware considerations you should check before deploying any of the new functions. For a large corporate rollout, it is best to simulate a real world environment before beginning deployment. This environment can be used to gather data for the current rollout and future expansion.

#### **1.3.1 Remote boot capability**

All clients that are going to be used in this environment need to be able to RPL from the network. The network driver for the Network Interface Card (NIC) must be *remote install enabled*. This implies that the card should have a remote boot chip, and the chip must be enabled. This is usually done using DIP switches or a software setup program.

The client machine must also have a BIOS setting that allows the network adapter to be one of the boot devices.

#### **Support**

For support, IBM requires that both requirements are met, and the network adapters selected are on the list of supported adapters.

### **1.3.2 Windows 95 and Windows 98 hardware**

The following is a list of the base hardware requirements for Windows 9x type clients. Note that these specifications are dependent on your application requirements.

#### **1.3.2.1 Processor**

The Windows 95/98 Client Feature runs on any INTEL-based system (personal computer or network computer) that is capable of running Windows 95 or Windows 98. For performance reasons, we recommend at least an INTEL 486 33 MHZ.

#### **1.3.2.2 Memory**

The client's operating system requires at least 8 MB of memory. We found that at least 16 MB of memory is sufficient. The amount of memory also depends on the number of applications that you are running on your clients.

Because memory prices have dropped, it is the most cost effective method of increasing workstation performance. With 64 MB of memory, the Windows 95/98 Client Feature performs best in a standard office environment.

#### **1.3.2.3 Diskspace**

As mentioned earlier in this chapter a hard drive is required on your clients.

The default installation partition size is 250 MB.

The maximum hard disk size supported on the Feature for Windows Clients workstations is 8 GB.

Only FAT16 partitions will be created on the target client.

### **1.3.3 Windows NT 4.0 hardware**

The following is a list of the base requirements for Windows NT clients. Note that these specifications are dependent on your application requirements.

#### **1.3.3.1 Processor**

An INTEL 486/66 or an INTEL Pentium 90 processor is the minimum recommendation for a Windows NT 4.0 client.

#### **1.3.3.2 Memory**

A minimum of at least 12 MB of memory is required on x86-based computers, and 32 MB is recommended for a better performance. However, the exact amount of RAM required on your client depends on the number of applications that you intend to run concurrently.

Again, the more memory you have available for your clients, the better performance you get. The Windows NT Client Feature in the office environment also performs well with 64 MB of memory installed. There may be performance improvements with 128 MB installed.

#### **1.3.3.3 Disk space**

As mentioned earlier in this chapter, a hard drive is required on your clients.

The default installation partition size is 250 MB, which can be customized. It is recommended to increase the default partition size for Windows NT machines to 600 MB to accommodate the service pack. If possible, make the default partition size equal to the smaller of 8 GB or the largest disk available in your rollout environment. The other disk space is not going to be used any way.

The maximum hard disk size supported on the Feature for Windows Clients clients must not be greater than 8 GB.

Only FAT16 partitions will be created on the target client. Use of any other file system is dependent on the particular operating system's ability to modify/convert the partition as part of the unattended install. (For example, Windows NT 4.0 allows the option to convert a FAT16 partition to NTFS as part of the install process by customizing the installation response file UNATTEND.TXT.)

#### **1.3.4 Device support**

Devices that are automatically recognized and configured by the setup process of Windows 95, Windows 98, and Windows NT are supported by the default response file that will be shipped with the product. Support of new devices require manual modification of the response file by the system administrator using information from Microsoft's Windows 95, Windows 98, and Windows NT Resource Kits. Later in this redbook you will find sample response files showing how additional and/or newer devices can be added to the install process.

---

### **1.4 Licensing and packaging**

The Feature for Windows Clients is available as an upgrade to WorkSpace On-Demand 2.0.

Each WorkSpace On-Demand workstation that needs to use the Feature for Windows Clients requires a WorkSpace On-Demand 2.0 client licence and the appropriate Windows licence from Microsoft, that is, a Windows 95, Windows 98, or Windows NT licence.

Each WorkSpace On-Demand 2.0 registered user is entitled to use the chargeable services of any OS/2 Warp Server Version 4, OS/2 Warp Server Advanced Version 4, or OS/2 Warp Server Advanced Version 4 SMP feature in your enterprise.

Figure 2 shows that for the traditional OS/2 type client where no additional licensing is required. For each of the Windows platforms, an additional licence is required.

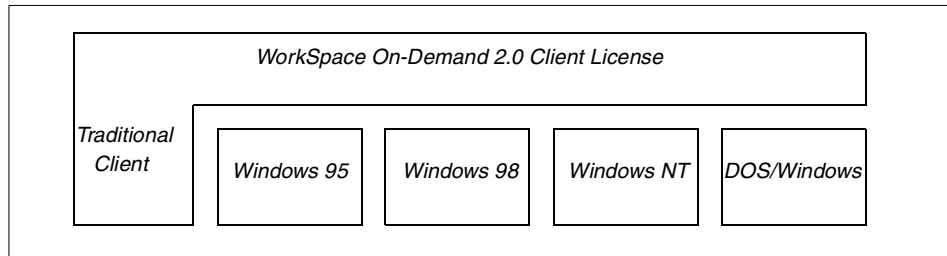


Figure 2. Client licensing





---

## Chapter 2. Feature for Windows Clients architecture

This chapter will focus on providing you with an overview of the IBM WorkSpace On-Demand 2.0 Feature for Windows Clients architecture. This chapter helps you better understand how the product functions as well as provides an overview of what steps are required to perform particular functions.

The topics include the new remote-install paradigm, the boot process, and user administration.

---

### 2.1 Overview

WorkSpace On-Demand 2.0 provides the ability to manage machines, users, and applications between OS/2 clients and OS/2 Warp Servers. The IBM WorkSpace On-Demand 2.0 Feature for Windows Clients extends this notion to Windows 95, Windows 98, and NT clients.

Figure 3 on page 14 gives you a high-level view of the newly added components to WorkSpace On-Demand.

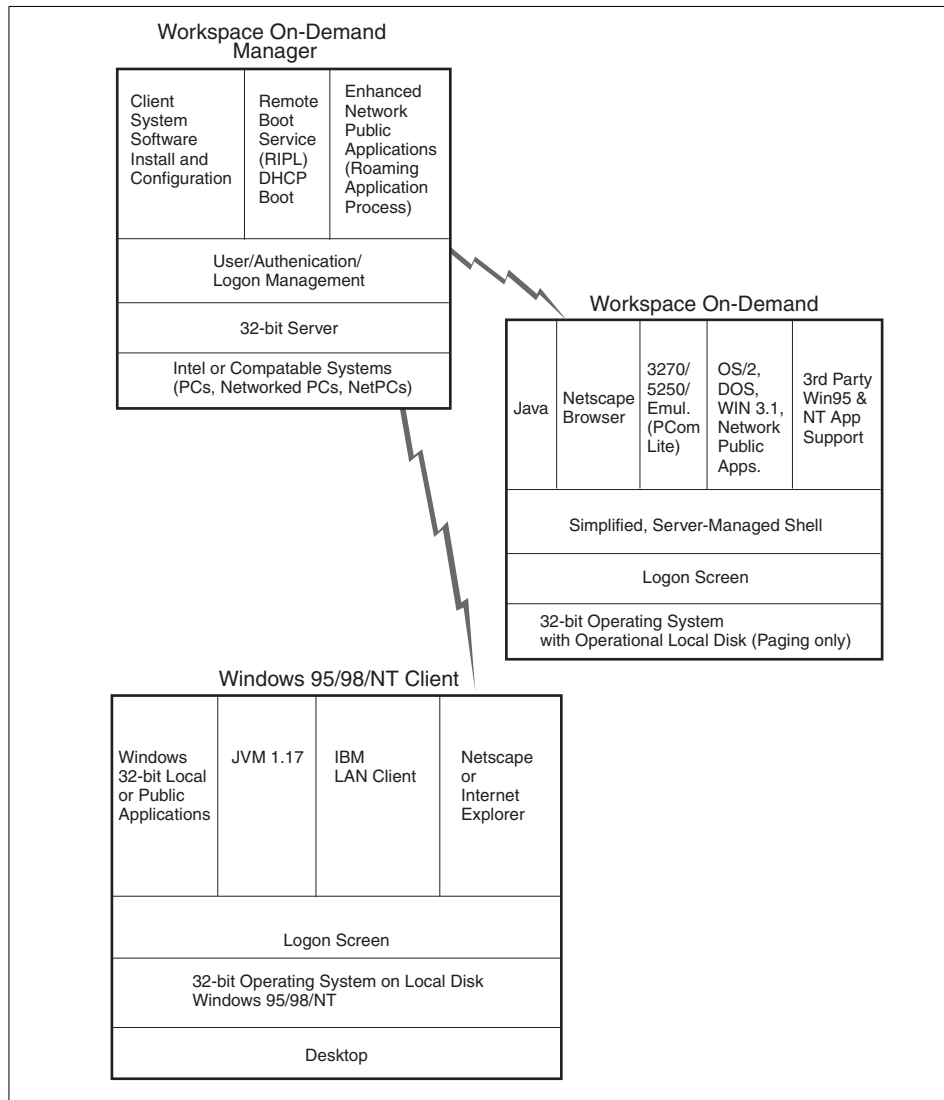


Figure 3. Feature for Windows Clients, architecture

The new components on both the client and the server are described in more detail in the sections that follow.

### 2.1.1 Client-side architecture

The Feature for Windows Clients introduces three new clients based on Microsoft's Windows 95, Windows 98, and Windows NT 4.0 Workstation.

These new clients operate differently and support a different application set to those clients supported on the current WorkSpace On-Demand. The following paragraphs describe the new client features.

- **Boot Process**

The boot process is significantly different than the other versions of WorkSpace On-Demand. With the Feature for Windows Clients, the operating system is downloaded and installed on the client workstation using RPL. Once the operating system is downloaded, the operating system is booted from the local hard disk and not from the server as was previously the case. This process is described more fully in Section 2.2, “The boot process” on page 18.

- **Operating System**

The operating system on the client is based on either Windows 95, Windows 98, or Windows NT. Each workstation that is defined can have only one of the three systems defined. The supported versions of the above operating systems are listed in Chapter 1, “Introduction” on page 1.

- **Network Logon Client**

New network logon clients have been developed based on the IBM Networks Client for Win95 and the IBM Networks Primary Logon Client for Windows NT. These new clients are called:

- WorkSpace On-Demand Logon Client for Windows 95 (or 98)
- WorkSpace On-Demand Logon Client for Windows NT

These new clients provide the end user authentication and the download of the user specific data. Modifications have been made to provide the WorkSpace On-Demand look-and-feel of the desktop and logon panel and to disable the user from being able to save changes made to his or her desktop.

The clients also manage user application updates.

These new clients are not general purpose and will not be distributed separately. An automated installation is provided to install them onto the end user machine.

- **Restricted Desktop**

Windows 95/98 and Windows NT use a combination of Windows User Profiles and system policy files to control the desktop appearance and core system functionality. A default policy file is provided that restricts the users’ access to their system settings and hard drive. This policy information is downloaded as part of the WorkSpace On-Demand Logon Client. This policy file can be edited using existing available editors.

For example, client-side graphics, logon panel bitmaps, and default desktop background images are installed and provide a common look-and-feel.

- **Web Browser**

Instructions are provided for the setup of either Internet Explorer or Netscape Navigator. Since these browsers are freely available, they have been included in the architecture chart. It is also assumed that customers wanting to reduce their total cost of ownership will move their applications architectures toward applications that are browser based.

- **Java Virtual Machine (JVM)**

The IBM Win32 Runtime Environment, Java Technology Edition, Version 1.1.7 is included with the product. This enables customers to run and develop applications that are 100 percent pure Java.

Customers will still be able to use Microsoft's JVM from Internet Explorer.

- **Windows Applications**

Applications that have been developed for Windows 95/98 or Windows NT are able to run natively on any of their respective platforms. This support differs from that available on previous WorkSpace On-Demand versions that required an additional third party server.

The application is also managed differently. With the Feature for Windows Clients, depending on the application, some code may have to be installed on the client machine.

- **Tivoli Management Agent (TMA)**

The Tivoli management agent is an extension of the Tivoli Management Framework. A machine running TMA is called an endpoint. The Windows (95, 98, NT) Tivoli management agent has been added to the Feature for Windows Clients and is available for installation on the client machine.

Once installed, these endpoints can receive distributions, execute tasks, run monitors, send events, and so forth. The TMA has a very small footprint, but it can be fully managed by all Tivoli applications.

### **2.1.2 Server-side changes**

Additional functions have been introduced into the server to support the new Feature for Windows Clients. These functions are described briefly in the following paragraphs:

- **User Management**

The basic user management structure has remained unchanged. However, routines have been added to enable the administrator to assign application packages and the user's interface (desktop). They also manage the users access to locations on the server containing Windows application packages.

These routines are all command line driven. User information with regard to Windows applications and their assigned desktops are stored outside of the current user database structures.

- **Application Management**

The software management currently supported by WorkSpace On-Demand for OS/2, 16-bit Windows, and DOS applications will be unchanged. Support is added for the creation of Windows application packages, the management of these applications on Warp Servers, and the ability to manage which Windows applications a given user can execute. All management of Windows applications is done through Command Line Interfaces (CLI).

- **Group Management**

There are no group management routines in the Feature for Windows Clients. It is not possible to add a Windows application to a group of users or perform any other group functions.

- **Desktop Management**

Support has been added to WorkSpace On-Demand that enables the management of restricted desktops on Win95/98 and Windows NT clients. System administrators at the domain controller have the ability to select a desktop for a user, add applications to a desktop for a user, remove applications from a user, and enumerate the applications a user has on a given desktop. Individual Win95/98 and Windows NT desktops are administered for each user. At logon time, the appropriate desktop is delivered to the user.

Systems administrators can control the capabilities of users by creating customized restricted desktops. Through a set of pre-built system policy files, the system administrator can control some of the functionality Windows 9x and Windows NT 4.0 users have.

The restricted desktop has a WorkSpace On-Demand look-and-feel that includes a familiar background.

- **Command Line Interface**

Using the command line interface is the only way to administer all of the new functions introduced with the Feature for Windows Clients. There are five new commands available to the WorkSpace On-Demand administrator

to cover the complete field of managing the Feature for Windows Clients. These commands are described in the *WorkSpace Guide* that is shipped with the product. These commands are also used extensively throughout this redbook.

- **WorkSpace On-Demand Boot**

The boot architecture for current WorkSpace On-Demand 2.0 clients remains unchanged. The Windows clients, Windows 9x and Windows NT Workstation, have a different boot/install paradigm. This is described in more detail in the sections that follow.

---

## 2.2 The boot process

One of the key differences between the clients supported by WorkSpace On-Demand 1.0 and WorkSpace On-Demand 2.0 and the new Feature for Windows Clients is the location and manner in which the operating software is loaded into memory on the client machine.

With current versions of WorkSpace On-Demand, the client code was always located only on the hard file of the server. Using either the RPL or DHCP boot protocol, the client machine loaded the operating software and applications into memory.

The Windows client is implemented differently. The operating software used here was developed by Microsoft, and it is not inherent in the software design to have it boot from any remote server. Therefore, instead of having all the code reside on the server, the Feature for Windows Clients uses a remote-install paradigm to setup the code on the client's hardfile. Once this installation process is completed, during subsequent reboots, the client machine checks the server for updates. If an update was made to the software, the client machine is reinstalled. If no update was made, the machine continues the boot process from the operating system image on the server. The boot process completes with the client machine booted up to the WorkSpace On-Demand logon panel. This process is explained in much more detail in the sections that follow.

The boot process is dependent on the condition of the machine, that is, whether the machine is newly defined or already installed. If a machine is newly defined, the operating software still has to be installed and configured. If a machine is already installed, it only has to check whether there has been an update. This section describes, in some depth, both processes.

## 2.3 The installation boot sequence

In this section, we describe the boot process for a machine that has been newly defined on the server.

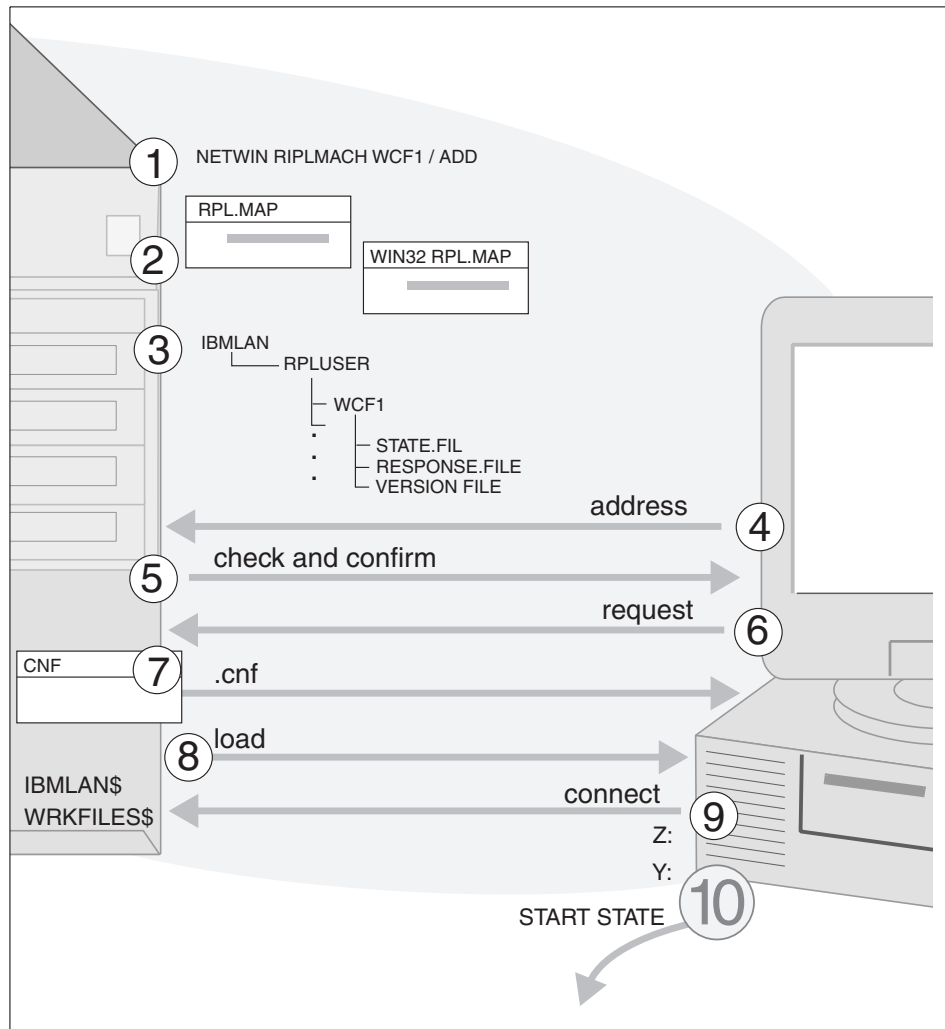


Figure 4. Install boot sequence

Figure 4 describes the first 10 steps, and Figure 5 on page 23 describes the remaining the steps. The steps are as follows:

1. The systems administrator defines a Windows client through the command line interface. This is done using the `NETWIN RIPLMACH` command with the `/ADD` switch.

For example:

```
netwin riplmach WCF1 /add /mac:0004ac7763ec /release:nt40 /type:winnt  
/dhcp:n /enabled:y /image:w32dossb.img /recordid:r_dtk_ndis  
/partition:600 /cdkey:... /REGUSER:"Indran Naick"
```

2. The `NETWIN` command uses the existing RPL Process running on the server to create the client. This puts an entry in the `RPL.MAP` and `W32RPL.MAP` files.

The entry in the `RPL.MAP` file for the above client is:

```
0004AC7763EC WCF1 ~ W32DOSSB GA_SRV GA_DOM ~ ~ ~ , , , Z R_DTK_NDIS ~ ~ ~
```

The entry in the `W32RPL.MAP` file for the above client is:

```
0004AC7763EC WCF1 W32DOSSB R_DTK_NDIS
```

Both these entries are constructed from the command line definition in step 1 on page 20.

3. A directory for the client is created under the `RPLUSER` subdirectory in which the `NETWIN` command places three files. This directory is the same as the name assigned to the client. In our example, the directory `c:\BMLAN\RPLUSER\WCF1` contains the following files:

**STATE.FIL** - The state file contains, most importantly, the current state of the machine that, when newly created, is set to `NEW`. Other information contained in the file is the operating system type, the operating system release, the partition size, the install image to use, and the client type. All of this information is gathered from the parameters on the command issued in step 1 on page 20.

**unattend.txt** - The `NETWIN` command also creates a response file that the client will use for installation. The information in the response file is also gathered from the parameters on the command issued in step 1 on page 20.

**VERSION FILE** - This is used for administration.

4. The client is powered on, and the RPL feature (on the network adapter) obtains control. The RPL feature attaches the workstation to the network and sends a `FIND` frame. `FIND` frames contain the LAN adapter address and are repeated periodically until a server responds.
5. The server receives the `FIND` frame sent by the workstation. The server then checks the `RPL.MAP` file for a workstation record entry that matches



the network address contained in the FIND frame. The line in the RPL.MAP that corresponds to our definition is as follows:

```
0004AC7763EC WCF1 ~ W32DOSSB GA_SRV GA_DOM ~ ~ ~ , , , Z R_DTK_NDIS ~ ~ ~
```

The workstation record contains two pointers. The first pointer is to a server record entry (R\_DTK\_NDIS). The server record entries are also contained in the RPL.MAP file. The server record entry for R\_DTK\_NDIS is:

```
YYYYYYYYYYYYY dosndtr.cnf 3 10 N IBMLAN$ DOS~TOKEN~RING~NDIS ~ ~ , , , Z  
R_DTK_NDIS ~ ~
```

The server record contains the name of the boot block configuration file (dosndtr.cnf) that defines how the workstation will boot up.

The second pointer is to a DOS image file. In our definition, the file is W32DOSSB. The DOS image file also defines the boot environment that will be sent to the workstation.

6. The workstation receives the FOUND frame with the address of the server and sends a SEND.FILE.REQUEST frame back. This frame is a request for the server to send the bootstrap program.
7. The bootstrap program information is contained in the boot block configuration file (.CNF) on the server. The CNF file derives from the server record entry in the RPL.MAP file. The server sends the bootstrap program to the workstation.
8. The workstation receives the bootstrap program and places it into memory. Once the last frame is received, control transfers from the RPL feature to the bootstrap program. After the boot block loads, control passes to RPLBOOT.SYS. RPLBOOT.SYS hooks interrupt 21h to handle file I/O requests and moves the DOS drivers to high real memory (just below the 640 KB boundary) and initializes them. It then transfers control to the appropriate loader.
9. The loader defines the environment sent down to the workstation. The workstation is able to do the following with the boot image send down:
  - It starts a mini requester.
  - The autoexec.bat calls CONNECT.EXE to establish the following connections:

**X: \\BOOTSERVER\\BMLAN\\RPL**

The access to this share is read-only. The Windows operating system files reside on this share. For example, you can access the Windows NT files from a client by using the paths X:\\NT40\\I386 and X:\\I386\\RSP.

**Y: \\BOOTSERVER\\BMLAN\\RPLUSER\\machine\_name**

machine\_name represents the name of a client. Access to this share is read-write. This share is referred to as the client directory. Log files, result files, and client-specific files reside in this directory.

#### **Z: \\BOOTSERVER\\IBMLAN**

Access to this share is read-only. The share allows clients to access various configuration programs that reside in the DOSLAN\\DOS and DOSLAN\\NET subdirectories. For example, you can access STATE.EXE from a client by using the path Z:\\DOSLAN\\NET.

The autoexec.bat also calls RPLTERM.BAT to terminate A: drive boot image use.

10. Next, the workstation starts STATE.EXE from the autoexec.bat file. STATE.EXE is the function that monitors and changes the state of a workstation. It reads and records the state of the machine in the state.fil under the IBMLAN\\RPLUSER\\workstation directory that was mentioned earlier.

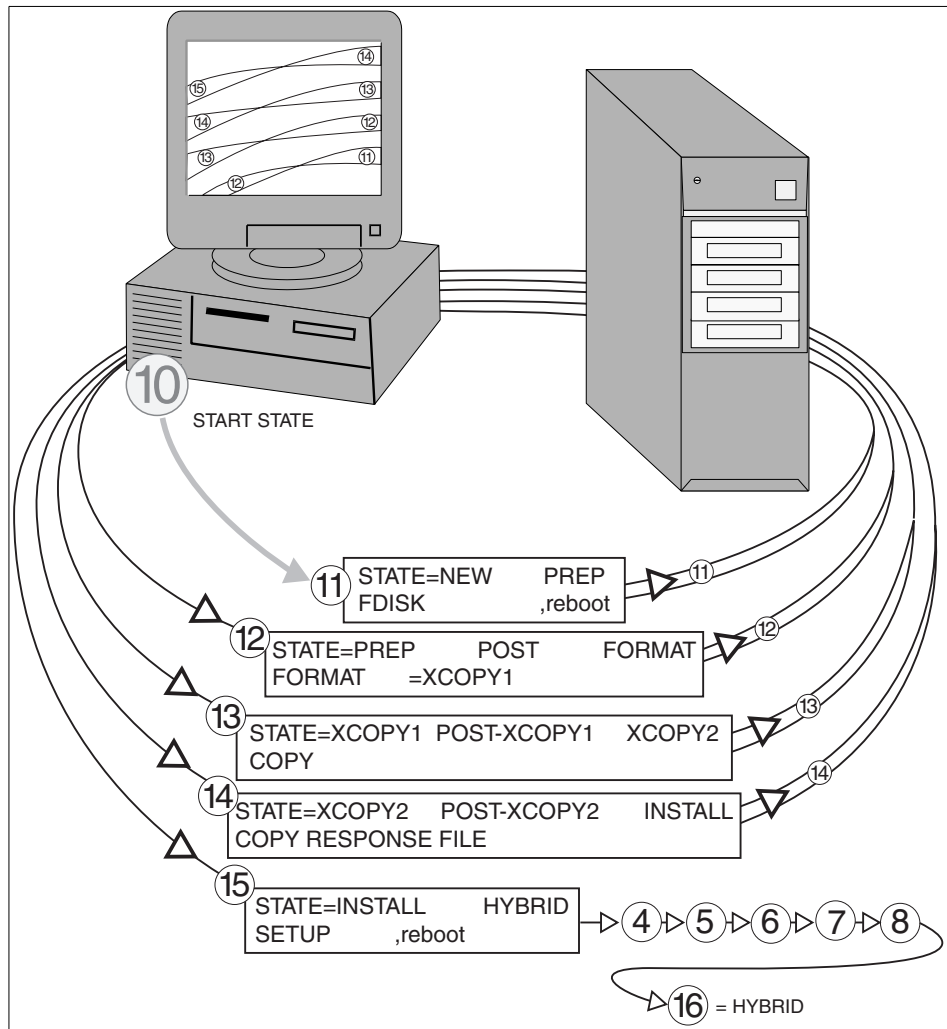


Figure 5. The install process

11. Since the state is NEW, as it is a new client, int13h and function 41h are used to see if the IBM/MS API extension is present in the BIOS. If this function is present, it will use int 13h and function 48h to get the drive capacity. Otherwise, it issues int13h function 8 to get the drive capacity. At this point, the system also checks to see how much memory is available. If there is insufficient memory to run the respective Windows setup programs, DOS is installed onto the hard drive. This is not depicted in Figure 5.

12. It then gets the primary partition size from the state.fil to prepare the hardfile and issue an FDISK. If there is already a visible primary partition, it deletes the partition and repartitions the drive. The state.fil is then updated to STATE=PREP and the workstation is rebooted.
13. The workstation then goes through steps 4 through 10. When STATE.EXE reads the state.fil and finds STATE=PREP, it issues a FORMAT and updates the state.fil with STATE=PRE-COPY. At this stage, it is preparing to copy the operating software images across the network. State is then changed to STATE=XCOPY1. Only fat16 is used, since NT does not support fat32.
14. STATE.EXE still has control, and now since STATE=XCOPY1, it copies the operating system image files from the server to the workstation and updates the state.fil to STATE=POST-XCOPY1 when it does a verify of the files it has copied. State then changes to STATE=XCOPY2. Still in control with a STATE=XCOPY2, the STATE.EXE copies the response files placed in it's directory on the server as well as other control files. and then updates the state.fil to STATE=POST-XCOPY2 when it does a verify of the files it has copied. State then becomes STATE=INSTALL. The remote install of the selected Windows operating system and selected components start at this point. It then cleans up the image files that were used for install. After a successful install, state.fil is updated to STATE=HYBRID (the final state).
15. At this point, the state daemon on the server updates the RPL.MAP on the server and changes the workstation record to the following:  
0004AC7763EC WCF1 ~ W32DOSSB GA\_SRV GA\_DOM ~ ~ ~ , , , Z R\_D\_REBOOT ~ ~ ~  
The corresponding server record is:  
YYYYYYYYYYYY doshbrid.cnf 3 10 N IBMLAN\$  
DOS~HYBRID~BOOT~TOKENRING/ETHERNET ~ ~ ~ , , , Z R\_D\_REBOOT ~ ~ ~  
This is a unique server record. Each time the workstation boots, it sends an environment down that redirects the boot to the workstations own hard disk.  
At this point, the workstation starts the configuration process. There are a few more re-boots before installation is complete. Each time the workstation reboots, it goes through steps one through eight. The image then sent down forces it to boot from it's own local hard disk.
16. The workstation is now considered operational. The operational boot sequence is described in the next section.

### 2.3.1 Operational boot sequence

This section describes the boot sequence of a machine that is operational. In other words, a machine that has gone through all of the steps described in

Section 2.3, “The installation boot sequence” on page 19. The initial boot steps are similar to those performed during the installation boot sequence, but they are repeated here for completeness.

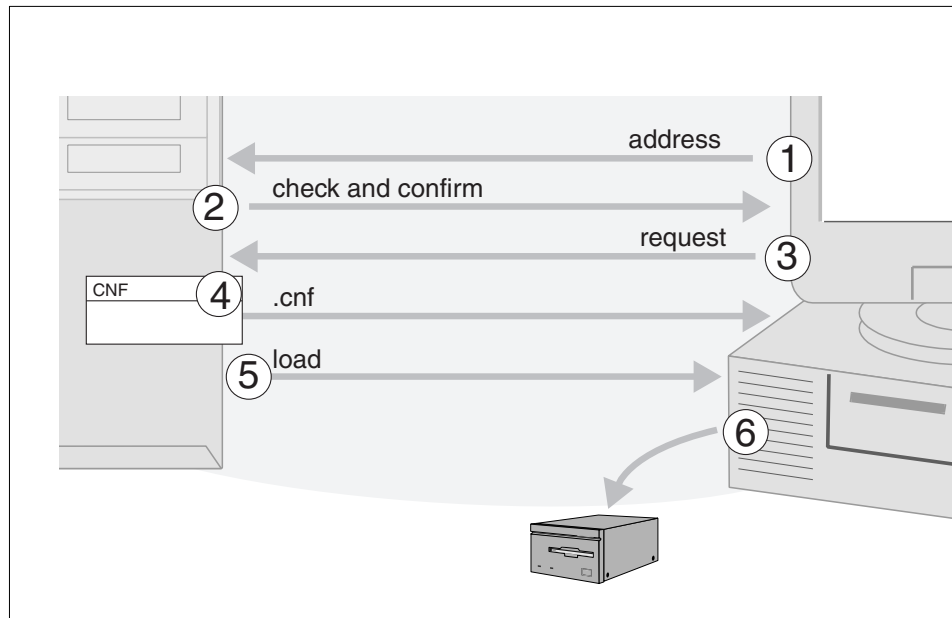


Figure 6. The operational boot process

1. The client is powered on, and the RPL feature (on the network adapter) obtains control. The RPL feature attaches the workstation to the network and sends a FIND frame. FIND frames contain the LAN adapter address and are repeated periodically until a server responds.
2. The server receives the FIND frame sent by the workstation. The server then checks the RPL.MAP file for a workstation record entry that matches the network address contained in the FIND frame.  
  
The workstation record contains two pointers. The first pointer is to a server record entry, contained in the RPL.MAP file, and the second pointer is to a DOS image file.
3. The workstation receives the FOUND frame with the address of the server and sends a SEND.FILE.REQUEST frame back. This frame is a request for the server to send the bootstrap program.
4. The bootstrap program information is contained in the boot block configuration file (.CNF) on the server. The CNF file used is derived from

- the server record entry in the RPL.MAP file. The server sends the bootstrap program to the workstation.
5. The workstation receives the bootstrap program and places it into memory. Once the last frame is received, control is transferred from the RPL feature to the bootstrap program. After the boot block is loaded, control is passed to RPLBOOT.SYS. RPLBOOT.SYS hooks interrupt 21h to handle file I/O requests and moves the DOS drivers to high real memory (just below the 640 KB boundary) and initializes them. It then transfers control to the appropriate loader.
  6. The environment sent down is defined in the RPL.MAP file. It includes a file called HDBOOT.COM that reads the first sector of hard disk 0 on the client and passes control to it, thereby loading the newly installed operating system. Should a client be redefined, deleted, or reinstalled, the workstation record in the RPL.MAP file is updated and the corresponding environment sent down at boot time.

## 2.4 RPL control files

There are a number of files involved in the boot process. This section describes each of the files and their purpose in more detail.

### 2.4.1 RPL.MAP

The RPL.MAP file contains two basic types of records: workstation records and server records, as shown in Figure 7.

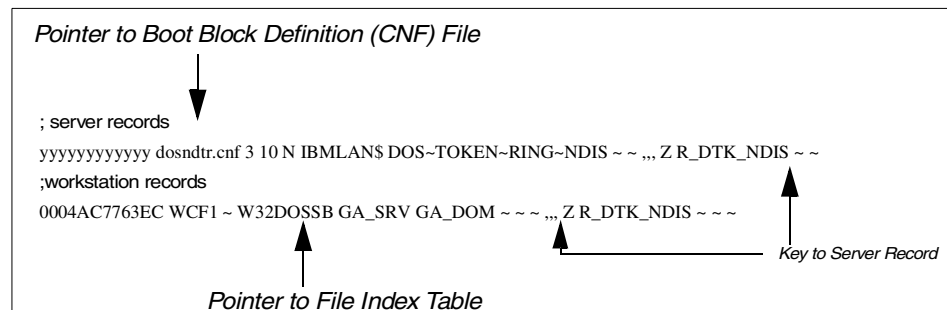


Figure 7. Sample RPL.MAP file

The workstation record is the primary means by which the server determines whether it should respond to a client's FIND frame. Each client that you

define on a WorkSpace On-Demand server has its own unique workstation record in RPL.MAP, which is created at the time you define the client.

**Note**

A client workstation may have more than one workstation record in the RPL.MAP file. However, only one record may be active at any time.

When the server receives the FIND frame, it searches RPL.MAP for a workstation record with a network address that matches the client's address contained within the FIND frame. If it finds a match, it responds to the client with a FOUND frame. If it cannot find a match, it ignores the FIND frame.

After the server responds to the client with a FIND frame, and the client sends a SEND.FILE.REQUEST frame, the server must determine how to assemble the boot block to be sent to the client. The required information includes:

- The type of network adapter installed in the client.
- The operating system to download: DOS, OS/2, or WorkSpace On-Demand.
- For an OS/2 or WorkSpace On-Demand client, the FIT file that should be included in the boot block.
- The files that should make up the boot block.

Again, RPL.MAP plays a key role in this process. A client's workstation record in RPL.MAP contains certain information, such as the client's workstation name and the location of the FIT file, that should be included in the boot block. In addition, the workstation record contains a key field (as shown in Figure 7 on page 26) that references a server record.

RPL.MAP contains one server record for each client operating system supported on the server and for each network adapter installed in the client workstation. For example, there is one server record for an IBM Token-Ring Adapter booting OS/2 Warp 4, and another record for an IBM Token-Ring Adapter booting WorkSpace On-Demand 2.0.

The server record contains pointers to the remaining information required to build the boot block. This information is actually contained in the boot block definition (CNF) file, which is described in detail in Section 2.4.1.1, "Boot block definition (CNF) file" on page 34.

### **The Workstation Record**

The workstation record in the RPL.MAP file contains 15 fields (1 through 9 and A through F). Fields are delimited by spaces. An empty field must contain a tilde (~) character that serves as a place holder. Table 1 describes each of the fields in the workstation record.

Table 1. The RPL.MAP file - Workstation record field descriptions

Field	Name/Value	Description
1	100FFFFFFFFF	Unique, burned-in 12-character adapter address of the primary network adapter in the client workstation to which this entry refers. This field can also contain question mark (?) characters for DOS Remote IPL records that are used as wildcards. Do not use these wildcard characters in OS/2 or WorkSpace On-Demand Remote IPL records.
2	WCF1	Workstation machine ID. Can be up to 15 characters in length.
3	-	Not used. Must contain a tilde character.
4	W32DOSSB	<p>For DOS Remote IPL, this is the name of the diskette image used to IPL the client (an extension of .IMG is assumed). This field is limited to eight characters in length for a DOS client.</p> <p>For WorkSpace On-Demand, this is the name of the FIT file to be used when booting the client. The path name is relative to the RPLDIR path (which is typically \BMLAN\RPL). An extension of .FIT is assumed.</p>
5	GA_SRV	For DOS Remote IPL, this field contains the name of the Remote IPL server where the diskette image file and other files for the DOS workstation are stored. This name is limited to 15 characters.
6	Z or GA_DOM	For DOS Remote IPL, this is the name of the domain. For OS/2 and WorkSpace On-Demand clients, this is the drive letter for the RIPL boot drive. Use a letter from C through Z. Do not use a local hard disk drive.



Field	Name/Value	Description
7,8,9	-	<p>These fields specify parameters for the IBM LAN Support Program drivers 1, 2, and 3:</p> <ul style="list-style-type: none"> <li>- Driver 1 corresponds to DXMA0MOD.SYS</li> <li>- Driver 2 corresponds to DXMC0MOD.SYS</li> <li>- Driver 3 corresponds to DXMT0MOD.SYS</li> </ul> <p>By default, these fields contain tilde characters, separated by spaces, indicating that no parameters are being passed. You can replace the tilde characters with valid parameters that you may wish to pass. For example, you can specify a locally administered address for the client's LAN adapter by passing the address to driver 2.</p>
10	,,,	<p>For DOS RIPL, this field specifies additional memory for device drivers 1, 2, and 3. The default is three commas. You should modify the driver 3 portion of this field if you are increasing NetBIOS parameters beyond the default settings. Be aware that additional memory may be required at the client. You should determine the value for this field experimentally.</p> <p>This field is not used when the NDIS protocol stack is specified in the boot block configuration file referenced in Field 12 of the server record.</p>
11	Z or ~	For DOS RIPL, this is the LASTDRIVE value. The default is Z. For OS/2 and WorkSpace On-Demand, this field must contain a tilde character.
12	R... or D...	This field designates the server record to use for this client. This field must match Field 12 of a server record in the same RPL.MAP file. D... stands for a disabled record, while R... stands for an enabled record.
13	~	Reserved. This field must contain a tilde character.
14	~	Reserved. This field must contain a tilde character.

### **The Server Record**

The server record in the RPL.MAP file contains 14 fields. The syntax rules for this record are the same as those listed for the workstation record. Table 2 describes each field in the server record.

*Table 2. The RPL.MAP file - Server record field descriptions*

Field	Name/Value	Description
1	yyyyyyyyyyyyy	This field consists of 12 y characters.
2	xxxxxxx.CNF	This field specifies the boot block configuration file used to start a client. See Section 2.4.1.1, "Boot block definition (CNF) file" on page 34.
3	3 or 0	<p>This field specifies the number of retries in a given period (specified in Field 4) that a requester with no specific entry in the RPL.MAP file must request IPL before a RPL.MAP entry with a wildcard character is used.</p> <p>If there is only one RIPL server, this value should be 0. Otherwise, the value must be non-zero so that every Remote IPL server has a chance to service the request.</p>
4	10	This field specifies the time interval in seconds during which the number of unanswered IPL requests specified in Field 3 must be received before a default IPL configuration is used.
5	N or A	This field specifies whether the remote IPL server acknowledges each packet sent. The valid values are A (acknowledge) or N (do not acknowledge). Usually, N is specified since A increases network traffic.
6	IBMLAN\$ or ~	For DOS RIPL, this is the net name where the DOS images are located. The images are assumed to be in the \IBMLAN\DCDB\IMAGES directory path from this share. For OS/2 and WorkSpace On-Demand RIPL, this field must contain a tilde character.

Field	Name/Value	Description
7	DOS~IBM~ and others	<p>This is a descriptive comment. This field is required when an OS/2 client is running in a PC Network II or PC Network II/A network environment. OS/2~PCNET indicates running in a PC Network environment, whereas OS2~PCNETA indicates PC Network/A. The GETRPL post-install utility uses this field to enable and disable server records correctly according to the adapter type of the RIPL server. The following naming conventions apply:</p> <p>Token-ring server records must include the string TOKEN.</p> <p>Ethernet server records must include the string ETHER.</p> <p>PC network server records must include the string PCNET.</p>
8	~	Reserved. This field must contain a tilde character.
9	~	Reserved. This field must contain a tilde character.
10	,,,	This field must contain three commas.
11	Z or ~	For DOS RIPL, this field is the LASTDRIVE value. The default is Z. For OS/2 and WorkSpace On-Demand, this field must contain a tilde character.
12		This key field identifies a server record. Each identical yyyyyyyyyyyy line must have a different value for this field. Field 12 must begin with R, which enables the record. For DOS server records, use R_Dxxxx where xxxx may be any unique string.
13	~	Reserved. This field must contain a tilde character.
14	~	Reserved. This field must contain a tilde character.

Apart from pointing to the LAN adapter support to be used at the Remote IPL client, the 12th field in the OS/2 server record IDs also determines the client operating system version to be started at the client. The rules for naming WorkSpace On-Demand server record identifiers (that is, Field 12 in the server record) are as follows:

The record identifier must be 48 characters or less and have the format s\_BBvv\_nn\_aaappppp....

- s defines the state of the remote IPL client:

- R - indicates the client is enabled
- D - indicates the client is disabled

WorkSpace On-Demand clients can be created in the enabled or disabled state.

- vv defines the major version ID of WorkSpace On-Demand:
  - 10 - indicates the initial WorkSpace On-Demand release
- nn defines the language ID version of WorkSpace On-Demand; that is, US, FR, DE, and so on.
- aaa defines the network adapter type:
  - OTK - indicates token ring
  - OET - indicates Ethernet
- ppppp... defines a unique string that identifies the network adapter or protocol stack the remote IPL client is to use. This string must match field 4 of the adapter definition record in the NDISDD.PRO file. Usually, the name of the LAN adapter is mentioned here. With WorkSpace On-Demand, these characters are also used for the country code, such as R\_BB10\_US\_OTKNTR, which stands for loading WorkSpace On-Demand, U.S. English version with IBM Token-Ring LAN Adapter support.

The rules for naming server record identifiers for other versions of OS/2 are as follows:

The record identifier must be 48 characters or less and have the format s\_BBvv\_nn\_aaappppp....

- s defines the state of the remote IPL client:
  - R - indicates the client is enabled
  - D - indicates the client is disabled

WorkSpace On-Demand clients can be created in the enabled or disabled state.

- vv defines the major version ID of OS/2:
  - 0 - indicates OS/2 2.0
  - 1 - indicates OS/2 2.1
  - 30 - indicates OS/2 3.x
  - 40 - indicates OS/2 4.0
- aaa defines the network adapter type:

- OTK - indicates token ring
- OET - indicates Ethernet
- 3CE - is an old naming convention originally used for 3COM Ethernet adapters. OET is the recommended prefix for Ethernet.
- ppppp... defines a unique string that identifies the network adapter or protocol stack that the remote IPL client is to use. The only requirement for this string is that it results in a unique server record identifier.

From each of the server records a number of events take place. These are described in Section 2.2, “The boot process” on page 18. These events are controlled by a number of batch files as described in Figure 8.

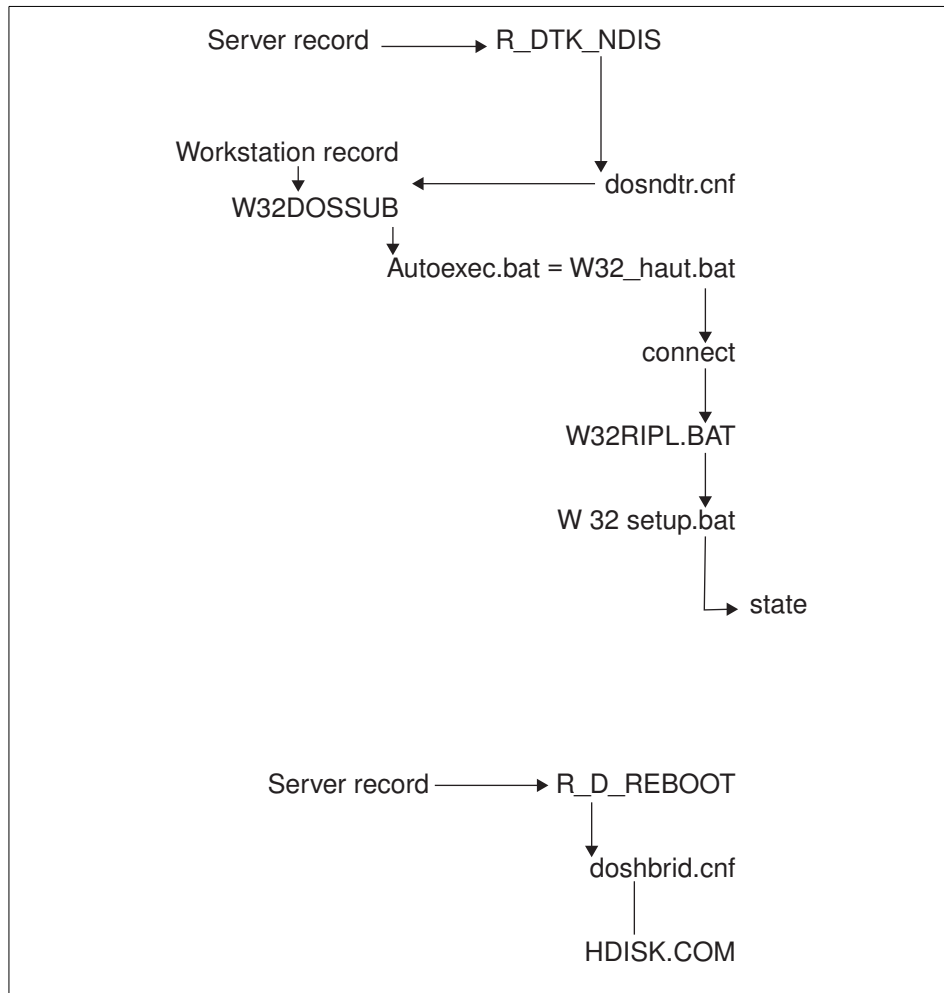


Figure 8. Going from the different server records

#### 2.4.1.1 Boot block definition (CNF) file

Section 2.2, “The boot process” on page 18, describes the boot block that is sent to the client by the server in response to a SEND.FILE.REQUEST frame. The contents of this boot block for the Windows clients are determined by the server using two sources:

- The IMG file to be included in the boot block is determined from the workstation record in RPL.MAP.

- The remaining contents of the boot block are determined by reading the boot block definition (CNF) file referenced in the server record in RPL.MAP.

The boot block definition files are located in the \IBMLAN\RPL directory.

These definition files define an operating system and the way it is loaded into a Remote IPL workstation. A definition file is required for DOS, OS/2, and WorkSpace On-Demand requesters.

Every server record in RPL.MAP must contain a reference to a valid CNF file. A number of default CNF files are provided for all network adapters and operating systems supported by WorkSpace On-Demand.

A number of client workstations can use the same CNF file. For example, if two clients boot the same operating system and have the same type of network adapter, they will use the same CNF file. Typically, you do not need to change CNF files unless you change the network adapter in a client workstation.

Using the data contained in the CNF file, the server creates a boot block that is sent to the client workstation in a series of FILE.DATA.RESPONSE frames. This transmission occurs at the very beginning of the IPL process.

Figure 9 on page 36 shows the default DOS CNF file for the IBM Token-Ring Network Adapter.

```

; DOS Boot Block Configuration (IBM Token Ring)
; NDIS LAN Support Program Drivers
BASE 7C0H
RPL DOS\RPLBOOT.SYS
LDR DOS\RPLLOADR.COM ~
DAT DOS\TOKENRNG\PROTOCOL.INI
DAT C:\IBMLAN\DOSLAN\LSP\DXM.MSG
DAT C:\IBMLAN\DOSLAN\LSP\DOS\LT2.MSG
EXE C:\IBMLAN\DOSLAN\LSP\NETBIND.COM ~ ~ ~
; **NETBIOS and IEEE 802.2*****
; DRV C:\IBMLAN\DOSLAN\LSP\DXMTOMOD.SYS PBA=0~S=12~ST=12~C=14~O=N ~ ~
; DRV C:\IBMLAN\DOSLAN\LSP\DXMEOMOD.SYS ~ 10 ~
; **NETBIOS and IEEE 802.2*****
;
; **NETBIOS*****
DRV C:\IBMLAN\DOSLAN\LSP\DXMJOMOD.SYS ~ 17 ~
; **NETBIOS*****
DRV C:\IBMLAN\DOSLAN\LSP\DXMAOMOD.SYS 001 ~ ~
DRV C:\IBMLAN\DOSLAN\LSP\DOS\IBMTOK.DOS ~ ~ ~
DRV C:\IBMLAN\DOSLAN\LSP\PROTMAN.DOS /I: ~ ~

```

Figure 9. Standard IBM Token Ring, boot block definition file

Figure 10 shows the definition file sent to the client to force it to boot from it's own hard disk. Once a client is completely installed, it uses this configuration file, unless it is deleted, reset, or redefined.

```

; Hybrid Boot Configuration
BASE 8C0H
RPL DOS\HDBOOT.COM
LDR DOS\HDBOOT.COM

```

Figure 10. Boot from hard disk, boot block definition file

HDBOOT.COM is a small bootstrap executable that reads the first sector of the hard drive and gives control to the OS bootstrap program located on the hard disk. This boot image avoids the overhead of loading/initializing DOS, CONFIG.SYS, AUTOEXEC.BAT, and so forth, just to have the system reboot from the local hard drive.



All filenames included in a CNF file may be expressed relative to RPLDIR, which is specified in the REMOTEBOOT section of the IBMLAN.INI file on the WorkSpace On-Demand server:

```
RPLDIR = C:\IBMLAN\RPL
```

Alternatively, the paths can be fully qualified. If you provide a fully qualified path and filename, you must specify the entire path including the drive letter and all subdirectories.

All fields, including the CNF file parameters, are separated by spaces. Some values in the CNF file can contain parameter lists used by other software. Fields that contain these embedded parameter lists must be separated by tilde characters.

Table 3 describes the entries in the CNF file along with their expected file names and parameters.

*Table 3. The boot block definition file - Field descriptions*

Entry	Description
RPL	A CNF file may contain only one RPL entry. This entry specifies the first program to be run on the client. No parameters are supplied with this entry. In Workspace on Demand RPL, this is the RPLBOOT.SYS.
ORG	A CNF file may contain only one ORG entry. The entry specifies the hexadecimal segment number of a contiguous memory block on the client. Files following this entry in the CNF file are bound to this memory address. No parameters are supplied with this entry.
DAT	A CNF file may contain several DAT entries. These entries specify files that are to be stored in the boot block and passed to the client. These files are not used by RPLBOOT.SYS, but they can be read by DOS to handle I/O functions. No parameters are supplied with this entry.
LDR	A CNF file may contain only one LDR entry. This entry specifies the name of the loader to be used on the client. For WorkSpace On-Demand, this entry is BB20.US\OS2LDR ~ OS2LDR UFSD.SYS MFSD20.SYS-. In this case, the OS2LDR is initializing using the RPL micro- and mini-FSDs.
EXE	A CNF file may contain one or more EXE entries. These entries specify the names of executable programs that are to be used on the client. Parameters may be passed to an executable program as part of the EXE entry.

Entry	Description
DRV	<p>A CNF file may contain one or more DRV entries. These entries specify the names of device drivers to be used on the client. Each DRV entry requires the following parameters:</p> <p>The parameter list for the device driver. Fields within this embedded parameter list must be separated by tilde characters.</p> <p>The additional memory requirements of the device driver, if any, are expressed in decimal kilobytes.</p> <p>Use the character M if the driver can be moved after initialization. Otherwise, this parameter must be a tilde character.</p> <p>If the driver can be moved, and it requires less memory than the original driver image, RPLBOOT.SYS moves the driver to reclaim the unused memory and adjusts all interrupt vectors that point into the driver's memory area.</p>
BASE	<p>A CNF file may contain only one BASE entry. This entry specifies the hexadecimal segment number (paragraph) that is the boot block base address. The default base address is X'00C0H'.</p>

#### 2.4.1.2 The Image Files, W32DOSSB and W32DOSUB

RIPL sends DOS to the client to initiate the Windows operating system installation. The RIPL environment creates a redirected drive and causes the client to boot DOS as if it was booted from the A drive. The files required to start DOS reside in an image file on the boot server, (\BMLAN\DCDB\IMAGES). This feature includes two default image files, W32DOSSB.IMG and W32DOSUB.IMG. Using the files in the \BMLAN\DOSLAN\DOS and \BMLAN\DOSLAN\NET directories creates these images.

To create the required files, MAKEIMG.EXE uses a file that contains a list of files:

- To create W32DOSSB.IMG, MAKEIMG.EXE uses W32DOSSB.DEF.
- To create W32DOSUB.IMG, MAKEIMG.EXE uses W32DOSUB.DEF.

Both of the .DEF files reside in the \BMLAN\DCDB\IMAGES directory. To change a file that is contained in the image, change the file on the server, then re-create the file by using the MAKEIMG utility. Depending on the image that is used, change the corresponding file on the server, then re-create the image by using the MAKEIMG utility, as follows:

1. Verify that the image name is not already in use.

2. From a command prompt, type:

```
MAKEIMG def_file_name
```

3. `def_file_name` represents the definition file that is used to make the image. Do not run this command from the `\IBMLAN\DOSLAN\NET` directory on the boot server.

### ***Distinguishing W32DOSSB.IMG from W32DOSUB.IMG***

W32DOSUB.IMG uses EMM386.EXE in the CONFIG.SYS mapped by file W32\_UCFG.SYS to allow drivers to load in the upper memory blocks, which provides additional memory. W32DOSSB.IMG does not use EMM386.EXE in the CONFIG.SYS mapped by file W32\_HCFG.SYS; instead, W32DOSSB.IMG uses HIMEM.SYS. To avoid conflicts between EMM386.EXE and network cards ROM BIOS, use W32DOSSB.IMG first. If you need additional memory, and W32DOSUB.IMG loads the client installation programs without problems, you can use W32DOSUB.IMG as an alternative. If you know how to configure DOS memory managers, try different parameters with EMM386.EXE by changing W32\_UCFG.SYS and remaking W32DOSUB.IMG.

## **2.4.2 The STATE files**

As explained in the boot process, the STATE.EXE file and the STATE.FIL file on the workstation play a key role in moving the client from a pristine state to being fully installed.

STATE.EXE acts as an agent between the server and the Windows operating system installation on the client. STATE.EXE ensures that the client receives all the files necessary to install the Windows 95 or Windows 98 operating system (this includes the client application files).

STATE.EXE retrieves the client information from STATE.FIL. The following is an example STATE.FIL created for our Windows NT Workstation defined in Section 3.7, “Initial testing” on page 76.

```
[Client State]
State=NEW
OSType=WINNT
OSRelease=nt40
PartitionSize=600
InstallImage=R_DTK_ND
ClientType=Standard
```

Typically, you should not manually edit the STATE.FIL file, because the `NETWIN RIPLMACH` and `STATE.EXE` commands provide most of the values the file

contains. The only values supported through manual editing are the values for the keyword STATE. These values are NEW and Hybrid.

While preparing the system for a Windows installation, STATE.EXE moves the client through the states shown in Table 4.

Table 4. Values for the State variable in STATE.FIL

State	Description
NEW	Indicates the first stage of the client installation process. STATE.EXE creates a partition, as specified in STATE.FIL under keyword - PartitionSize.
PREP	Partition preparation stage. STATE.EXE formats the partition.
Post-Format	Format verification.
Pre-Copy	Preparation for copying files across the network.
XCOPY1	Windows files that are brought to the client.
Post-Xcopy1	Copy verification.
XCOPY2	STATE.EXE copies any other files across the network.
Post-Xcopy2	Copy verification.
INSTALL	Windows Setup process is ready to start.
HYBRID	STATE.EXE sends HDBOOT.COM to the client on all subsequent boots (this boots the client locally) until an administrator resets the client.
ERROR	An error occurred while STATE.EXE tried to prepare the system for the installation of Windows. The error number will follow the error. For more details, consult the log file on the client directory on server. Help is also available on the error number. For example, type <code>HELP STA0007</code> for help on error number seven.

#### 2.4.2.1 The STATE file (STATE.FIL)

In addition to the STATE field, STATE.FIL contains the following fields, as shown in Table 5, when it is created.

Table 5. Fields in the STATE file

Fields	Description
State	See Table 4 on page 40.

Fields	Description
OSType	Win95/Win98/WinNT image directory name. Release name could be specified.
OSRelease	Win95/Win98/WinNT.
PartitionSize	Administrators can set the partition size using the <code>NETWIN RIPLMACH /PARTITION</code> command. The default is 250 MB.
InstallImage	<code>R_DTK_NDIS</code> defines the connection to the server record string in the <code>RPL.MAP</code> file in the server.

#### 2.4.2.2 State switch daemon on server (STATEDMN.EXE)

A new server state daemon service `STATEDMN.EXE` is introduced in the Feature for Windows Clients to monitor the state of the clients boot phases and to switch from a remote boot to a local hard drive. Since Windows NT and Windows 95 cannot remote boot from Warp Server, the clients first remote boot the DOS image from the server and install a Windows operating system. Once the operating system is installed on the clients hard drive, this daemon checks the `STATE.FIL` and updates the clients entry in the `RPL.MAP` file to HYBRID image from DOSWIN RIPL image. The HYBRID boot image reboots the clients from the local hard drive.

This state daemon is required for every boot server.

The state daemon is installed as a service under the `IBMLAN\SERVICE` directory. The `IBMLAN.INI` is updated by the Feature for Windows Clients installation program.

`IBMLAN.INI` file changes:

1. Added `STATEDMN` to `SRVSERVICES` to start the daemon. This daemon should be started after the `RIPL` service is started.
2. Added the following line under the `[SERVICES]` section of the `IBMLAN.INI` file to specify the location of the state daemon exe file:

```
statedmn = services\statedmn.exe
```

3. Add the following optional section. The default delay time is 30 seconds:

```
[STATEDMN]
delay_time = 30
```

The state daemon maintains a list of all the client's states. It builds this list from the value of the state variable in the `State.fil` contained in the clients

subdirectory. The clients subdirectory is the directory named after the client under the IBMLAN\RPLUSER subdirectory. This daemon runs one round and pauses for a default 30 seconds delay time. The delay time can be set in the IBMLAN.INI file as a parameter. The parameter is DELAY\_TIME= in seconds.

#### **2.4.2.3 Log file (workstation.LOG)**

This is a progress log file that exists in the client-specific directory on the server and is written to by the STATE.EXE program from the DOS client image. This file is created and initialized when the client is created. It contains entries with event names, progress, and time stamps. The time stamp is based on the client system time.

The NETWIN RIPLMACH /RESET copies the workstation.LOG file into the workstation.OLD file and wraps the new log entry in the workstation.LOG file. Once the operating system is installed on the client, subsequent boots are from the hard drive of the client and nothing is logged on the server.

### **2.4.3 A sandbox**

A sandbox is a Windows workstation that allows for changes. Sandbox defined machines are used as part of the application package creation process as well as the designing of different desktops.

A command line switch /SANDBOX is used with the NETWIN RIPLMACH command to create a sandbox client. An entry in STATE.FIL identifies the client type. The standard client will be identified as STANDARD and the sandbox client will be as SANDBOX.

The boot logic checks the client type in STATE.FIL. If the client type is SANDBOX, it will add a command line switch in the logon client installation command. With the sandbox switch, install is executed with a unique system policy file. If the switch is not specified, standard Feature for Windows Clients install is executed on the client using a restricted system policy file.

#### **2.4.3.1 Operating system (OS) response files**

A default unattended response file is shipped for Windows 95, Windows 98, and Windows NT 4.0 Workstation. This default response file is copied into the client-specific directory at the time of machine creation. The response file is then modified based on the optional input parameters (for example, IP address). The resulting response file is unique to a target client machine.

The default response file that we ship allows the target client system to be installed with full support for devices that are autodetected and processed by

the operating system install program. These devices must have the default drivers that are shipped with the operating system.

For new OEM devices that are not part of the base operating system package, the response file has to be modified by hand by the system administrator. A sample response file and cook-book instructions are shipped with this product to give some guidance to the administrators in customizing new devices and their device drivers. The administrator is primarily pointed to Microsoft resources (for example, Windows NT Resource Kit, Microsoft on-line knowledge base on the Web, and so on) for additional information on how to customize response files. More information on this topic can be found in Section 5.4, "Machine classes and response files" on page 155.

#### **2.4.4 Additional boot process options**

While still in its installation boot phase, there is an option the administrator can use to install additional features to the clients.

By adding lines to an installation script, and placing the installable code in the proper directory structure, these features are going to be installed automatically. The features can include IBM's Java Virtual Machine (JVM) or the Tivoli Management Agent (TMA) that both ship with the Feature for Windows Clients. This can also be any Fixpack available for the operating systems.

Detailed information on how to customize these installation scripts as well as the proper positioning of the subdirectory structure can be found in Section 5.4, "Machine classes and response files" on page 155.

---

### **2.5 The user environment**

The OS/2 Warp Server user data structures have not been changed in this product. Neither the NET.ACC, or DCDB, that is used to store user data has been changed. Instead, the Feature for Windows Clients makes use of the IBM Logon Client to access additional user information and procedures to implement many of the required functions.

The Feature for Windows Clients Logon Client is based on code developed for the IBM Network Clients. The Logon Client for the Feature for Windows Clients Windows 9x and Windows NT clients has been modified to account for unattended installation, changes to the locations accessed for files, and Windows registry updates that must be executed at the Windows client platform.

The functions that are required to support the Windows client can be separated into the following three areas:

1. Desktop Profile Management

Each user has a desktop profile that defines the user's desktop information. This information is stored and handled separately. A user can have two profiles defined, one for Windows 9x and another for Windows NT 4.0.

2. Users Application Profile

Data has to be stored that describes which application has been assigned to which user. Files are used to store the application assignments.

3. Management

The information associated with the users desktop and application information has to have a management interface for it to be viewed or updated.

## 2.5.1 Building the desktop

Windows 95/98 and Windows NT platforms use a combination of Windows User Profiles and system policy files to control the desktop appearance and core systems functionality. The combination of these files deliver the restricted desktop that is part of the Feature for Windows Clients. When a user logs on his or her desktop, the appearance of the desktop and the functions are made up of a combination of the desktop assigned to the user as well as the system policy file. This is graphically described in Figure 11.

$$\begin{array}{c} \textit{user's desktop assignment} \\ + \textit{system policy file} \\ \hline = \textit{User's Desktop} \end{array}$$

Figure 11. User's desktop creation

### 2.5.1.1 The user profile

Windows NT and Windows 9x maintain a desktop and shell environment for each user. This is sometimes collectively referred to as a user profile, not to be confused with an OS/2 Warp Server User Profile. This user profile contains information about a user's environment and preferences so that each time he or she logs on a consistent interface is presented.



The user profile provides many benefits for the user. Roaming users can be presented with the same interface no matter which workstation they log on to, and users sharing the same workstation are presented with their own interface each time they log on.

Administrators can customize user profiles and assign them to users to enforce a corporate standard. They can also set up user profiles so that users cannot change them. In the Windows environment, there are three types of user profiles:

1. Local Profiles

These profiles are specific to each computer. To access a local profile, a user must log on to that specific computer.

2. Roaming Profiles

These type of profiles, as the name indicates, allows the user to access his or her profile from any computer.

3. Mandatory Profiles

These profiles are preconfigured profiles that, like roaming profiles, can be accessed from any computer with the additional restriction that these profiles cannot be changed by the user.

The Feature for Windows Clients supports roaming and mandatory profiles. In order for a user to be able to log on, the user must be assigned a mandatory profile. Each user can have two profiles, one for Windows 9x and one for Windows NT.

The profile is assigned by using the `NETWIN USER` command with the `/DESKTOP` switch. When this command is issued, a user profile is copied to the users home directory. For Windows NT, this is stored in the profiles subdirectory within the users home directory. For Windows 9x, the profile is stored in the profiles.w95 directory within the users home directory.

#### Visible files

The Windows profile subdirectories are hidden. They are in the users home directory, and to see them you need to change the attributes of the directory. For example, if you have a user Manuel whose homedirectory is at `c:\homedirs\manuel`, he would have a Windows NT Users profile stored in `c:\homedirs\manuel\profiles`.

The Windows registry is the database used to store computer-specific and user-specific settings. Portions of the file can be saved as files, called hives,

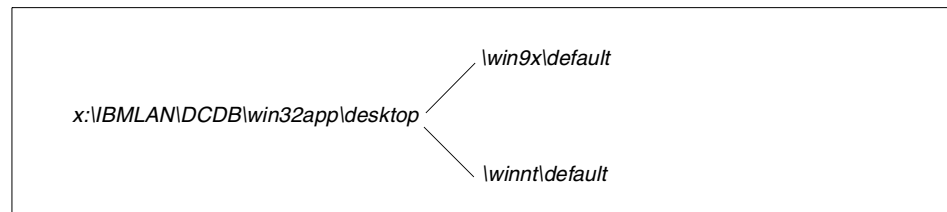
for storage. Hives can then be reloaded at a later time for usage. The first part of a user profile is a registry hive called Ntuser.dat for Windows NT that maintains the user's environment preferences and, when the user is logged on, is the HKEY\_CURRENT\_USER portion of the registry. The Ntuser.dat registry hive stores those settings that maintain network connections, Control Panel configurations unique to the user (such as the desktop color and mouse settings), and application-specific settings. During logon, this part of the subtree is merged into the local registry.

The second part of the user profile is a series of folders on the hard drive, each of which serves a purpose in the user's environment. The folder structure stores shortcut links, desktop icons, startup applications, and so forth.

In addition to the registry file, the system maintains a log file called NTUSER.MAN.LOG or USER.MAN.LOG. This file contains a list of changes that the user has made to his or her environment since he or she has logged on. When the user logs out, these changes are applied to the NTUSER.MAN or USER.MAN file. If there is a problem that prevents the changes from being applied, the log file keeps the changes until the next logon when it can be applied.

At logoff, the user portion of the registry will be deleted, and the rest of the user profile specific changes are removed. During logon by the next user at the machine, these settings will be reestablished. This is standard Windows client functionality.

Default desktops are provided. The location of the default desktops are shown in Figure 12 on page 46. An administrator can also create new desktops, put them in a different location, and assign them to users.



*Figure 12. Default desktop locations*

Examples of creating and manipulating desktops are given in Chapter 6, "User and desktop administration" on page 177.

#### **2.5.1.2 System policy files**

In addition to the user profiles described above, an administrator can also make use of system policy file to control the user environment. A system policy file can be implemented on the domain controller so that all users that log on will have these restrictions or settings applied.

Since the desktop logon and network access settings are stored, for the most part, in the computer's registry database, system policy files for users overwrite settings in the current user area of the registry, and system policy for computers overwrites the current local machine area of the registry. This allows you to control user actions (user profiles) as well as computer actions for users and groups.

The default system policy file contains a default user setting and a default computer setting. Use the System Policy Editor to manage the user desktop by changing the default user settings and manage the logon and network settings by changing the Default Computer settings.

All systems logging on to a domain with system policy files implemented will use the same uniform policy defined by the system policy file.

When a user logs on to the OS/2 Warp Server Domain, the operating system looks in the NETLOGON SHARE to see if there is an NTConfig.pol file present for Windows NT or a Config.pol file present for Windows 9x. If the file is found, the contents of the file are copied to the local computer's registry and is used to overwrite the current user and local machine portions of the registry.

With the Feature for Windows Clients, there are three System Policy files: one for administrators, one for a sandbox machine, and one for all other IBM WorkSpace On-Demand 2.0 Feature for Windows Clients. The location for each of these is shown in Figure 13.

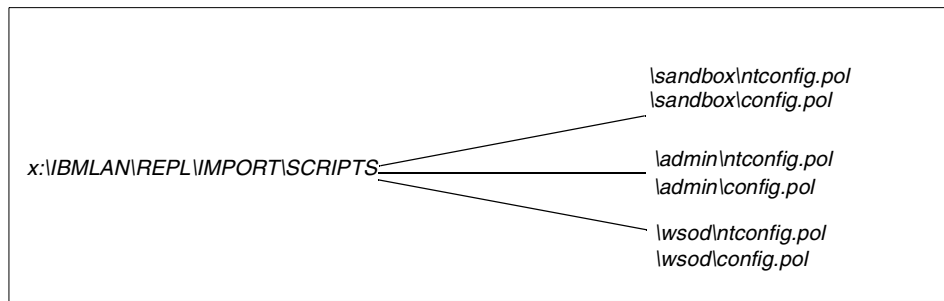


Figure 13. System policy files:location

If a user is defined as an administrator, he or she uses the system policy files in the admin directory. These are not as restrictive.

If a machine is defined as a sandbox, it will use the system policy files in the sandbox directory. It determines this by a value in the local registry. These too are not as restrictive. What happens if both apply?

#### Sandbox

When a client is defined with the `NETWIN RIPLMACH` command, and the `/SANDBOX` switch is used, a client registry value `Sandbox_enabled=1` is set. On logon, this forces the client to use the system policy file in the `/SANDBOX/` subdirectory in the NETLOGON share.

If a user is defined as anything other than an administrator, and the machine he or she logs onto is not defined as a sandbox, the user uses the system policy files in the WorkSpace On-Demand directory. These are very restrictive.

The System Policy Editor entries change local computer registry settings in the following ways:

1. Desktop settings for default user in System Policy Editor modify the `HKEY_CURRENT_USER` key in the registry that defines the contents of the user profile that is in effect for the computer.
2. Logon and network access settings for the default computer in System Policy Editor modify the `HKEY_LOCAL_MACHINE` key in the registry.

When a user logs on to the domain, the contents of the `NTConfig.pol` file on the server are merged with the `NTuser.dat` file found in the user profile location for the user logging on. Settings in `NTuser.dat` that do not match `NTConfig.pol` settings are overwritten, and thus, system policy controls the

user profile settings for the entire domain. Settings for the default computer that are not contained in the user profile are added to the local machine portion of the registry.

It is possible to build a system policy file. Using the policy editor supplied with Windows, you create a file called NTConfig.pol for Windows NT and Config.pol for Windows 9x that contains settings for users (user profiles) and computers (logons and network access settings). By saving the file to one of the locations defined in Figure 13 on page 48, you implement the policy.

This is described in greater detail in Chapter 6, “User and desktop administration” on page 177.

### 2.5.1.3 Assigning a desktop to a user

A desktop is required for a user to log on to a Feature for Windows Clients machine. Administrator authority is required to assign a desktop to a user.

This activity is done on the WorkSpace On-Demand Domain Controller where the users are defined and from which the application(s) are provided. The command takes a desktop indicated by the administrator and assigns it to the user. The desktop will be a mandatory desktop that will restrict the user from making changes to it.

1. Log on using an account with system administrator authority.
2. A default restricted desktop is shipped for each platform. The administrator can, alternatively, also use a sandbox machine to create other desktops that can be assigned to users.
3. To assign the desktop issue the command:

```
NETWIN USER userid /DESKTOP:pathname /TYPE:{WIN9X | WINNT}
```

The `userid` must be a user account name defined at the domain. `pathname` is the fully qualified path to the default desktop the administrator is assigning. `/TYPE` is the platform the administrator is providing a restricted desktop for.

4. The user account must have been previously defined in the domain and is used to get the user information.
5. The user's home directory is accessed. If the user does not have a home directory, one is created in the `PKG_DIRvalue\USERS\userid` directory using NET APIs. The User Account Subsystem account for the user is updated to hold the home directory value when the home directory is created. The Windows User Profile will be added (the `.MAN` file and the folders used by Windows) by copying it from the `pathname`, specified by the `/DESKTOP` parameter, to a location in the user's home directory. If a

Windows User Profile already exists at the location, an error will be generated and the Windows User Profile is not overwritten. The PKG\_DIR value is a fully qualified pathname that is retrieved from \IBMLAN\DCDB\WIN32APP.INI.

6. An entry is created for the user in the user datastore:

c:\ibmlan\dcdb\win32app\USRSTORE.INI

The entry will be specific to the type of desktop.

#### **2.5.1.4 The WorkSpace On-Demand desktop structure**

WorkSpace On-Demand has already established several brand characteristics, specifically:

1. The bitmap that is part of the logon panel from a Windows 95/98 and Windows NT system has been updated to be consistent with the WorkSpace On-Demand bitmap. This is part of the operating system that is installed at the client systems through the Feature for Windows Clients (boot management) support.
2. The wallpaper background to the default desktops that is shipped with the Feature for Windows Clients is consistent with the WorkSpace On-Demand standard wallpaper. This bitmap is part of the operating system that is installed at the client systems through the Feature for Windows Clients (boot management) support.

### **2.5.2 The Feature for Windows Clients Networks Logon Client**

The Feature for Windows Clients is based on code developed for the IBM Network Clients. The Logon Client for the Feature for Windows Clients Windows 9x and Windows NT clients has been modified to account for unattended installation, changes to the locations accessed for files, and Windows registry updates that must be executed at the Windows client platform.

#### **2.5.2.1 Installation**

The Logon Client is installed by default since it is required by many of the other management features of the product. This client code has been converted to InstallShield. This change allows for an unattended installation of the product.

The NETWIN RIPLMACH command that is used to define a workstation includes a switch /SANDBOX. When this switch is part of the command, a registry entry Sandbox\_enabled=1 is included in the workstation registry. If the /SANDBOX switch is excluded, the entry Sandbox\_enabled=0 is set.

This value is used during the logon process to determine which system policy file to use. For more information, see Section 2.5.1.2, “System policy files” on page 47.

#### **2.5.2.2 The user profile**

The Logon Client retrieves the Windows User Profile for each user from their home directory. This is not stored with other user information in the DCDB.

Due to the raw size of this file for each user (200 K to 500 K), if this were part of the DCDB, there would be a concern with its replication across the network. In addition, the Windows client logon support is designed (by Microsoft) to receive a user home directory location that, in turn, is used to retrieve the user desktop. This is also the default location for the storage of user information. For these reasons, the Windows User Profile is stored in the user's home directory versus being part of the DCDB.

### **2.5.3 The logon process**

To accomplish the Windows registry changes required when applications are added or removed from a user desktop, a Windows registry merge operation must be executed. This merge updates the user portion of the Windows registry. With the change, the logon flow from the Windows clients is updated.

1. The user will have a WorkSpace On-Demand logon panel displayed on the client system. The user will enter their userid, password, and domain name for the WorkSpace On-Demand server they will authenticate at.
2. The userid and password is passed to the domain for authentication. Upon successful authentication, the server is accessed for the Windows User Profile and for the applicable system policy file.
3. In the restricted user environment of WorkSpace On-Demand, the Windows User Profile is the version that has been placed in the user's home directory by the administrator. The only change that is part of this step is the actual Windows User Profile that is accessed at the server.
4. In the restricted user environment of WorkSpace On-Demand, the system policy file used is the version that the system administrator has placed in the NETLOGON share. The only change that is part of this step is the system policy file that is placed in the NETLOGON share. When the client registry value is `Sandbox_enabled=0`, the `/WSOD/` subdirectory in the NETLOGON share is accessed for the system policy file. When the client registry value is `Sandbox_enabled=1`, the `/SANDBOX/` subdirectory in the NETLOGON share is accessed for the system policy file.
5. The WorkSpace On-Demand Networks Client code will attempt to download an additional file (Windows Application Updates) from the user's

DCDB directory. This is a batch file, NTUPDATE.BAT for Windows NT and 95UPDATE.BAT for Windows 9x. If it exists, it includes a set of INF file(s) or REG files (s) and APPIDUSER.INF or APPIDUSER.REG. This information defines the set of Windows registry changes that need to be made due to the specific applications that have changed.

The Windows registry merge support is used to update the user portion of the Windows registry using the entries that have been uniquely specified for the given application. Windows 95/98 create .REG files, and Windows NT generates a file of the INF format containing the user registry entries. The user's Windows registry is updated using the following command that is issued by the Workspace On-Demand Networks Client:

```
C:\windows\RUNDLL.EXE setupx.dll,InstallHInfSection DefaultInstall 132  
FILENAME.INF
```

FILENAME.INF is the name of the file downloaded from the server. This is for Windows NT.

REG files are installed with:

```
REGEDIT /s filename
```

6. Following the update of the user's user profile with the Windows registry changes, the Windows User Profile is written back to the user's home directory and the file containing each of the Windows registry updates (Windows Application Updates) at the domain controller is deleted.
7. The changes included in the system policy file are then merged into the Windows registry at the client machine.
8. Logon processing continues with operations, such as Logon Assignment processing and Logon Script processing (as specified by the user account information in the domain).

The capabilities supported for a Windows client logon include:

- Logon assignment connections, as defined in the domain user account, are made.
- Logon scripts, as defined in the domain user account, are executed.
- PROFILE.BAT execution, as defined in the domain user account, is executed.
- Registered Network Providers are called.



Due to the fact users are enabled to roam between workstations (logon from any workstation), autologon support is not provided. Autologon support in Microsoft 95 and NT is only enabled when user profiles are not used.

#### **2.5.4 User home directory usage**

To support the Feature for Windows Clients, user-specific home directories are used. Home directories can exist on any server within the domain. The home directory contains two critical parts that have been mentioned previously:

- *Windows User Profile*: Each user will have a default Windows User Profile assigned to them by the system administrator for each desktop type they use (Windows 95/98 and Windows NT). The user profiles for each type are uniquely named and stored in unique locations. This file will range from 200 K to 500 K per user. Note: The same user profile will be used from both Windows 95 and Windows 98 client platforms.
- *Windows Application Updates*: This file is created when an application is added to a user's desktop. The file is stored in the IBMLAN\DCDB\USERS\userid subdirectory. For Windows NT, it is NTUPDATE.BAT and, for Windows 9x, it is 95UPDATE.BAT.

#### **2.5.5 User management**

Figure 14 on page 54 describes the User Application Management interfaces. Since application access is controlled by the restricted desktops used by the users, this function is also responsible for updates to the user desktop.

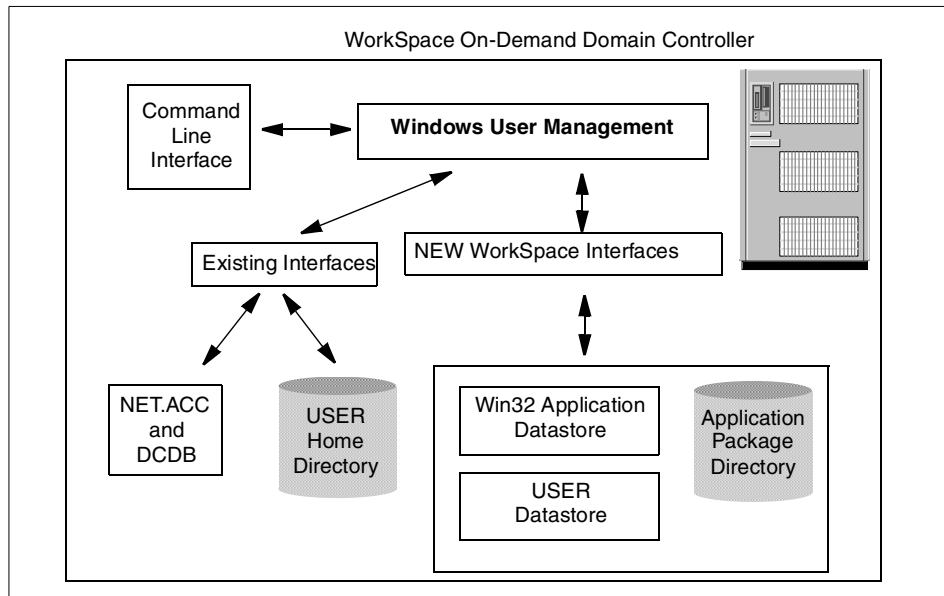


Figure 14. User application management structure

The *command line interface* is the only interface available to add, delete, or modify the attributes of a user that is related to the Windows environment. It allows the administrator to manage users desktops and applications through the command line. The *Windows User Management interface* uses the *existing interfaces* to update or get to existing structures, such as the NET.ACC, the DCDB, or the user's home directory. The Windows User Management interface uses the *new Workspace interfaces* to access data structures or locations that are specific to the IBM Workspace On-Demand 2.0 Feature for Windows Clients.

The `NETWIN USER` command is the one command available for user management. This command enables you to assign a desktop to a user account and add, delete, and list the Windows applications that a specified user account is assigned to. This command is covered in more details in Section 6.1.1, "The NETWIN USER command" on page 177.

---

## Chapter 3. Installation and planning

This chapter describes the steps required to plan and install the Feature for Windows Clients. Since every live environment promises to be different, we describe only one installation scenario.

---

### 3.1 Install environment

The IBM WorkSpace On-Demand 2.0 Feature for Windows Clients can be integrated into most computing environments. A typical environment may consist of only Windows NT servers, only OS/2 servers, or a combination of both. Each of these environments will require some consideration. These considerations are covered here.

The Feature for Windows Clients requires that WorkSpace On-Demand 2.0 is installed on any server that Feature for Windows Client is going to be installed on. This means that OS/2 Warp Server is a prerequisite. This feature has to be installed on the primary domain controller and on every boot server that manages Windows clients in the domain.

#### 3.1.1 Single-server environments

When you install the Feature for Windows Clients, there are two components that are selectable. These two components are the *application management support* and *remote boot support*.

If the installation environment has only one local OS/2 Warp Server Domain Controller, both components must be installed onto this server. If the environment has only a Windows NT server, a Warp Server Domain Controller must be introduced to support the Feature for Windows Clients.

The Windows Clients that are managed by the Warp Server machine would be authenticated by the Warp Server machine but would still be able to access resources or applications that are resident on the Windows NT machine. There are two methods to achieve a measure of integration:

1. Duplicate the user IDs on both machines and ensure that the passwords of the users are synchronized. If you intend to make use of the network neighborhood browser on the clients, install the component on the server.
2. Install OS/2 Warp Server for e-business as the primary domain controller and make the Windows NT server an additional server within this domain.

### 3.1.2 Multiple-server environments

If you have a multiple-server environment based on OS/2 Warp Server, you can select to separate the components you install.

Additional servers only have the capability to be boot servers and cannot do any application management. For this reason, when you install the software, it detects whether the server is a primary domain controller. If it is a primary domain controller, you can select to install both application management support and remote boot support.

If the machine is an additional server, you only have the option to install remote boot support. You can have multiple remote boot servers in your environment. Users and applications would need to be associated with at least one domain controller that has application management support installed.

Figure 15 on page 57 describes the services that a client can get from the various server members on the network. These services include the following:

- A client can boot from an image on either the Primary Domain Controller (PDC) or an additional boot server.
- A client needs to log on to the OS/2 Warp Server PDC for the userid to be verified and for desktop and application access.
- A client can, however, access applications from all servers on the network including Windows NT servers.

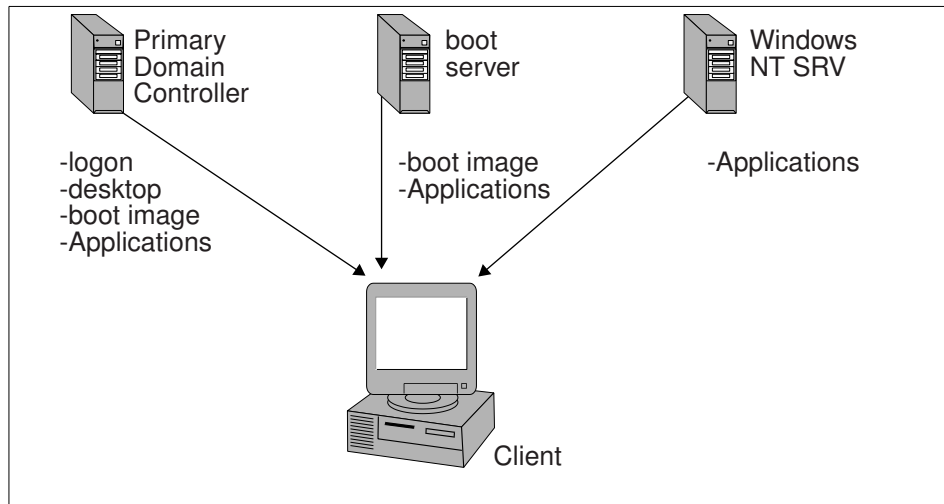


Figure 15. Accessible services

### 3.1.3 Windows NT integration

Windows NT servers can be administered separately or as part of a Warp Server Domain. This requires OS/2 Warp Server for e-business on the OS/2 Server and the appropriate software loaded on the Windows NT server. For more information, see the redbook, *Inside OS/2 Warp Server for e-business*, SG24-5393.

Irrespective of which method you choose to intergrate, all applications or services that are available on the Windows NT server will be available to the Windows Clients. For example, a client will be able to access Microsoft Exchange Servers, use file and print on the Windows NT server, and access any other available services, even though they were authenticated by the OS/2 Warp Server.

### 3.1.4 File systems

The primary domain controller must also have a partition that at least supports long filenames to support the WIN32APP directory. The WIN32APP directory is the storage location for application packages and desktop files. Application packages and desktop files require long filename support. When you install this product, you choose the location of the WIN32APP directory. The default location for the WIN32APP directory is \\BMLAN\DCDB\WIN32APP.

The location selected for the user's home directory also have to be on a drive that supports long file names. This is because the user's desktop or profile is stored in the users home directory and this needs to support long file names.

This directory also tends to get very large. Ensure that you have sufficient space on the drive for all the client operating systems as well as application packages.

When you create new application packages, you will need to create new directories on the server. If you install this product on a drive that is formatted HPFS or FAT, you need to manually propagate the access control using the `NET ACCESS` command to grant access to these new directories. Use the `/APPLY` option with the `NET ACCESS` command to propagate the access controls to all the subdirectories within the new directory.

#### **Access Controls**

Partitions formatted HPFS386 or JFS automatically inherit access controls from the parent directory. Propagation of access controls is not required on partitions formatted HPFS or JFS. You can grant access for parent directories on these partitions using the `NET ACCESS` command.

---

## **3.2 Hardware prerequisites**

The Feature for Windows Clients requires additional hardware on both the server and clients. The following section covers the minimum hardware requirements and some recommendations.

### **3.2.1 Processor**

We recommend a 90 MHZ Pentium processor or equivalent as a minimum. OS/2 Warp Server exploits the features of Pentium hardware if it is present. You should consider an Symmetric Multiprocessor (SMP) machine in cases where multi-function support is required and where supported applications are written to exploit parallel processing, such as Lotus Domino.

### **3.2.2 Disk space**

Installing all the features for WorkSpace On-Demand 2.0 requires approximately 215 MB of disk space. The Feature for Windows Clients requires an additional 20 MB disk space for all the components to be

installed. Additional disk space of about 400 MB is required for the Windows NT, Windows 95, and Windows 98 clients.

*Table 6. Disk space requirements for the Feature for Windows Clients*

Components	Disk Space Required
Install application management support (for primary domain and backup domain controllers)	12 MB
Install remote boot support (for boot server)	40 MB
<b>Total</b>	<b>52 MB</b>

The Feature for Windows Clients also needs client operating systems to be copied onto the server's hard drive. Table 7 shows disk space requirements for the individual client operating systems.

*Table 7. Disk space requirements for Window Clients Feature clients*

Component	Disk space Required
Windows NT Client Feature	125 MB
Windows 95 Client Feature	100 MB
Windows 98 Client Feature	200 MB
<b>Total</b>	<b>425 MB</b>

### 3.2.3 Memory

The WorkSpace On-Demand server memory size and hardware speed has a major influence on performance. A minimum of 64 MB of memory is recommended for optimal performance in many environments. Although slower machines provide adequate performance with small numbers of clients, a Pentium, or better, based server is recommended. An SMP machine will not increase the speed of simultaneous RIPL but may provide a performance boost during the application phase if other client/server applications are running on the server.

---

## 3.3 Software prerequisites

The Feature for Windows Clients requires that the following software be installed on the server:

1. IBM OS/2 Warp Server with the following components installed:

- File and Print Sharing
- Remote IPL Support for OS/2

OS/2 Warp Server would need to have the latest fixpacks to ensure year 2000 readiness. For the latest fixpack information, see the Web site:

<ftp://service.boulder.ibm.com/ps/products/os2/fixes/>

The supported OS/2 Warp Server environments are:

- IBM WARP Server for e-business.
- IBM WARP Server 4.0 Entry.
- IBM WARP Server 4.0 Advance.
- IBM WARP Server 4.0 SMP.

**Note**

We recommend that you use the network-optimized 32 bit High Performance File System (HPFS386) that provides enhanced file system throughput for your server, improving boot-time performance at the client.

## 2. WorkSpace On-Demand 2.0.

WorkSpace On-Demand 2.0 includes a number of fixpaks that are applied as part of WorkSpace On-Demand 2.0 installation.

- LAN Fixpack
  - IPx8405 - OS/2 Warp Version 4
  - IPx8506 - OS/2 Warp Server and OS/2 Warp Server SMP
- MPTS Fixpaks
  - WRx8503- OS/2 Warp Server SMP
  - WR0x8421 OS/2 Warp Server and OS/2 Warp Version 4

Should you be installing WorkSpace On-Demand 2.0 on OS/2 Warp Server for e-business, then the installation will not apply these service packs as they are already included in the base server package.

The Feature for Windows Clients requires DOS RIPL services on your system. If DOS RIPL services is not installed on your server, the Feature for Windows Clients installs it for you with the related DOS Year 2000 fixes.



### 3.4 Installation roadmap

The installation of the Feature for Windows Clients is multi-part. The steps are described more clearly in Table 8 on page 61.

*Table 8. Feature for Windows Clients, installation steps*

Part	Steps
Prerequisite Setup	Install OS/2 Warp Server (see Section 3.3, "Software prerequisites" on page 59). Install WorkSpace On-Demand 2.0.
Install	Install the IBM WorkSpace On-Demand 2.0 Feature for Windows Clients product. This is described in Section 3.5.1, "Attended installation" on page 62.
Client Operating System Setup	Select and set up the Windows client operating systems that will be used in your environment. Multiple selections are acceptable. The following operating systems are supported: Windows 95 Windows 98 Windows NT Workstation For specific version information on the above operating systems, see Section 3.3, "Software prerequisites" on page 59.  For details on the installation, see Section 3.5.7, "Adding the client operating system" on page 72.
Year 2000 Considerations	Each of the client operating systems supported have Year 2000 related fixpacks.  For installation details, see Section 3.6, "Year 2000 readiness" on page 73.

#### Readme

The readme in the root directory of the Feature for Windows Clients installation CD-ROM contains the latest information regarding the installation of this feature.

#### Note

We strongly recommend that you back up your server before installing this feature on an existing WorkSpace On-Demand 2.0 environment. The only way to return the server to the state it was before Feature for Windows Clients was installed is to restore the server from an existing backup.

---

### 3.5 Install and client operating system setup

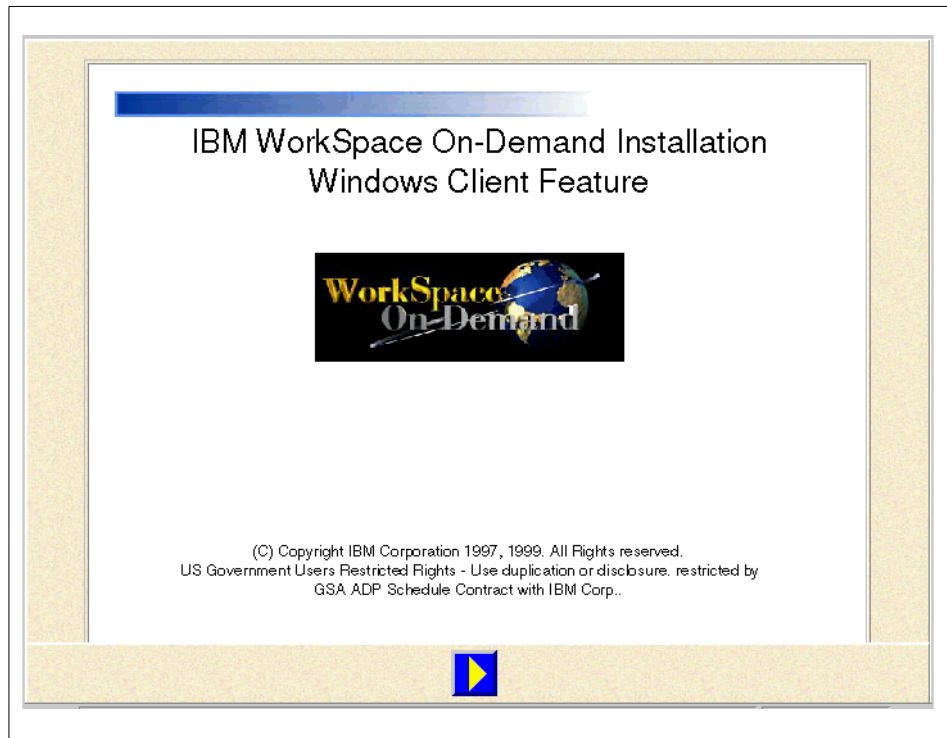
Before you begin installation please check Table 8 on page 61 to ensure that you have completed the prerequisite tasks. The Feature for Windows Clients can be installed using one of the following three installation methods:

1. An *attended installation* uses the graphical interface (GUI) and interactively prompts the user for information and options during the install.
2. A *lightly attended installation* uses the configuration, installation, and distribution (CID) method. Using this method, the user invokes the install command using a response file to provide the parameters.
3. An *unattended installation* has no requirement for an end user or administrator to be present at the system being installed. This is the most complex scenario. In this instance, the invocation of the install is handled by a Software Distribution Manager (SDM). To allow the software distribution manager to dictate when the installation should begin and what should be installed, the client system must have a distribution agent installed. This agent is started during the client system IPL. When a software distribution manager wishes to start installing a product, it will communicate with an agent that will prepare itself to execute the install process defined by the software distribution manager.

#### 3.5.1 Attended installation

Before beginning the installation, you must stop the requester and server service. It will stop automatically if you fail to do it before starting the install process. To stop all network services, type `net stop req` at an OS/2 command prompt. To install Feature for Windows Clients, complete the following steps:

1. Open an OS/2 window.
2. Insert the Feature for Windows Clients CD into the CD-ROM drive.
3. Make the current drive your CD-ROM.
4. Type `Install` and press **Enter**.



*Figure 16. Introductory panel*

This is the introductory panel. Select the **yellow arrow** to go to the next panel.

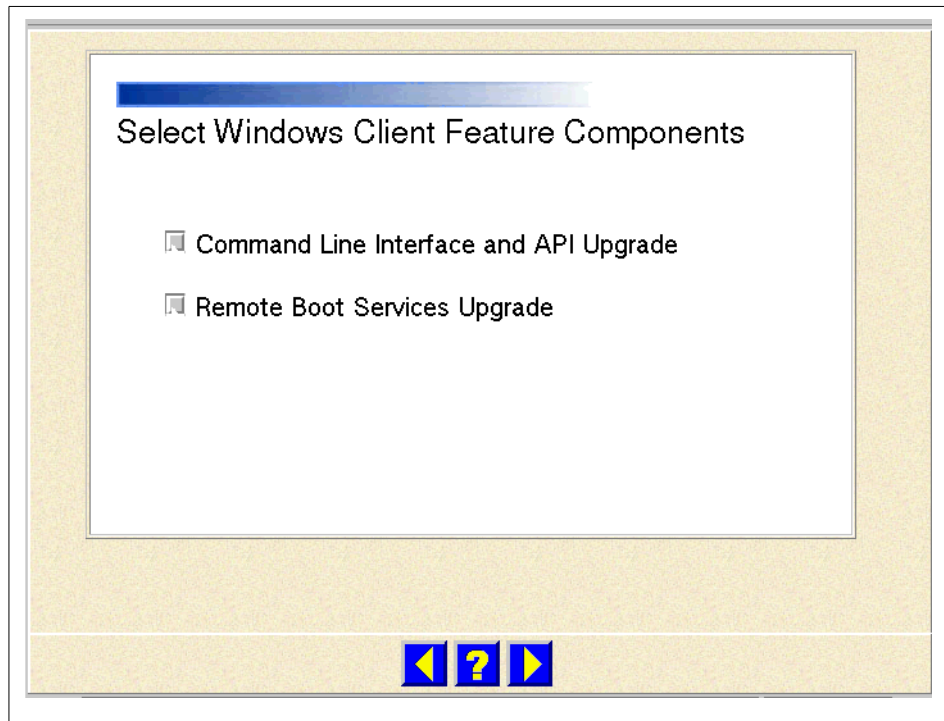


Figure 17. Feature for Windows Clients components

This panel is shown if the machine is a primary domain controller. If the machine is not a primary domain controller, only the second option is available for selection. Using the yellow arrows, you can go forward with the installation or go back to the previous install screen. The options on this panel are:

- **Command Line Interface and API Upgrades**

Installation must be performed on the primary domain controller for managing applications packages from the command line interface. Select this option to upgrade the command line interface to be able to support the Feature for Windows Clients commands and functions.

- **Remote Boot Services Upgrade**

Select this option to upgrade your remote boot services. Select the components that you need and click on the **forward arrow** to continue with the installation.

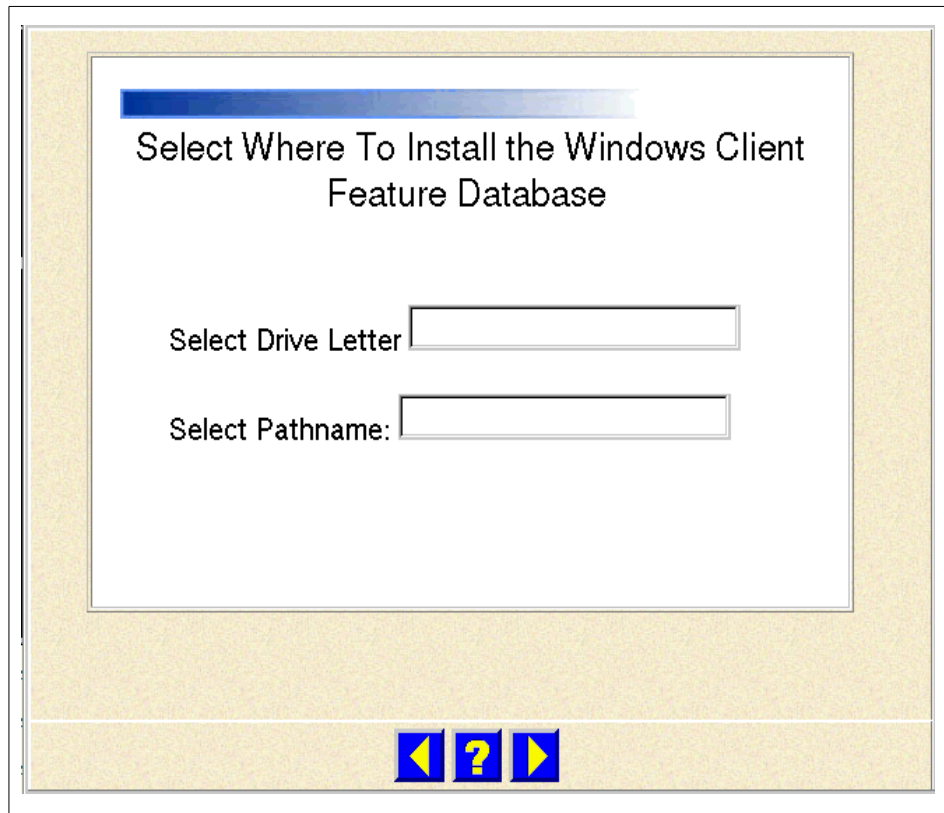


Figure 18. Installation parameters

This panel allows you to select the drive and the path to install the Windows application data repository. The default path is \IBM LAN\DCDB. These parameters are stored in the \IBM LAN\DCDB\WIN32APP.INI file.

**Note**

The above panel will not be shown if you reinstall this product. If you reinstall, and you want to change the default path, delete the win32app.ini file in the X:\IBM LAN\DCDB subdirectory, and the panel will reappear when you reinstall.

This repository can get quite large, so it is imperative that you select a drive with sufficient space for all the Windows application data. The amount of

space on the drive should be at least equal to the sum of the space required by all the Windows applications.

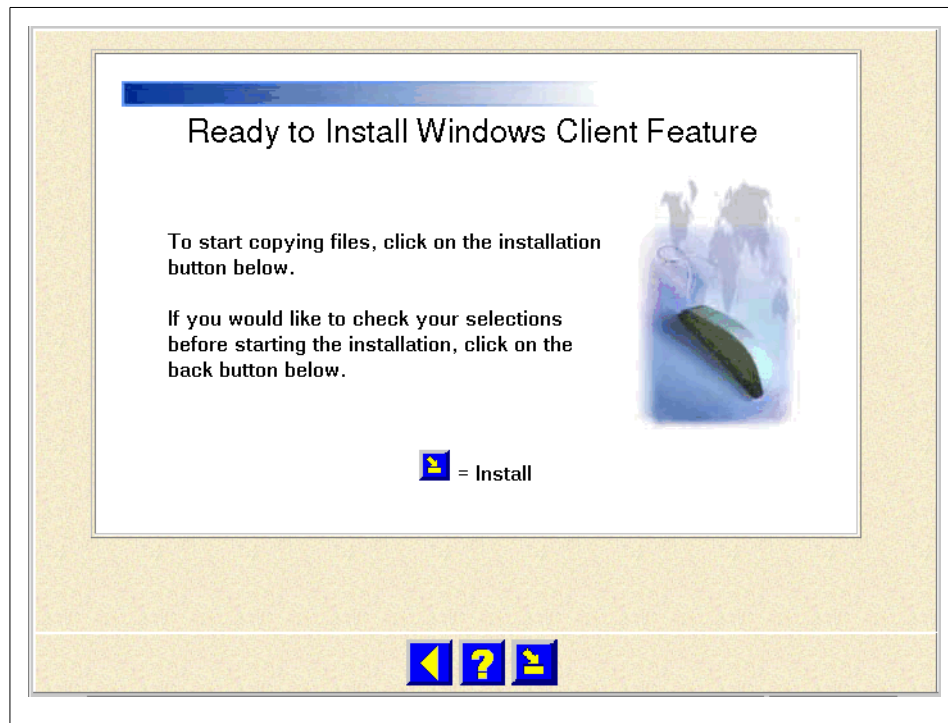


Figure 19. Ready to install panel

This is the last panel presented before the files are copied across to the server. Remember, you cannot uninstall this product. You will need a backup to restore the server to its current state.

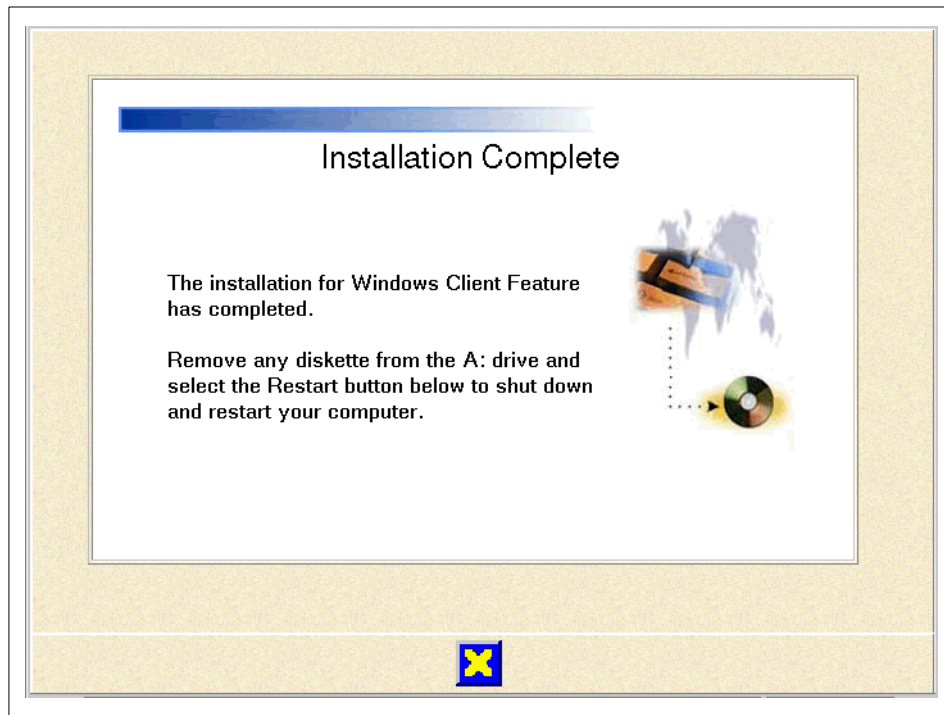


Figure 20. Installation complete panel

When the installation is complete, remove any diskettes in your drive and reboot the machine. If you encounter any problems during the installation, check the following log files C:\OS2\INSTALL\BB20ERR.LOG, C:\OS2\INSTALL\BB20HST.LOG, and C:\OS2\INSTALL\WPINSTALL.LOG. These files will help you to determine what caused the installation failure. Correct the problem and rerun the `install` command. The operating system is able to determine where it failed and will resume the installation from where it left off.

### 3.5.2 Lightly attended installation (CID)

You can install the Feature for Windows Clients in an unattended mode using a response file to provide the necessary installation parameters. The Feature for Windows Clients contains a sample response file BBWFCID.RSP. This file resides in the install directory on the Feature for Windows Clients CD. You can copy this response file to your X:\ drive and modify it to meet your requirements (X: represents a read drive other than your CD-ROM drive).

The default response file bbwfcid.rsp:

```
***** **
Workspace On-Demand 2.0 Feature for Windows Clients
* Sample Partial Response file *
*****
*-----*
* To install any Components for the Feature for Windows Clients, the *
* following selection must be set to 1. *
* ----- *
WCF_Win32.Selection=1
WCF_netwin.Selection=All
WCF_MachineManagement.Selection=All
* ----- *
* Client Management and Application Management Support for primary *
* domain controller *
* Valid Values: *
* None = Do not install Client Management and Application Management *
* Support on the primary domain controller *
* All = Install Client Management and Application Management Support** on the
primary domain controller (default) *
* Please note: All selections must be either set to ©All© or to ©None© *
* -----
*WCF_DCExternalPackages.Selection=All
WCF_DCMachineManagement.Selection=All
WCF_AppManagement.Selection=All*
----- *
* Remote Boot Support *
* Valid Values: *
* None = Do not install Remote Boot Support
* All = Install Remote Boot Support (default) *
* Please note: All selections must be either set to ©All© or to ©None© *
-----
*WCF_LogonClient.Selection=All
WCF_DSExternalPackages.Selection=All
WCF_DSMachineManagement.Selection=All
```

On a drive other than the CD-ROM drive, type `MD install` to create a new directory and save the updated response file to the new directory (install) you created.

Type `E:` (E: represents the CD-ROM drive) to assign the CD-ROM as the current drive.

Type `INSTALL` and the path to the working copy of the response file, as well as all other required parameters.

Using on line, type `INSTALL /R:X\install\bbwfcid.rsp`  
`/L1:os2\install\bb20err.log /L2:os2\install\bb20hst.log.`

The feature install should run and then restart the system.

### 3.5.2.1 Additional parameters for lightly attended installation

The following are additional parameters for unattended install:

**/B:BootDrive**



The boot drive of the server or the system administration client. This drive letter may be different from the drive letter that was used to boot the system if you are installing IBM WorkSpace On-Demand Feature from a maintenance partition. The default is the drive from which the system was booted. This is an optional parameter.

#### **/HELP**

Provides task-oriented syntax and help for this command. This is an optional parameter.

#### **/L1:ErrorLogFilePath**

The fully qualified path and file name of the error log file. The default is X:\OS2\INSTALL\BB20ERR.LOG (X represents the boot drive).

#### **/L2:HistoryLogFilePath**

The fully qualified path and file name of the historylog file. The default is X:\OS2\INSTALL\BB20HST.LOG (X represents the boot drive)

#### **/R:ResponseFilename**

The file name of the feature install response file /R:BBWFCID.RSP.

#### **Note**

If you are installing this product on a domain controller and boot server using a response file, the remote boot values in the response file must be set to `all`. If you are installing this feature on the primary domain controller, the remote boot value in the response file must be set to `none`. See "The default response file bbwfcid.rsp" on the previous page.

### **3.5.3 Unattended installation**

You can install the Feature for Windows Clients using an unattended installation method. This will require the use of a Software Distribution Manager. For more information, see the redbook, *OS/2 Warp CID Software Distribution Guide*, SG24-2010.

### **3.5.4 Reinstalling the Feature for Windows Clients**

You can reinstall the IBM WorkSpace On-Demand 2.0 Feature for Windows Clients. If you reinstall this feature, you do not get to determine the location of

the WIN32APP subdirectory, unless you delete the win32app.ini file in the X:\IBMLAN\DCDB subdirectory. You should also consider the following.

#### 3.5.4.1 Updating RPL.MAP

Typically, when you reinstall the IBM WorkSpace On-Demand 2.0 Feature for Windows Clients, the reinstallation does not change the RPL.MAP file. There are multiple methods to add server records to RPL.MAP. They are:

1. Manually remove the existing server records before you reinstall.

Manually remove the existing server records and then reinstall this feature on the server. The installation process then updates most of the server records in RPL.MAP. You must remove the existing server records. It is not sufficient to surround these records with comment symbols. Then run GETRPL.EXE after you reinstall the feature. See the following lists of CNF files to determine which files to delete before you reinstall this feature.

2. Manually update server records without reinstalling the feature and removing the existing records.

You can manually update some server records without deleting the records before you reinstall this feature. Use the commands that are provided on this feature's CD to update these server records manually. Run GETRPL.EXE after you update the server records. See the following lists of CNF files to determine the commands that manually update RPL.MAP.

The following is a list of the CNF files that are created as server records when you install this feature. The installation process only adds these files the first time you install this feature. To update these files in RPL.MAP, you can either delete these files before you reinstall this feature, or you can run a command. CNF files:

dosn3fe.cnf, dosncla.cnf, dosne10.cnf, dosnej.cnf, dosnfe.cnf, dosnlsf.cnf, dosnmdg.cnf, dosnmfp.cnf, and dosnsmc.cnf

To manually update the RPL.MAP for these files, run the following command from the INSTALL directory of this feature's CD:

```
UPDRPLMP /F:CNF1.DAT /I:<LanDrive> /P:<RIPLPath> /O /B:<BootDrive>
```

<LanDrive> represents the drive that LAN Services is installed on.

<RIPLPath> represents the drive and directory where RPL.MAP resides.

<BootDrive> represents the drive used to boot the system with LAN Services.

The following CNF files are added to RPL.MAP if your server did not have DOS RIPL support installed before you installed the Feature for Windows Clients. These files are never reinstalled; you must copy them manually if you want to replace them. You must also update RPL.MAP server records manually. CNF files:

dosbbet.cnf, dosbbpc.cnf, dosbbtr.cnf, dosn164.cnf, dosn316.cnf, dosn3e3.cnf, dosnaae.cnf, dosnd3ei.cnf, dosnd3em.cnf, dosndet.cnf, dosndtr.cnf, dosnlae.cnf, dosnls.cnf, dosnps2.cnf, and dosnwdp.cnf

To manually update the RPL.MAP for these files, run the following command from the INSTALL directory of this feature's CD:

```
UPDRPLMP /F:CNF2.DAT /I:<LanDrive> /P:<RIPLPath> /O /B:<BootDrive>
```

<LanDrive> represents the drive that LAN Services is installed on.

<RIPLPath> represents the drive and directory where RPL.MAP resides.

<BootDrive> represents the drive used to boot the system with LAN Services.

### 3.5.5 Post-installation procedures

After the installation is finished, the machine must reboot. In a CID installation, this happens automatically since the installation program return code is xFEOO indicating a successful program termination with a required reboot. After installation, you need to run the `GETRPL` command.

#### Running the GETRPL command:

Before running `GETRPL`, you must ensure that:

1. The server services is started.
2. The RIPL services is *not* started.
3. You are logged on to the server with a user ID that has administrator authority. You can determine the services that are started in your server by typing the `NET START` command (without specifying a service name) from a command line.

If you run `GETRPL` without meeting the above conditions, the utility will fail to run.

To run the `GETRPL`, enter the following from an OS/2 window or full-screen command prompt:

```
GETRPL
```

and press **Enter**.

For more information about `GETRPL` and its optional parameter, see the online Workspace On-Demand Administrator's Guide that is provided on the WorkSpace On-Demand 2.0 CD.

**The `GETRPL` command provides these functions:**

- Migrates the RPL.MAP workstation and server records from previous levels of Warp Server into the RPL.MAP on the current remote IPL server.
- Creates a group, RPLGROUP, that all IPL workstations will belong to.
- Moves DOS remote IPL users from the previous level of Warp Server into a group called RPLGROUP and creates an access control profile for RPLGROUP granting all privileges to the users in that group.
- Adds new remote IPL users to the RPLGROUP that is created with OS/2 Warp Server.
- Updates the operating system remote IPL default directories.
- Creates the access control profiles for the \IBMLAN\RPL and \IBMLAN\RPLUSER.
- Updates the RPL.MAP file to enable server records that correspond to a LAN adapter type found in the server.

### 3.5.6 Uninstalling the Feature for Windows Clients

There is no uninstall for the Feature for Windows Clients. If you wish to remove this feature, you can do this manually. You should back up your WorkSpace On-Demand server before installing this product.

### 3.5.7 Adding the client operating system

This product provides the utilities to copy each operating system from a Windows CD to a client image directory on the boot server. Table 9 on page 72 shows the commands you run on the command line to copy the client operating system to the boot server

*Table 9. Utilities to install Windows operating systems on the server*

Operating system	Type the following command
Windows NT 4.0	WCF_NT E C
Windows 95	WCF_95R2 E C
Windows 98	WCF_98 E C

E represents the CD-ROM drive and c represents the boot drive.

After the operating systems are copied to the client image directory on the boot server, you will need to install the service packs as described in Section 3.6, "Year 2000 readiness" on page 73.

---

### **3.6 Year 2000 readiness**

Year 2000 readiness is a natural prerequisite at this time and even in the future. It would be best to check the referenced or related Web sites for updated information before preparing to install.

#### **3.6.1 OS/2 Warp Server and WorkSpace On-Demand 2.0**

WorkSpace On-Demand 2.0 is year 2000 ready. For OS/2 Warp Server, you will need at least fixpack 32 for OS/2 Warp 3.0 and LAN fixes IPX8266 (for Warp Server and Warp Server Advanced) or IPX8504 (for Warp Server Advanced - SMP). These should be installed prior to installing WorkSpace On-Demand 2.0.

#### **3.6.2 Windows client operating system**

Microsoft has a Year 2000 Readiness Disclosure & Resource Center located at:

<http://www.microsoft.com/year2000/>

This site contains up-to-date information on the Microsoft client operating systems.

##### **3.6.2.1 Windows NT 4.0 and Y2K**

Microsoft provided the following information at a time close to publication on their Web site:

*While Microsoft continues to recommend that customers install the most current Service Pack/Release for non-Year 2000 reasons, we understand that, for many reasons, this may not be possible. In order to aid our customers Year 2000 efforts, Microsoft intends to maintain Windows NT Workstation 4.0 Service Pack 4 as compliant through January 1, 2001. Newer Service Packs are also to be maintained as compliant, and may include additional non-Year 2000 updates. This is intended to minimize the Year 2000 as a reason to upgrade.*

*Microsoft Windows NT 4.0 is not Year 2000 ready. You need to apply service pack 4 to the server client directory. Microsoft recommends Service Pack 4*

since there are some Year 2000 complaint issues with Service Pack 3. For more information check: <http://www.microsoft.com/technet/year2k/product/>

This section describes the installation of Windows NT 4.0 Service Pack 4. Post-Service Pack 4 Year 2000 updates are available as separate downloads for Service Pack 4 users and are included in Service Pack 5. They are available for download at:

<http://www.microsoft.com/ntserver/nts/downloads/recommended/nt4y2kpostsp4/default.asp>

To add Service Pack 4 to the client operating system image:

1. Log on to the boot server.
2. From the command line, change directory to  
\\BMLAN\rpl\nt40\i386\%\$OEM\$ and make a subdirectory called SP4.
3. Copy Sp4i386.exe to \\BMLAN\rpl\nt40\i386\%\$OEM\$\SP4.
4. Open the \\BMLAN\rpl\nt40\i386\%\$OEM\$\CMDLINES.TXT file by any editor.
5. At the end of the file, type: ".SP4\SP4i386.EXE -Z -U"

#### Note

When you install Windows NT 4.0 Service Pack 4 using a Cmdlines.txt file and Update.exe, you may receive file copy error messages for the following files: Appserver.class, Context.class, lmtxas.class, Objectcontent.class, and Securityproperty.class. Only files with 8.3 file names can be copied using a Cmdlines.txt file and Update.exe. To work around this problem, use the compressed Sp4i386.exe file from the Service Pack 4 CD-ROM, or SP4I386.EXE from the standard download from the Microsoft Web site.

#### 3.6.2.2 Windows 95 and Y2K

Microsoft provided the following information at a time close to publication on their Web site:

*While Microsoft continues to recommend that customers install the most current Service Pack/Release for non-Year 2000 reasons, we understand that, for many reasons, this may not be possible. In order to aid our customers' Year 2000 efforts, Microsoft intends to maintain Windows 95 version OSR 2, 4.00.1111 as compliant through January 1, 2001. Newer Service Packs are also to be maintained as compliant, and may include additional non-Year 2000 updates. This is intended to minimize the Year 2000 as a reason to upgrade. Microsoft Windows 95 is Year 2000 compliant with*

*minor issues. For more information and where you can download the service pack check: <http://www.microsoft.com/technet/year2k/product/>*

After you download the Year 2000 Update for Windows 95, a compressed executable file named W95y2k.exe appears on your hard drive. This executable file contains the files you need to update your Windows 95 system for the year 2000.

To add the service pack to the client operating system image:

1. Log on to the boot server.
2. From the command line, change directory \IBMLAN\RPL\W95\C\ and make a directory called \$Win95y2k.
3. Copy the service pack to IBMLAN\RPL\W95\C\WIN95Y2k.
4. Open the IBMLAN\RPL\W95\RSP\MSBATCH.INF using any editor.
5. Under the [LOGONCLT.REG] section, add:

```
HKLM, "Software\Microsoft\Windows\CurrentVersion\RunOnce", Y2K, "C:\$win95y2k\win95y2k.exe"
```

### **3.6.2.3 Windows 98 and Y2K**

Microsoft provided the following information at a time close to publication on their Web site:

*While Microsoft continues to recommend that customers install the most current Service Pack/Release for non-Year 2000 reasons, we understand that, for many reasons, this may not be possible. In order to aid our customers' Year 2000 efforts, Microsoft intends to maintain Windows 98 as compliant through January 1, 2001. Newer Service Packs are also to be maintained as compliant, and may include additional non-Year 2000 updates. This is intended to minimize the Year 2000 as a reason to upgrade.*

After you download the Year 2000 Update for Windows 98, a compressed executable file named W98y2k.exe appears on your hard drive. This executable file contains the files you need to update your Windows 98 system for the year 2000.

To add the service pack to the client operating system:

1. Log on to the boot server.
2. From the command line change directory to \IBMLAN\RPL\W98\C\ and make a directory called \$Win98y2k.
3. Copy the service pack to IBMLAN\RPL\W98\C\WIN98Y2k.

4. Open the IBMLAN\RPL\W98\RSP\MSBATCH.INF using any editor.
5. Under the [LOGONCLT.REG] section add:
 

```
HKLM, "Software\Microsoft\Windows\CurrentVersion\RunOnce", Y2K, , "C:\$win98y2k\win98y2k.exe"
```

---

### 3.7 Initial testing

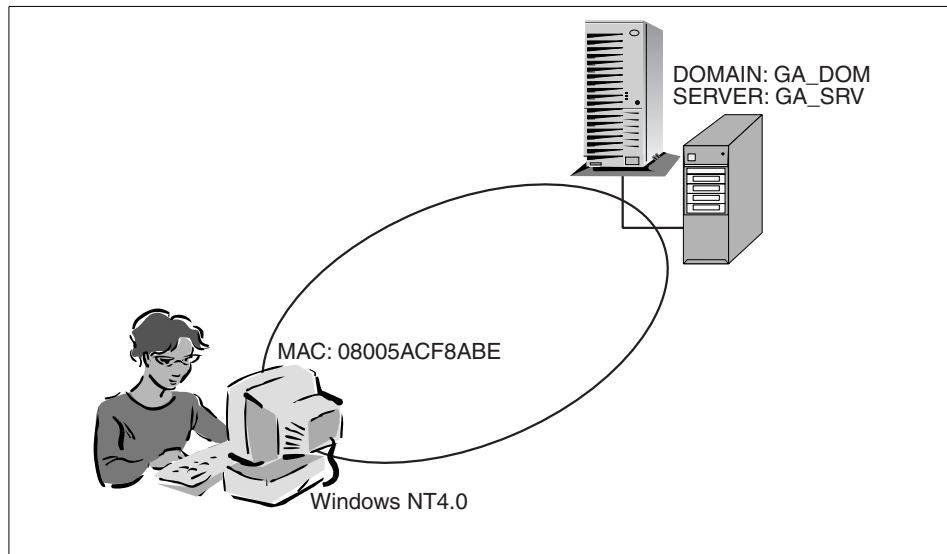
Once all the steps laid out in Table 8 on page 61 have been completed, the machine can be defined and set up. More information is available on defining clients in Chapter 5, "Workstation customization and administration" on page 131. To test the installation, define a single client by completing the following steps:

1. Log on to the boot server as an administrator.
2. Open an OS/2 Window and type: `NET START RPL`
3. From the command prompt, type:

```
NETWIN RIPLMACH machiname /ADD /MAC:08005ACF8ABE /RELEASE:NT40
/TYPE:WINNT /DHCP:N /ENABLED:Y /IMAGE:W32DOSSB.IMG /RECORDID:Z R_DTKTRK
/PARTITION: 500 /CDKEY:XXX-XXXXX
```

<b>/ADD</b>	Indicates that a client is to be added.
<b>/CDKEY</b> {:cdkey}	Specifies the key of the product installed.
<b>/MAC</b> {:AdapterAddress}.	
<b>/RECORDID</b> {:ServerRecordID}	Specifies the server record that services the client's RIPL request when the client is started.
<b>/IMAGE</b> {:ImageFileName}	Specifies the name of the image definition. The default is W32DOSSB.IMG.





*Figure 21. The lab environment*

In our lab environment, we have a number of workstations attached by token ring to our server, GA\_SRV. The domain is called GA\_DOM. The workstation we use is an IBM PC 350 containing a SVGA Video card and an IBM T-R Shared RAM Family Token-Ring network card with the MAC address shown in Figure 21. Above, we define it as a generic machine and do not use a custom response file.

#### Note

After the client is installed, a listing of the  
\\IBMLAN\RPLUSER\workstation" is as follows:

```
7-01-99  3:50p    <DIR>          0  .
7-08-99  11:09a    <DIR>          0  ..
          49      0  CLIENT.INI
7-01-99  3:48p    4042          0  CP.RES
7-01-99  3:45p    102          0  DOFORMAT.BAT
7-01-99  3:48p     69          0  LOGONCP.INI
7-01-99  3:46p    772          0  MAPLCP.BAT
7-01-99  3:48p    354          0  MAPLCP.INI
7-01-99  3:46p     53          0  MEM.INI
7-01-99  3:45p   1526          0  msbatch.inf
          44      0  NETWORK.INI
7-01-99  3:46p     65          0  PRECOPY.BAT
7-01-99  3:46p     70          0  RUN.INI
7-01-99  3:48p    168          0  State.fil
2-18-99  2:41p     28          0  verlevel.txt
7-01-99  3:48p   2228          0  WCF95.LOG
          16 file(s)    9570 bytes used
```

The directory appears to be corrupt. This is not a defect. It is caused by the manner in which the DOS client used in the initial boot writes to the server.

## 3.8 Tivoli Management Agent (TMA) and Java Virtual Machine (JVM)

The IBM Java Virtual Machine (JVM) and the Tivoli Management Agent (TMA) are included as part of WorkSpace On-Demand 2.0 feature. JVM is installed on clients through commands in the default response files for Windows 95 and Windows 98 or CMDLINES.TXT for Windows NT. If you want to use TME 10 products on your server to provide additional client management capability, modify the default response files. For more information on Tivoli, see Chapter 7, "The Tivoli TMA client" on page 205.

### 3.8.1 Installing the TMA

By default, the TMA software is not installed on clients. If you want to use TME 10 products, you need to modify the default response file and uncomment a line on the MSBATCH.INF file on Windows 9X clients or the CMDLINES.TXT on Windows NT clients and the TMA will be installed.

**To install TMA on Windows 95 clients:**

1. Change to the X:\IBMLAN\RPL\W95\OS directory. (X represents the drive letter on the boot server where you installed your Windows 95 operating system.)

2. Open the file MSBATCH.INF

At the bottom of the file, uncomment (remove the ; from) the following line under the section [LOGONCLT.REG]:

```
HKLM, "Software\Microsoft\Windows\CurrentVersion\RunOnce", TMA, , "c:\$TIVOLI\TIVOLI.bat"
```

#### **To install TMA on Windows 98 clients:**

1. Change to the X:\IBMLAN\RPL\W98\OS directory. (X represents the drive letter on the boot server where you installed your Windows 98 operating system.)

2. Open the file MSBATCH.INF

At the bottom of the file, uncomment (remove the ; from) the following line under the section [LOGONCLT.REG]:

```
HKLM, "Software\Microsoft\Windows\CurrentVersion\RunOnce", TMA, , "c:\$TIVOLI\TIVOLI.bat"
```

#### **To install TMA on Windows NT clients:**

1. At an OS/2 command prompt, change to the X:\IBMLAN\RPL\NT40\I386\\$OEM\$ directory. (X represents the drive letter on the boot server where you installed your Windows NT operating system.)

2. Open the file CMDLINES.TXT.

3. At the bottom of the file, uncomment (remove the ; from) the following line under the section [Commands]:

```
"C:\$TIVOLI\SETTIVOLI.CMD"
```

### **3.8.2 To exclude the Java Virtual Machine (JVM)**

The JVM is installed to the client operating system by default. You can change your response file to remove this step from the installation. The steps are described below.

#### **To avoid installing the JVM on Windows 95 clients:**

1. Change to the X:\IBMLAN\RPL\W95\OS directory. (X represents the drive letter on the boot server where you installed your Windows 95 operating system.)

2. Open the file MSBATCH.INF

At the bottom of the file, comment (add the; from) the following line under the section [LOGONCLT.REG]:

```
;HKLM, "Software\Microsoft\Windows\CurrentVersion\RunOnce", JVM
, , "c:\$IBMJVM\IBMJVM.bat"
```

**To avoid installing the JVM on Windows 98 clients:**

1. Change to the X:\IBMLAN\RPL\W98\OS directory. (X represents the drive letter on the boot server where you installed your Windows 98 operating system.)
2. Open the file MSBATCH.INF
3. At the bottom of the file, comment (add the; from) the following line under the section [LOGONCLT.REG]:

```
HKLM, "Software\Microsoft\Windows\CurrentVersion\RunOnce", JVM, , "c:\$IBMJ
VM\IBMJVM.bat"
```

**To avoid installing the JVM on Windows NT clients:**

1. At an OS/2 command prompt, change to the X:\IBMLAN\RPL\NT40\I386\\$OEM\$ directory. (X represents the drive letter on the boot server where you installed your Windows NT operating system.)
2. Open the file CMDLINES.TXT
3. At the bottom of the file, comment (add the; from) the following line under the section [Commands]:

```
; "C:\$IBMJVM\SETJVM.cmd"
```

---

### 3.9 Client performance

Performance is a vast topic. This section will serve only as an introduction. Because of the boot and startup requirements, OS/2 type clients for WorkSpace On-Demand have different system requirements for the Feature for Windows Clients.

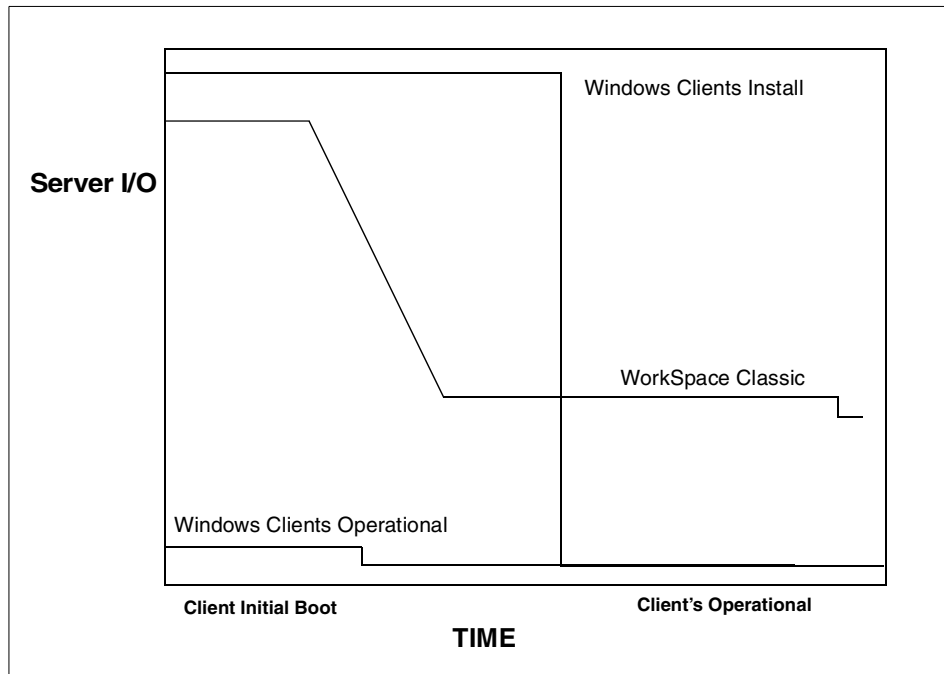


Figure 22. Server I/O and client requirements

Figure 22 shows the different client requirements. The curve for the Classic WorkSpace On-Demand client is high when the client boots. It drops and flattens out when all the clients are up and running. It will not drop lower since the clients all still have a dependency on the server for all of their files.

The Windows clients have two curves. The install curve represents the clients during their install phase. Initially, the clients copy their files from the server, which requires a lot of I/O. Once the files are copied from the server, the I/O drops. At this point, the only dependency the Windows client has on the server is for the boot image that forces the client to boot from the local hard disk. The size of this image is within a megabyte. Each client that is installed is using the same image, so this server cache does not need to be refreshed.

The operational curve maintains this low level. Again, at boot time, the client requests only the small boot image that forces it to boot from the local drive.

More information on performance is available at:

[http://www.software.ibm.com/network/workspace/library/whitepapers/wod\\_capacity.html](http://www.software.ibm.com/network/workspace/library/whitepapers/wod_capacity.html)

Please consider the factors mentioned above when reading the white paper.

### **3.9.1 File system considerations**

During the install phase, a high number of files are copied from the server. For this reason, the efficiency of the file system is the most significant performance factor. If you have more than a few clients, HPFS386 would provide the best performance. JFS on OS/2 Warp Server for e-business does not provide the same level of performance. For more information on comparing the file systems, refer to the redbook, *Inside OS/2 Warp Server for e-business*, SG24-5363.

---

## **Chapter 4. About application creation and management**

Setting up application software is a complex task. This chapter describes the steps required to plan and install the application packages.

There are two types of application packages: Windows 9x application packages and Windows NT application packages. Although Windows 98 applications should run on Windows 95, there is no guarantee that this will work. Therefore, it is best to create application packages for each platform.

The chapter begins with a description on how application processing is organized, followed by a brief high level overview on how application details are captured, defined, and distributed.

The rest of the chapter describes the process in greater depth using the StarOffice application suite as an example.

---

### **4.1 Application processing**

Currently OS/2 WARP Server supports DOS, Windows 3.x, and OS/2-based applications. Now, support has been added to support Windows 9x and Windows NT applications as well. However, again due to the nature of the Windows platform, the manner in which applications are made available to users is quite different. This section describes the architecture of the processes that are used to integrate and manage application software.

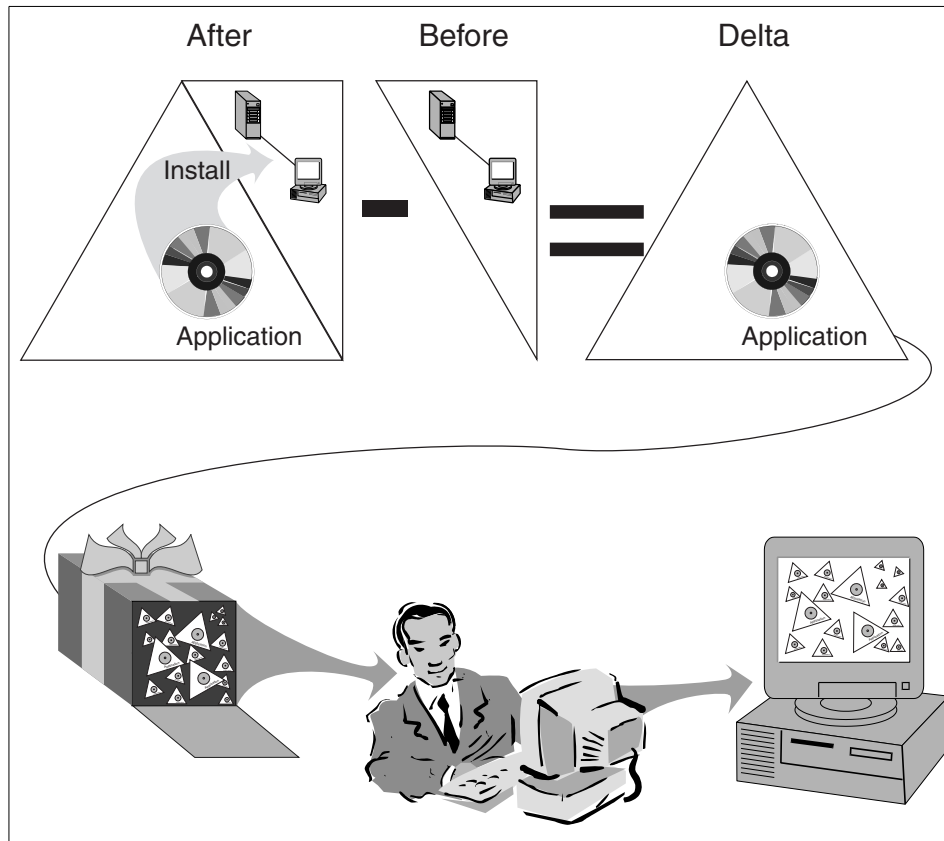


Figure 23. Application process overview

Applications are usually designed to be installed at a server or workstation and, in some cases, a part on each platform. For applications that are distributed in large distributed environments, this process is not practical. Figure 23 is a high level view on how applications information is collected and distributed. The process is explained greater detail in the sections that follow.

In Figure 23, an application is installed into a *sterile* or *sandbox* environment. After the application is installed, the delta is captured. Multiple applications can be installed and a single delta captured for all the applications, or you can create multiple deltas by treating each process separately. A single delta, or several deltas, can be combined to create an application package. These packages can be assigned to a user (defined in the domain) of a Windows workstation. When the package is assigned to a user, the user's desktop is updated to contain the new applications. Depending on the application, files on the workstation may also need to be updated.



Application management within Feature for Windows Clients can be broadly broken down into two general areas:

1. Application Management:

The Windows application management feature provides the ability for a system administrator to manage Windows applications to a degree that is based on the characteristics of the application. Unlike OS/2 applications, some Windows applications have to be, like the operating software, installed at the workstation.

For applications that have been designed to be server-based, the applications can be installed to a WorkSpace On-Demand server and delivered from the server without changes being required at the client machines. For applications that have not been designed to be server-based, the applications must be installed on the client systems from which users wish to execute them.

2. User Access to Applications:

The Windows application management feature provides the ability for a system administrator to completely control what applications a given user has the ability to access/use. Independent of whether the application resides fully on the server, partially on the client and partially on the server, or fully on the client, the system administrator has the ability to manage user access to it.

#### **4.1.1 Overview: Capturing and creating an application**

In order to gain a better understanding of a process that can be long and complicated, we first provide an overview of all the steps that are required to capture and create a Windows application package for distribution. This is described in greater detail later in this chapter.

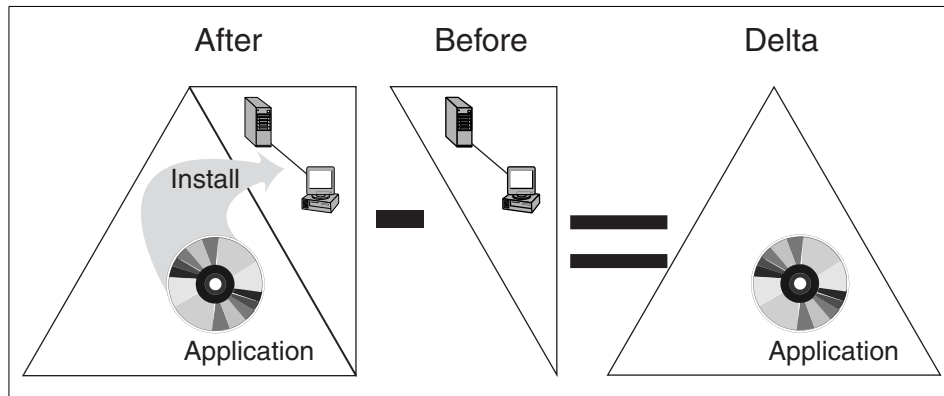


Figure 24. Creating an application package

To create an application package for distribution, an administrator needs to install the application onto a machine. The changes the application introduces are captured and packaged. There are a number of changes an application may introduce. Each of the changes are stored in a different place on the server. To create an application package, you need to complete the following steps:

1. Define a sandbox machine.
2. Logon to the boot server and create a user account that does not have any applications assigned.
3. Create the application directory on the server and define a share.
4. Create the application package by:
  - Logging on to the sandbox with the username defined in 1.
  - Checking the environment and starting the `CRTPKG` utility.
5. Install the application on the sandbox machine.
6. Restart the sandbox and customize the applications.
7. Run the `CRTPKG /FINISH` command.
8. Define the application package on the server.
9. Distribute the application files to the clients.
10. Assign the applications to users.

#### 4.1.2 Overview: Distributing applications

Once a package has been created it is available for distribution.



Figure 25. Application distribution, components

The following steps are required to enable a user to use the application:

1. The client image needs to be updated with the new package. These changes then need to be distributed to the client workstations that will be using this software.
2. Assign the package to the user using the `NETWIN USER userid /ADD /APP:appid.`
3. The user's profile is updated to include the link(s) to the icon(s) for the application.
4. The Win32 Application Updates file is updated to include the user registry information associated with the added application (from the `appiduser.inf` or `appiduser.reg` file). If the Win32 Application Updates file does not exist, it is created in the user's home directory and then updated.
5. A Logon Assignment is added to give the user access to the Win32 shared directory for the given application if the executable files are on a shared directory at the server. The assignments are based on the `APPDRIVE` and `APPDIR` values. No changes will be made if the Logon Assignment previously existed. The `USERS` group is used to manage access to the application files. The user is already part of this group, and the `USERS` group is added to the access list for the alias.
6. The updated user profile is written back to the home directory for the user.

An entry is added to the User Datastore record for this user. The entry indicates that the application has been assigned to the desktop indicated by the `TYPE`.

The WorkSpace On-Demand Client downloads an additional file (Windows Application Updates) from the user's home directory that is to be merged into the user's unique Windows registry.

---

## 4.2 Application type

The type of application support is based on the application type. Applications are placed into three broad categories:

- Well-behaved applications
- Server-based applications
- Client-based applications

### 4.2.1 Well-behaved applications

There are multiple initiatives from Microsoft and other vendors to provide server-based management of client platforms and applications. Two efforts from Microsoft include the Zero-Administration Windows (ZAW) initiative and the Windows Terminal Server product family.

While the efforts differ in approach, tools, system administrator requirements, and completeness of offering, they have one common aspect. All efforts attempt to support *well-behaved* Windows applications. The current definition of well-behaved is not very clear. With the Windows 2000 and the Windows Terminal Server products, there is hope that a more singular definition and certification effort will be put in place and adhered to. Microsoft currently has guidelines published along with Product Cookbook Information and scripts for applications that do not meet the guidelines, since there are very few, if any, well-behaved applications.

The Feature for Windows Clients will also support server-based management and delivery of well-behaved Windows applications. Our definition of well-behaved will be consistent with the ZAW definition (guidelines are part of the Zero Administration Kit for Windows 95 and the Zero Administration Kit for Windows NT support). A well-behaved application should at least meet the following four conditions:

- Server-based (installation through run-from-server or a similar mode)
- No application-specific r/w files per user
- Application registry changes per user
- No changes or replacement of system modules

Again, since there are few, if any, well-behaved applications, most applications fall into the following two categories.

#### 4.2.2 Server-based applications

Server-based applications could be defined as applications that are installed mainly at the server. These applications require some files on the client side. An example of such an application would be Lotus Smartsuite. Smartsuite requires a server side install as well as a node installation. A server-based application should at least meet the following five conditions:

- Server-based (installation through run-from-server or a similar mode)
- No application-specific r/w files per user
- Application registry changes per user
- No changes or replacement of system modules
- Client-side application files

#### 4.2.3 Client-based applications

Client-based applications usually require all modules to be installed on the client. There is no application installation at the server. An application is client-based when it meets implements at least one of the following:

- System registry changes
- Application-specific r/w files per user
- Application registry changes per user
- Change, replacement, addition of system modules
- Complete control of the system during execution

#### 4.2.4 Selecting applications

At this time, there are very few applications that can be considered well-behaved even from Microsoft. Fortunately, application management of the Feature for Windows Clients, at a minimum, provides server-based management of the applications. The system administrator controls application access by either providing or not providing the icon necessary to initiate the application from a given user's desktop.

For this most extreme set of applications, the executable files can be stored in common locations on the client platforms that need to execute the application. These applications must be pre-distributed to each client platform intended to support users executing the application. Within the document, this set of applications is referred to as *client-based* applications.

When selecting applications for the future, and the environment is cost sensitive, it would be best to find applications that match the well-behaved definition as closely as possible.

---

### 4.3 Capturing and creating an application package

As we described in Section 4.1.1, “Overview: Capturing and creating an application” on page 85, capturing an application involves a number of steps. In this section, we describe, using the StarOffice application package, the steps involved in capturing and creating an application package in more detail.

Our environment is described in Figure 26.

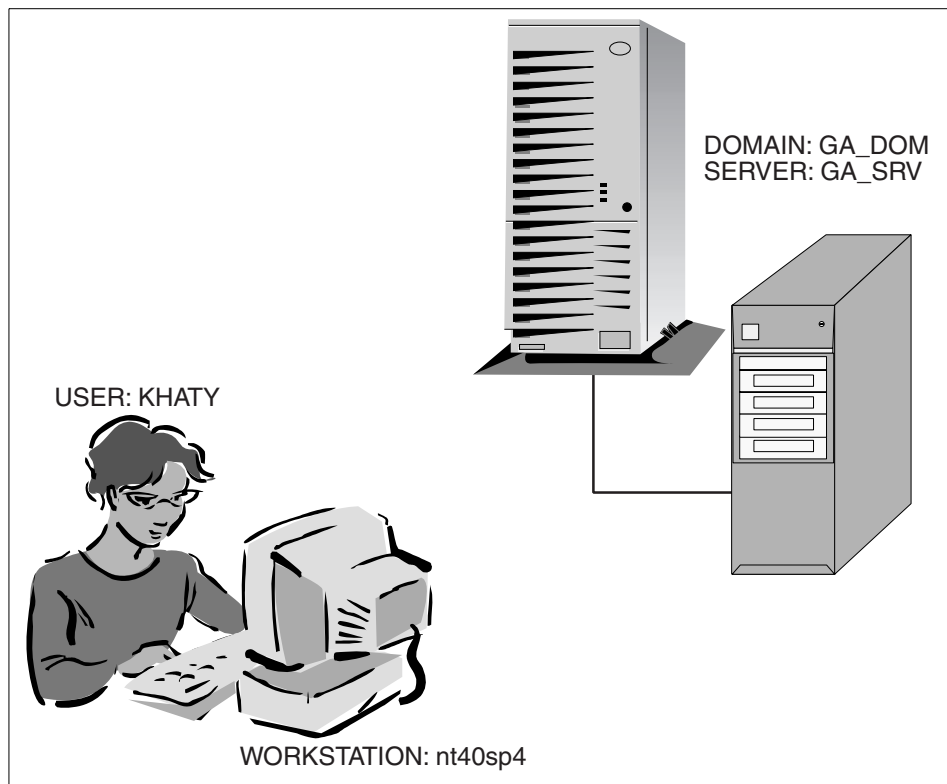


Figure 26. The lab environment

### 4.3.1 Define and set up the sandbox machine

#### Using a Clean Sandbox

The best way to install new application packages is to use a pristine client. Define a new client on the server or use the `/reset` option to refresh the installed operating system on your sandbox client.

This will ensure a clean install and a proper capture of all necessary data for the creation of your new application packages.

A sandbox is a client in the domain with an unrestricted desktop. This unrestricted desktop enables you to change registry entries, files on the sandbox, and save those changes. You must use a sandbox to create application packages. This unrestricted desktop also gives you access to the CRTPKG utility that allows you to capture application information.

See Section 2.4.3, “A sandbox” on page 42 for more information on sandbox clients. The sandbox machine is defined by using the `NETWIN RIPLMACH` command with the `/SANDBOX` switch.

The following are additional requirements for the sandbox:

- The sandbox must reside in the domain where you will run the application package.
- The client must be defined with the appropriate operating system. The system administrator must execute this on the platform that will be used by the end-user (for example, a Windows 95 system for applications that will be used by Windows 95 users).
- The sandbox must be used exclusively for package creation. Since all system changes are captured during this process, no other activity can be done during the package creation time.
- The sandbox has a similar hardware configuration to the other clients in the domain. Applications with specific hardware requirements will not operate on clients that do not have the correct environment. For example, if an application requires high video resolution and sound capabilities, the sandbox and all the clients must have those capabilities.

In our environment, we used the following procedure to complete this step:

1. Log on as an administrator to the domain where the client is going to be defined.
2. At a command prompt, issue the following command:

```
netwin riplmach nt40sp4 /add /mac:08005acf8abe /release:nt40 /type:winnt  
/dhcp:n /enabled:y /image:w32dossb.img /recordid:r_dtk_ndis  
/partition:600 /cdkey:010-1578707 /REGUSER:"Khatu Muvenda" /SANDBOX
```

For more details, refer to Section 3.7, “Initial testing” on page 76.

Once the sandbox has been defined, boot the sandbox machine and allow it to complete setup

#### 4.3.2 Define a sandbox user account

A sandbox user account is a user defined with administrator privileges. Administrator privileges are required to enable the CRTPKG utility to update files and system registry keys on the sandbox, as well as give the ID unrestricted access to the manipulate files on the server.

You will need to assign a home directory to the sandbox user. The home directory drive needs to be on a drive that support long file names.

If the user previously existed, delete any desktops that were associated with the user. To delete any desktops that were associated with the user account, use the `NETWIN USER` command.

##### Note

Do not use the name administrator for a user account. This term is reserved for local functions on Windows NT Clients.

In our example, we created a user called SBNTUSR. We created one userid for NT and one for Windows 9x for all our application package creation requirements. To create this user, we completed the following steps.

1. Log on to the boot server as an administrator.
2. Create a user account with administrator privileges:
  - Open **LAN Services** from the Desktop.
  - Open the **LAN Server Administration**.
  - Select **OK**.
  - Open **User Accounts**.
  - Drag the **UserID Template** to an open area in the user account folder.

The create notebook is displayed. Complete all the fields with the appropriate settings page. The fields we filled in set the password, set the account information to administrator, and assigned a home directory as show in Figure 27 on page 93.



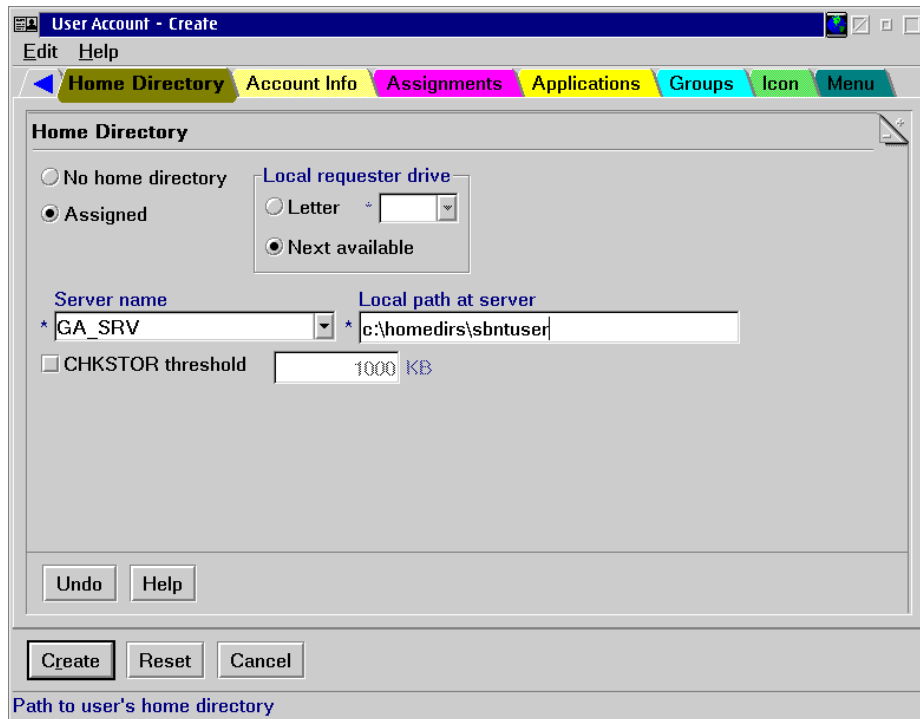


Figure 27. Creating a sandbox user account

3. Assign this user account a home directory on any directory with HPFS capabilities. HPFS supports long file names used by Windows operating system.

When you log on to the sandbox with this user account, you have an unrestricted Windows client desktop. This enables you to save and manipulate the desktop.

#### 4.3.3 Create application directories

Application directories are locations on the primary domain controller, or an application server, where you will store applications for your domain.

#### Note

To maintain security, store applications in a location outside of the IBMLAN directory. Subdirectories within the IBMLAN directories have limited access that are used to control, among other things, RPL files. Storing applications here could compromise these access controls.

This is the data location where the application data will be installed. Select a location for the application software. In our example, we created a directory called c:\apps\wntapps with an alias WNTAPPS for Windows NT applications. We also created an alias called W9xAPPS for the directory d:\apps\w9xapps for Windows 95 and 98 applications.

Aliases can be created using either the `NETGUI` command or the `NET SHARE` command. Since the sandbox user is an administrator, he or she has full access to these directories, and other users only need read and execute access.

Each application should have a unique subdirectory within the alias. For example, Netscape for Windows NT would have a directory d:\apps\wntapps\netscape and would be accessed from the client as x:\netscape where d:\apps\wntapps is the path defined by the alias.

#### Note

During this process, try to use the same drive letter each time you connect to a particular alias even if you connect to the aliases at different times. This could, at times, cause errors when you assign the application package to the user.

To create an application directory alias, we completed the following steps:

- Open **LAN Services** from the Desktop.
- Open **LAN Server Administration**.
- Open **Resource Definitions**.
- Drag the **Directory Template** to an open area in the Resource Definitions folder. Create notebook is displayed. Complete all the fields with appropriate settings page.

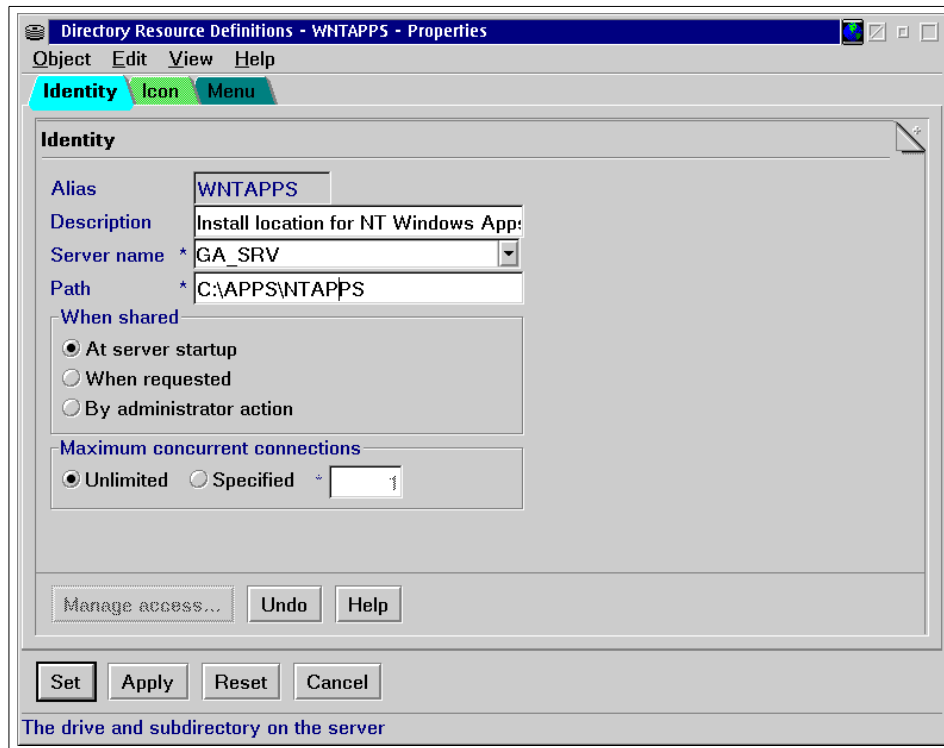


Figure 28. Creating an application directory

In our example, as shown in Figure 28, we created an alias called WNTAPPS that has a physical location on the C drive of server GA\_SRV.

#### 4.3.4 Create the application package

During this phase of the application capture process, you need to log onto the sandbox and create the application package. There are three steps to creating the application package:

- Start the CRTPKG utility.
- Install and customize the application.
- Complete the application creation.

Figure 29 on page 96 describes the process and the details that are captured.

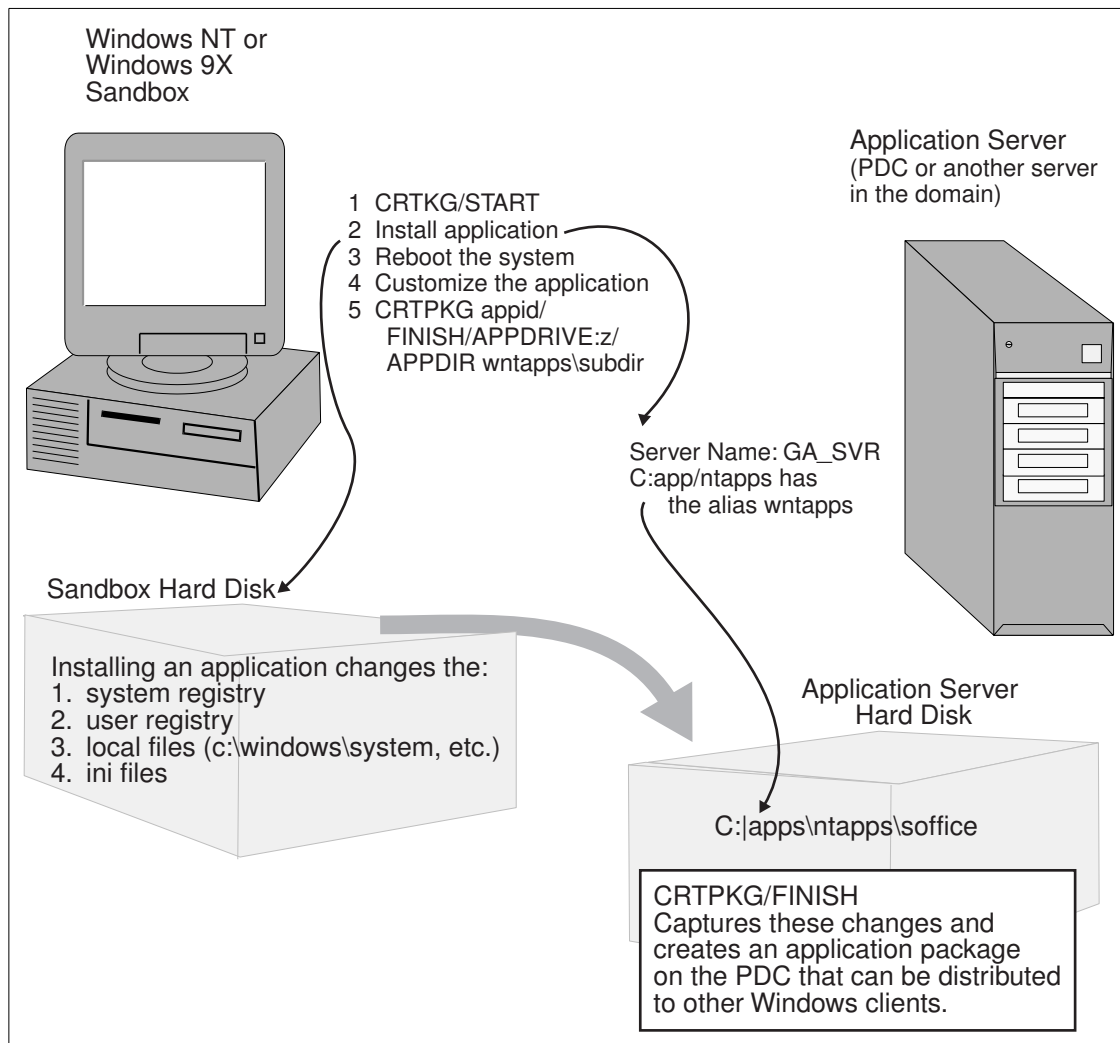


Figure 29. Capturing the application

#### 4.3.4.1 Starting the CRTPKG utility

The CRTPKG utility takes snapshots of the sandbox system before and after you install applications. The difference between the snapshots shows what files you installed and the changes the installation made to the Windows registry.

The CRTPKG.INI file specifies what directories, drives, registry keys, and files the CRTPKG utility excludes from the snapshots. Edit this file to change

the properties of the snapshots. The CRTPKG.INI file resides in the C:\APPLCPTR directory.

In our example, we did the following:

1. Log on to the sandbox with the user account sbntuser. This is the account we created in Section 4.3.2, “Define a sandbox user account” on page 92.
2. Verify that the CRTPKG.INI file represents all the requirements. In our case, the default was fine. The default should be fine in most cases.
3. If you are installing the application onto a local directory, this step would not be necessary.

Connect to the primary domain controller by using your alias and the `NET USE` command. This connection enables you to remotely install your application onto your server.

**Note**

You must associate a unique drive letter with each alias whenever you create an application package. Write down the drive letter you associate with the alias. If you associate the same drive letter with different aliases when you create subsequent application packages, you will be unable to assign the application package to a user account.

In our example, at a DOS prompt, we issued the command:

```
NET USE W: \\GA_SRV\WNTAPPS
```

**Note**

With Windows 95 and Windows 98, you would use the `DLSNET` command to connect to the server with only an alias and a remaining path (rather than a network path). Run this command on sandboxes before you connect to the server.

4. Close any open windows and change to the directory c:\applcptr directory where your CRTPKG.EXE file resides.
5. At a DOS prompt, type:

```
CRTPKG /START
```

#### 4.3.4.2 Installing and customizing the application

The following set of instructions would be different for each application that you are installing. The *Administrator's Guide* contains instructions for a number of other applications. This section focuses on installing and

customizing StarOffice. To install StarOffice on our server and sandbox client, we did the following:

1. Attach to an alias containing the StarOffice Application Source. In our case, we copied the StarOffice CD into an alias on the server. If the sandbox machine had a CD drive, the application could be installed from there. We issued the command: `net use s: \\GA_SRV\source`
2. On the S drive or CD drive, change to the application directory and start the application install. StarOffice is a two part installation. The first part installs all the server components, and the second part installs just the workstation components. So, the first part instructs you to do a setup with a /net switch. We issued the command: `w:\soffice\prod_w95\setup /net`
3. On the welcome window, click the on **Next** button. Accept the license agreement, and click on the **Accept** button.
4. Read the information displayed and click on the **Next** button. Then select **Custom Install** and click on the **Next** button.
5. Select `w:\SOFFICE` as the location in the space provided. This is the location where the server component of StarOffice is going to be installed into. Then select the **Next** button.
6. In our example, we selected the default components and clicked on the **Next** button.
7. To begin the installation of StarOffice, click on the **Complete** button.
8. The installation starts and takes about 10 minutes. When the installation completes, click on the **Complete** button. This completes the server part of the StarOffice installation.
9. Next, we needed to install the client component of StarOffice. To do this, go to the command line and change to the `w:\SOFFICE` directory. The client component needs to be installed from the directory that was installed onto the server.
10. To start the client installation issue, the `setup.exe` command.
11. On the Welcome window, click on the **NEXT** button. Accept the license agreement by clicking on the **Accept** button.
12. Read the information displayed and click on the **NEXT** button.
13. In the registration form, fill in the Company field exactly. For example, the company field could be International Business Machines. Then click on the **NEXT** button.
14. Next, fill in the media field and click on the **NEXT** button.

15. Select **Standard Workstation Installation**, then click on the **NEXT** button. This installs just the small workstation component that is less than 5 MB and uses the other files that were installed previously on the server. Alternatively, you could install the entire image which is more than 100 MB. This would increase the size of the corresponding image by the same amount.
16. You need to select a location. We selected to install StarOffice on the C drive in the \Office50\ directory. Click on the **NEXT** button and acknowledge the folder creation.
17. To start the file copy, click on the **Complete** button. At the end of the installation, you are asked for Java installation. *Do not install* this component.
18. You will see the StarOffice folder on the desktop. Do not close the folder for now. Just restart the sandbox once the installation is done.
19. After restarting the sandbox, logon with the sandbox userid, in our example, sbntuser.
20. You should now customize the Windows NT desktop. We wanted just the StarOffice icon on the desktop and did not want it available through the Start button, so we had to customize our desktop as follows:
  - Drag and drop the **StarOffice 5.0** icon from StarOffice 5.0 folder to the desktop.
  - Delete the **StarOffice Setup** icon.
  - Close the **StarOffice 5.0** folder.
  - Open the **Taskbar properties** window.
  - Go to the **Start Menu Programs**.
  - Click on the **Advanced** button.
  - Delete the **StarOffice 5.0** folder.
  - Close the **Exploring** window.
  - Close the **Taskbar properties** window.
21. Start StarOffice and allow it to initialize, then close it to complete initialization.
22. Since we are still connected to the two connections we made earlier, we could disconnect from the source drive by issuing the command `net use s: /d`. The application installation and customizing is now complete. Now, the application needs to be defined on the workstation and the server. These steps are defined in Section 4.3.4.3, "Completing the application package" on page 100.

#### 4.3.4.3 Completing the application package

Depending on the application you installed, there are a number of items that you may want to check before you continue to define the application.

- The way you customize the links and Start menu entries is the way that those links and entries will be displayed on the user's desktop. For example, you can:
  - Add shortcut icons to the desktop, as we did above.
  - Remove links to toolbars.
  - Add README files or text files that explain applications to users.
  - Place items in the start folder.
  - Remove items from the start up folder.
  - Update the Start menu entries.
- Close all windows and application windows that may cause updates to occur to the application image.

Once you have checked all the above, you need to perform the following steps to complete the application package:

1. Change to the c:\APPLCPTR directory.
2. At a DOS prompt, type:

```
CRTPKG appid /FINISH /APPDRIVE:X /APPDIR:"pathname"
```

In our example with StarOffice, we used the following command:

```
crtpkg soffice /finish /appdrive:w /appdir:"wntapps\soffice"
```

The output from this command was as follows:

```
Creating the application package: SOFFICE
Application package destination:
    \\GA_SRV\C$\IBMLAN\DCDB\WIN32APP\SERVER\SOFFICE
    \\GA_SRV\C$\IBMLAN\DCDB\WIN32APP\CLIENT\SOFFICE
Getting the file system difference...
Number of file changes captured: 221
Getting the system registry difference...
Number of registry changes captured: 263
Getting the user registry difference...
Number of registry changes captured: 10
Application package successfully created.
```

When you run the `CRTPKG /FINISH` command, a snapshot of the system is taken. This is compared against the snapshot taken when the `CRTPKG /START` command was issued and the differences extracted. These differences are collected and stored on the server.



The CRTPKG utility creates two sets of application files: files that you must copy to the client and files that always reside on the server. The CRTPKG utility sends both sets of files to the IBMLAN\DCDB\WIN32APP directory on the primary domain controller.

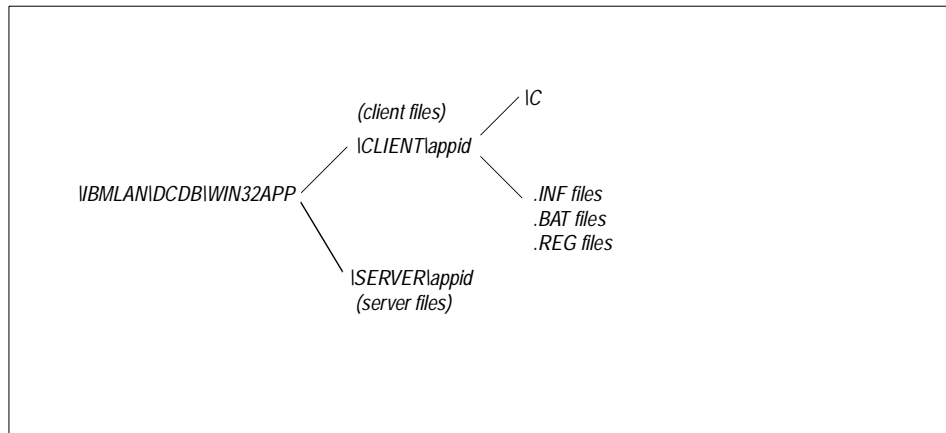


Figure 30. Directory tree for an application package

#### 4.3.4.4 Client application files

The client application files reside in the \\BMLAN\DCDB\WIN32APP\CLIENT\appid directory (appid represents the application package name). In our example, appid would be SOFFICE.

The files that are included here are the system registry entries, information files, and a directory tree that you must copy to the client hard disks:

- The client files also contain registry changes that are made to the HKEY\_LOCAL\_MACHINE key. Windows 95 and Windows 98 application packages store these changes in the appidSYS.REG file. Windows NT application packages store these changes in the appidSYS.INF file.
- Changes that are made to INI files reside in the appidINI.INF files (appid represents the application package name).

### Note

A Windows NT application package contains a single appidINI.INF file. A Windows 95 or a Windows 98 application package can contain multiple appidINI.INF files. These files are numbered in the following format, appid INI 1.INF and appid INI 2.INF.

- The client file also contain log files in the <IBMLAN\DCDB>\WIN32APP\CLIENT\ appid directory. These log files, NEWFILE.LST and MODFILE.LST, list the files that were added or changed when the application package was created.
- A RENFILE.BAT file contains any long file names for the files in the <IBMLAN\DCDB>\WIN32APP\CLIENT\ appid directory. When you create an application package, the CRTPKG utility reduces long file names to 8.3 characters. The RENFILE.BAT file enables you to rename the files in the application package to their original long file names. See Section 4.4, “Distributing the application files to client machines” on page 104.
- In case of StarOffice, the following files in IBMLAN\DCDB\WIN32APP\CLIENT\SOFFICE need to run on the client:
  - Modelfile.lst
  - Newfile.lst
  - Renfile.bat
  - SOfficeini.INF
  - SOfficesys.INF

If a Windows NT application changes registry entries or INI files on the sandbox, the application package contains a single appidSYS.INF file and a single appidINI.INF file. If a Windows 95 or a Windows 98 application changes registry entries or INI files on the sandbox, the application package contains a single appidSYS.REG file and multiple appidINI.INF files in the following format, appidSYS#.INF and appidINI#.INF.

The \IBMLAN\DCDB\WIN32APP\CLIENT\ appid directory may contain two log files, NEWFILE.LST and MODFILE.LST, and a batch file called RENFILE.BAT. The logs list the files that were added or modified when the application package was created.

The RENFILE.BAT file contains the long file names for the files in the \IBMLAN\DCDB\WIN32APP\CLIENT\ appid directory. When you create an application package, the CRTPKG utility reduces long file names to only 8.3 characters. The long file names are in the RENFILE.BAT. This file

enables you to rename the files in the application package to their original long file names when you distribute the client application files to the clients.

#### **4.3.4.5 Server application files**

Server application files reside in the <IBMLAN\DCDB>\WIN32APP\SERVER\appid directory. Server application files have the following attributes:

- They define the application package on the server and define the application icons and Start menu entries that are assigned to user accounts.
- They contain the registry changes that are made to the HKEY\_CURRENT\_USER key. These changes reside in the appidUSER.INF files (appid represents the application package name) for Windows NT application packages. These changes reside in the appidUSER.REG files (appid represents the application package name) for Windows 95 and Windows 98 application packages.
- The actual application resides in the directory you specified with your alias during the installation. Make any changes to the application before assigning the application to user accounts.

The server application files are sent to the \IBMLAN\DCDB\WIN32APP\SERVER\appid directory (appid represents the application package name). The file names are:

- SOfficeuser.INF
- SOffice.Dat

These files define the application package on the server and define the application icons assigned to user accounts. The registry changes that were made to HKEY\_CURRENT\_USER, which modify desktops, are captured in the appidUSER.INF for Windows NT and appidUSER.REG files for Windows 9x.

### **4.3.5 Define the application package**

Once the application has been installed and configured, and the changes captured, you must define the package on the primary domain controller.

Use the `NETWIN APP` command to verify that the application package is on the primary domain controller. This command also enables you to describe the contents of the application package and define the application package. When you define an application package, you add application package information to the APPSTORE.INI file.

The APPSTORE.INI file is in the <IBMLAN\DCDB>\WIN32APP directory on the primary domain controller. The path to the \IBMLAN\DCDB\WIN32APP directory may vary if you specified a location other than the default during the installation of the feature. The default location is the \IBMLAN\DCDB\ directory.

The APPSTORE.INI file stores information for every application package. This information includes the location of the application in the application package and the properties of the application package. This information is used later to assign the application package to user accounts. To define an application package:

1. Log on to the primary domain controller as an administrator.
2. At a command prompt, type: `NETWIN APP appid /ADD /REMARK:"text"`

In our example with StarOffice, we used the command:

```
NETWIN APP soffice /ADD /REMARK:"StarOffice for Windows NT 4.0"
```

---

## 4.4 Distributing the application files to client machines

Once the application has been captured, created, and defined on the server, most application packages require some files to be installed at the client machine. In order for a user to access the application, these files must be distributed to the client. These files can be distributed to the client using one of two methods:

- Use a commercial software distribution product
- Use the client image

### 4.4.1 Using commercial software distribution products

Many commercial software distribution products are available for Windows NT and Windows 9x. One is available from Tivoli and using the Tivoli Management Agent (TMA) during the client operating installation simplifies the process. For more information, refer to Chapter 7, "The Tivoli TMA client" on page 205.

Any product you select would need to be configured, and for each workstation, you would need to distribute the related product files and update the registry and INI files accordingly.

### 4.4.2 Distributing files within the client image

Distributing files using the client image means that when the operating system gets installed the files are automatically copied across. It is a post

operating system install processes that copies the application files to the appropriate locations and modifies the client image accordingly.

By distributing the client application files with the client image, the client application files are automatically copied to your clients anytime you reinstall the operating system.

When you create an application package, the `CRTPKG` utility copies client files to the `<IBMLAN\DCDB>\WIN32APP\CLIENT\appid` directory. These files are information files, system registry files, and a copy of the sandbox directory tree. These files must reside on the local hard disk of the client. If these files do not reside on a client, users cannot access applications from that client. These files are described in Table 10.

*Table 10. Application files on the primary domain controller hard disk*

<b>Files:</b> <b>x:\&lt;IBMLAN\DCDB\WIN32APP\CLIENT\appid</b>	<b>Description</b>
C subdirectory	Directory structure of all new or modified files
appidSYS.INF files (for Windows NT) appidSYS.REG files (for Windows 9x)	System registry changes
appidINI.INF	INI file changes
NEWFILE.LST, MODFILE.LST	List of new and modified files
RENFIL.BAT	BAT file for converting short filenames to long names. This file contains the long file names for the files in the <code>&lt;IBMLAN\DCDB&gt;\WIN32APP\CLIENT\appid</code> directory. When you create an application package, the <code>CRTPKG</code> utility shortens file names in the application package to 8.3 characters. The <code>CRTPKG</code> utility saves any long file names in the <code>RENFIL.BAT</code> file. This file enables you to rename the files in the application package to their original long file names when you distribute the client application files to the clients.

The `<IBMLAN\DCDB>\WIN32APP` directory is on the primary domain controller. The client image directory resides on either the primary domain controller or on any additional boot servers in your domain. When you define and start a client, the installation process copies the client image from a boot server to the client's hard disk.

#### Note

The Windows NT client image resides on the boot server. If you have multiple boot servers, copy the client application files to the Windows NT client image on every boot server, and update the CMDLINES.TXT file and \$RENAME.TXT file on every boot server.

### 4.4.3 Updating the Windows NT client image

If the application is a Windows NT application, and you created the application package on a client running Windows NT, copy the client application files to the Windows NT client image. Update the CMDLINES.TXT file to run the RENFILE.BAT file and a script that adds application package registry information to the client system registry when the operating system is installed on the client. Complete the following steps:

1. Log on to the boot server as an administrator. If you have HPFS386, logging on as administrator ensures that the user access controls are inherited when directories are created.
2. Change to the \IBMLAN\RPL\NT40\I386\SOEM\$ directory. Verify that the SOEM\$ directory contains a \$WINTMP subdirectory. If you do not have a \$WINTMP subdirectory, you must create one. At a command prompt, type: MD \$WINTMP
3. Change to the <IBMLAN\DCDB>\WIN32APP\CLIENT\appid\C\WINNT directory. This directory is on the primary domain controller. Compare the files in the WINNT directory to the files in the \IBMLAN\RPL\NT40\I386\SOEM\$\\$WINTMP directory. If any of the files are the same, place only the most recent version of the file on the client image.

#### Note

You do not need to compare files if you update the client image every time you create an application package on the sandbox.

4. Copy the contents of the c:\IBMLAN\DCDB\WIN32APP\CLIENT\appid\C\WINNT directory to the Windows NT client image at c:\IBMLAN\RPL\NT40\I386\SOEM\$\\$WINTMP.

At a command prompt, type:

```
XCOPY c:\IBMLAN\DCDB\WIN32APP\CLIENT\appid\C\WINNT\*.*
X:\IBMLAN\RPL\NT40\I386\SOEM$\$WINTMP/S /E /T /H
```

In our StarOffice example, we used the command:

```
XCOPY c:\IBMLAN\DCDB\WIN32APP\CLIENT\soffice\C\WINNT\*. *
X:\IBMLAN\RPL\NT40\I386\SOEM$\SWINTMP/S /E /T /H
```

5. Copy any additional subdirectories and files (other than the WINNT subdirectory that is located in the C directory to the Windows NT client image. To copy additional subdirectories, at a command prompt, type:

```
XCOPY c:\IBMLAN\DCDB\WIN32APP\CLIENT\appid\C\subdirectory
X:\IBMLAN\RPL\NT40\I386\SOEM$\C\subdirectory /S /E /T /H
```

(subdirectory represents the name of any directories in the C directory, other than the WINNT directory.)

This adds a new subdirectory in the \$OEM\$ directory on the client image. If a window or message prompts you to create a new subdirectory, type YES. If this subdirectory already exists, the files in the <IBMLAN\DCDB>\WIN32APP\CLIENT\appid directory are merged with the files in the existing directory.

In our StarOffice example, we used the command:

```
XCOPY c:\IBMLAN\DCDB\WIN32APP\CLIENT\appid\C\Office50
X:\IBMLAN\RPL\NT40\I386\SOEM$\C\Office50 /S /E /T /H
```

The Office50 directory was the only other directory with our example.

To copy any files in the C directory, at a command prompt, type:

```
XCOPY c:\IBMLAN\DCDB\WIN32APP\CLIENT\appid\C\filename
X:\IBMLAN\RPL\NT40\I386\SOEM$\C /T /H
```

(filename represents the name of any files in the C directory.)

6. If you copied the subdirectories PROGRA~1 or MULTIM~1 to the client image in the previous step, assign these subdirectories long names in the \$\$RENAME.TXT file.

To change the directory names:

- Change to the \IBMLAN\RPL\NT40\I386\SOEM\$\C directory.
- Open \$\$RENAME.TXT in any ASCII text editor (if the file did not exist, it was automatically created when you open the file).
- Type the following entry to rename PROGRA~1 and MULTIM~1:  
[]  
Progra~1="Program Files"  
Multim~1="Multimedia Files"

In the StarOffice example, we did not have any additional directories.

7. Change to the <IBMLAN\DCDB>\WIN32APP\CLIENT\appid directory. If you have any INF files in your <IBMLAN\DCDB>\WIN32APP\CLIENT\

appid directory, rename each file to a unique name that is up to eight characters in length.

For example, at a command prompt, type:

```
RENAM APPLICATIONINI.INF APPINI.INF
```

In our StarOffice example, at a command prompt, we entered the following commands:

```
RENAM SOFFICEINI.INF OFFINI.INF
RENAM SOFFICESYS.INF OFFSYS.INF
```

8. Copy the INF files you just renamed in the previous step in the <IBMLAN\DCDB>\WIN32APP\CLIENT\ appid directory to the \IBMLAN\RPL\NT40\I386\OEM\$ directory.

At a command prompt, type:

```
XCOPY <IBMLAN\DCDB>\WIN32APP\CLIENT\ appid\*.INF
X:\IBMLAN\RPL\NT40\I386\OEM$
```

In our StarOffice example, we entered the following command:

```
XCOPY <IBMLAN\DCDB>\WIN32APP\CLIENT\ soffice\*.INF
X:\IBMLAN\RPL\NT40\I386\OEM$
```

Write down the names of any INF files you copied. In our StarOffice example, the file names were OFFINI.INF and OFFSYS.INF

If the application package contains any files with long file names, you will have a RENFILE.BAT file in your <IBMLAN\DCDB>\WIN32APP\CLIENT\appid directory. If you have a RENFILE.BAT file, copy RENFILE.BAT to the \IBMLAN\RPL\NT40\I386\OEM\$ directory.

At a command prompt, type:

```
XCOPY <IBMLAN\DCDB>\WIN32APP\CLIENT\ appid\RENFILE.BAT
X:\IBMLAN\RPL\NT40\I386\OEM$
```

In our StarOffice example, we issued the command:

```
XCOPY <IBMLAN\DCDB>\WIN32APP\CLIENT\ soffice\RENFILE.BAT
X:\IBMLAN\RPL\NT40\I386\OEM$
```

If you are assigning multiple application packages, and this file already exists for another application package, assign the RENFILE.BAT file a unique name up to eight characters in length. You can also merge the contents of the two RENFILE.BAT files. To assign the file another name type:

```
RENAM RENFILE.BAT RENFILE2.BAT
```



Copy each of the multiple RENFILE.BAT files (RENFILE1.BAT and RENFILE2.BAT) to the \IBMLAN\RPL\NT40\I386\SOEM\$ directory.

9. Change to the \IBMLAN\RPL\NT40\I386\SOEM\$ directory. Open the file, copyfiles.cmd in any ASCII text editor. Verify that the following command resides in the file:

```
XCOPY .\SWINTEMP C:\WINNT /S /E /H /
```

#### Copying Application Files

You cannot redirect output or use the `RENAME` command in the `CMDLINES.TXT`. This is a change from the Administrators Guide. The reason for this is so the output from the `XCOPY` command can be redirected and so `RENAME` statements, for renaming files in use, can be done in the `COPYFILE.CMD` file.

10. Change to the \IBMLAN\RPL\NT40\I386\SOEM\$ directory. Open the file `cmdlines.txt` in any ASCII text editor. In the appropriate section of the file (the file is commented), uncomment the `copyfile.cmd` command:

```
".\copyfile.cmd"
```

11. Also in the `cmdlines.txt` file, after the `copyfile.cmd` file is called, type the following command for every INF file in your <IBMLAN\DCDB>\WIN32APP\CLIENT\appid directory:

```
"rundll32 setupapi.dll,InstallHinfSection DefaultInstall 128 .\name.INF"
```

(name represents the name of the .INF file.)

#### Note

If the `CMDLINES.TXT` file contains commands to install service packs, insert the `rundll` statements after the service pack information.

In our StarOffice example, the soffice subdirectory has two INF files, `OFFSYS.INF` and `NOTINI.INF`, so we typed:

```
"rundll32 setupapi.dll,InstallHinfSection DefaultInstall 128  
.\OFFINI.INF"  
"rundll32 setupapi.dll,InstallHinfSection DefaultInstall 128  
.\OFFSYS.INF"
```

12. If you have a RENFILE.BAT file in the `CMDLINES.TXT` file after the commands for the .INF files, type:

```
".\RENFILE.BAT"
```

If you have multiple RENFILE.BAT files, add this command for every RENFILE.BAT file. For example, if you have a RENFILE.BAT file and a RENFILE2.BAT file, type:

```
".\RENFILE.BAT"  
".\RENFILE2.BAT"
```

13. Change the version level of the client image. This enables you to identify the version of the client image on each boot server and on each client. You can update this file whenever you make changes to the client image. To change the version level:
  - Open the \BMLAN\RPL\NT40\RSP\VERLEVEL.TXT file in any ASCII text editor.
  - Find the entry:  
[VersionLevels]  
winnt40=001
  - Change the number 001 to any other alphanumeric character.
14. Verify that any subdirectories you created on the client image have read and execute access controls. Run `GETRPL` or the `NET ACCESS /APPLY` command to change the access controls on the new subdirectories. If you formatted your partitions by using HPFS386 or JFS, you do not have to change the access controls of subdirectories. Partitions formatted with HPFS386 or JFS automatically inherit access controls from the parent directory.
15. Define and start your clients. The operating system installation process should now copy the client application files to the client's hard disk.

#### 4.4.4 Updating the Windows 9X client image

If the application is a Windows 95 or Windows 98 application, and you created the application package on a client running Windows 95 or Windows 98, copy the client application files to the Windows 95 and Windows 98 client images. Update the Windows 95 and Windows 98 default response files to run the RENFILE.BAT file and a script that adds the application package registry information to the client system registry when the operating system is installed on the client.

##### 4.4.4.1 Distributing Windows 95 application files

For Windows 95 clients, you need to complete the following steps:

1. Log on to a boot server as an administrator.
2. Change to the \BMLAN\RPL\W95\C directory. Verify that the \C directory contains a \SWINTMP subdirectory. If you do not have a \SWINTMP subdirectory, you must create one. At a command prompt, type:

MD \$WINTMP

3. Change to the \BMLAN\DCDB\WIN32APP\CLIENT\appid\C directory.  
(appid represents the application package name.)
4. Change to the \BMLAN\DCDB\WIN32APP\CLIENT\ appid directory.
5. Copy the WINDOWS subdirectory to the Windows 95 client image. At a command prompt, type:  

```
XCOPY WINDOWS X:\BMLAN\RPL\W95\C\WINTMP /S /E /H /T
```
6. Copy any additional subdirectories and files (other than the WINDOWS subdirectory) located in the \C directory to the Windows 95 client image. At a command prompt, type:  

```
XCOPY subdirectory X:\BMLAN\RPL\W95\C\subdirectory /S /E /H /T
```

  
(subdirectory represents the name of any directories or files in the \C directory, other than the WINDOWS directory.)
7. Change to the \BMLAN\DCDB\WIN32APP\CLIENT\ appid directory.
8. In the \BMLAN\DCDB\WIN32APP\CLIENT\appid directory, change the INF file to a unique name eight characters in length. Type:  

```
RENAME SOFFICEINI1.INF OFFINI1.INF
```
9. Copy the OFFINI1.INF files in your \BMLAN\DCDB\WIN32APP\CLIENT\ appid directory to the \BMLAN\RPL\W95\C directory (appid represents the application package name). At a command prompt, type:  

```
XCOPY OFFINI1.INF X:\BMLAN\RPL\W95\C /S /E /T /H
```
10. In the \BMLAN\DCDB\WIN32APP\CLIENT\ appid directory, rename any file with extension REG to a unique name that is up to eight characters in length. For example, at a command prompt, type:  

```
RENAME SOFFICESYS.REG OFFSYS1.REG
```
11. Copy the OFFSYS1.REG files in your  
    \BMLAN\DCDB\WIN32APP\CLIENT\ appid directory to the  
    \BMLAN\RPL\W95\C directory (appid represents the application package name). At a command prompt, type:  

```
XCOPY OFFSYS1.REG X:\BMLAN\RPL\W95\C /S /E /T /H
```
12. If the application package contains any files with long file names, you will have a RENFILE.BAT file in your \BMLAN\DCDB\WIN32APP\CLIENT\ appid directory. Copy the renamed RENFILE.BAT file to the \BMLAN\RPL\W95\C directory. At a command prompt, type:  

```
XCOPY RENFILE2.BAT X:\BMLAN\RPL\W95\C /S /E /T /H
```
- 13.. Change to the \BMLAN\RPL\W95\LOGON directory.

14.. Open the file LOGONCLT.BAT in any ASCII text editor.

15.. At the top of the file, type:

```
XCOPY C:\$WINTMP\*.* C:\WINDOWS /S /E /H /Y
```

16.In the LOGONCLT.BAT file, after the `XCOPY` command, type the following command for all the files you have copied with the extension INF to the \IBMLAN\RPL\W95\C directory. Type:

```
rundll32 setupx.dll,InstallHinfSection DefaultInstall 128 C:\OFFINI1.INF
```

17.At the top of the LOGONCLT.BAT file, after the commands for the files with extension INF, type the following command for the FFSYS1.REG file that you copied to the \IBMLAN\RPL\W95\C directory:

```
REGEDIT /S C:\OFFSYS1.REG
```

18.In the LOGONCLT.BAT file, after the `REGEDIT` commands, type:

```
CALL C:\RENFILE.BAT
```

19.Change the version level of the Windows 95 client image.

This enables you to identify the version of the client image on the boot server and the version of the client image installed on each client.

- Open the \IBMLAN\RPL\W95\RSP\VERLEVEL.TXT file in any ASCII text editor.
- Find the entry:

```
[VersionLevels]  
win95=001
```

- Change the number 001 to any other alphanumeric character.

20.. Define and start your clients. The client application files are copied to the client's hard disk along with the operating system.

#### 4.4.4.2 Distributing Windows 98 application files

The Windows 98 client image resides in the \IBMLAN\RPL\W98\OS directory. To distribute Windows 98 application files, complete the following steps:

1. Log on to the boot server as an administrator.
2. Change to the \IBMLAN\RPL\W98\C directory. Verify that the \C directory contains a \WINTMP subdirectory. If you do not have a \WINTMP subdirectory, you must create one. At a command prompt, type:  

```
MD $WINTMP
```
3. Change to the \IBMLAN\DCDB\WIN32APP\CLIENT\appid\C directory. (appid represents the application package name.)
4. Change to the \IBMLAN\DCDB\WIN32APP\CLIENT\ appiddirectory.

5. Copy the WINDOWS subdirectory to the Windows 98 client image. At a command prompt, type:

```
XCOPY WINDOWS X:\IBMLAN\RPL\W98\C\SWINTEMP /S /E /H /T
```

6. Copy any additional subdirectories and files (other than the WINDOWS subdirectory) located in the \C directory to the Windows 98 client image. At a command prompt, type:

```
XCOPY subdirectory X:\IBMLAN\RPL\W98\C\subdirectory /S /E /H /T  
(subdirectory represents the name of any directories or files in the \C  
directory, other than the WINDOWS directory.)
```

7. Change to the \IBMLAN\DCDB\WIN32APP\CLIENT\appid directory.
8. In the IBMLAN\DCDB\WIN32APP\CLIENT\appid directory, change the INF file to a unique name eight characters in length. Type:

```
RENAME SOFFICEINI1.INF OFFINI1.INF
```

9. Copy the OFFINI1.INF files in your \IBMLAN\DCDB\WIN32APP\CLIENT\appid directory to the \IBMLAN\RPL\W98\C directory (appid represents the application package name). At a command prompt, type:

```
XCOPY OFFINI1.INF X:\IBMLAN\RPL\W98\C /S /E /T /H
```

10. In the \IBMLAN\DCDB\WIN32APP\CLIENT\appid directory, rename any file with extension REG to a unique name that is up to eight characters in length. For example, at a command prompt, type:

```
RENAME SOFFICESYS.REG OFFSYS1.REG
```

11. Copy the OFFSYS1.REG files in your \IBMLAN\DCDB\WIN32APP\CLIENT\appid directory to the \IBMLAN\RPL\W98\C directory (appid represents the application package name). At a command prompt, type:

```
XCOPY OFFSYS1.REG X:\IBMLAN\RPL\W98\C /S /E /T /H
```

12. If the application package contains any files with long file names, you will have a RENFILE.BAT file in your \IBMLAN\DCDB\WIN32APP\CLIENT\appid directory. Copy the renamed RENFILE.BAT file to the \IBMLAN\RPL\W98\C directory. At a command prompt, type:

```
XCOPY RENFILE2.BAT X:\IBMLAN\RPL\W98\C /S /E /T /H
```

- 13.. Change to the \IBMLAN\RPL\W95\LOGON directory.
- 14.. Open the file LOGONCLT.BAT in any ASCII text editor.
- 15.. At the top of the file, type:

```
XCOPY C:\SWINTEMP\*.* C:\WINDOWS /S /E /H /Y
```

16. In the LOGONCLT.BAT file, after the `XCOPY` command, type the following command for all the files you have copied with the extension INF to the `\IBMLAN\RPL\W98\C` directory:

```
rundll32 setupx.dll,InstallHinfSection DefaultInstall 128 C:\OFFINI1.INF
```

17. At the top of the LOGONCLT.BAT file, after the commands for the files with extension INF, type the following command for OFFSYS1.REG file that you copied to the `\IBMLAN\RPL\W98\C` directory:

```
REGEDIT /S C:\OFFSYS1.REG
```

18. In the LOGONCLT.BAT file, after the `REGEDIT` commands, type:

```
CALL C:\RENFILE.BAT
```

- Change the version level of the Windows 98 client image.

This enables you to identify the version of the client image on the boot server and the version of the client image installed on each client.

1. Open the `\IBMLAN\RPL\W98\RSP\VERLEVEL.TXT` file in any ASCII text editor.
2. Find the entry:

```
[VersionLevels]
win98=001
```

3. Change the number `001` to any other alphanumeric character.

19. Define and start your clients. The client application files are copied to the client's hard disk along with the operating system.

#### 4.4.5 Renaming files that are in use on Windows NT

If you receive a `file in use` error message when you copy the application files to the client, do the following:

1. Log on to the boot servers as an administrator.
2. Change to the `\IBMLAN\RPL\NT40\I386\%OEM%` directory.
3. Open the `COPYFILE.CMD` file in any ASCII text editor.
4. Rename the file that is in use in the `COPYFILE.CMD` file.
5. The following is an example of renaming the file `msvcrt.dll` in the `COPYFILE.CMD` file:

```
rename c:\winnt\system32\msvcrt.dll msvcrt.tmp
xcopy .\%wintemp c:\winnt /s /e /h /k
```

6. Reset your client. Refer to the log file (`C:\RENAME.LOG`) that resides on the client to view any errors that occur when the `RENAME` command runs.

#### 4.4.6 Renaming files that are in use on Windows 9x

If you receive a `file in use` error message when you rename a file in the `LOGONCLT.BAT` file, do the following:

1. On the boot server, change to the `<IBMLAN\DCDB>\WIN32APP\CLIENT\appid` directory (`appid` represents the application package name).
2. Open the file `WININIT.INI` in any ASCII text editor (if the file does not exist it is created).
3. Rename the file that is in use in the `WININIT.INI` file. The following is an example of renaming the file `msvcrt.dll` in the `WININIT.INI` file:

```
[rename]
c:\windows\system\msvcrt.dll=c:\windows\system\msvcrt.tmp
```

4. Rename the file that is in use in the `$WINTMP` directory. For example, to rename `msvcrt.dll`, at a command prompt, type:

```
REN \ $WINTMP\SYSTEM\MSVCRT.DLL \ $WINTMP\SYSTEM\MSVCRT.TMP
```

5. Reset your client. You can use the `WININIT.INI` file to move or rename files during the installation of the client image. See the Administrator's Guide for a detailed description of `WININIT.INI`.

---

#### 4.5 Assigning and deleting applications for users

Once an application has been captured and defined, and the client files distributed, the application can be assigned to users. The `NETWIN USER` command is used to assign the application package to the user. Before an application is assigned to a user, the following must be done:

1. The user must be defined. This is done using the `NETGUI` or the `NET USER` command.
2. The user must be assigned a desktop. This is done using the `NETWIN USER` command with the `/desktop` switch:

```
NETWIN USER userid /DESKTOP:pathname /TYPE:{WIN9X | WINNT}
```

##### 4.5.1 Assigning an application to a user

To assign an application package to a single user account, you need to do the following:

1. Log on to the primary domain controller as an administrator.
2. At the command prompt, type:

```
NETWIN USER username /ADD /APPID:appid
```

In our StarOffice example, to add the StarOffice application to the user Axel, we used the command:

```
NETWIN USER axel /ADD /APPID:soffice
```

When an application is assigned to a user, the following things happen:

1. The Windows Application Datastore, the APPSTORE.INI file, is read to determine if the package exists. If it exists, the application package directory (IBMLAN\DCDB\WIN32APP\appid) is accessed to get the needed information to update the user.
2. The user account is accessed to get the user information.
3. The User Datastore (USRSTORE.INI) is accessed to determine if the user already has the application assigned to their desktop. If so, an error message is issued and no further processing takes place.
4. The user's home directory is accessed to get the Windows user profile information.
5. The Desktop folder is updated to include the link(s) to the icon(s) for the application. The icon information will come from the Application Package directory.
6. The Start Menu folder is updated to include an entry(s) for the application. The information used will come from the Application Package directory.
7. The Programs folder is updated to include an entry(s) for the application. The information used will come from the Application Package directory.
8. The Windows Application Updates file is updated to include the user registry information associated with the added application (from the appiduser.inf file). If the Win32 Application Updates file does not exist, it is created in the user's home directory and then updated.
9. A logon assignment is added to give the user access to the Windows shared directory for the given application if the executable files are on a shared directory at the server. The APPDRIVE and APPDIR values will be used. No changes are made if the logon assignment previously existed. The USERS group is used to manage access to the application files. The user is already part of this group, and the USERS group is added to the access list for the alias.
10. The updated user profile is written back to the home directory for the user.  
  
An entry is added to the User Datastore record for this user. The entry indicates that the application has been assigned to the desktop indicated by the type.



11. While not part of this operation, it is important to understand how the Windows registry information specific to an individual application is used. While this is specified in the logon portion of these flows, here are a few of the details.

During a logon from a Windows client, two files are currently downloaded from the server: the user's Windows user profile (containing the user portion of the Windows registry, the Desktop folder, the Start Menu folder, and a series of other folders which is all stored in the user's home directory), and the System Policy file (from the NETLOGON share on the domain controller). In addition, the Workspace On-Demand Client will attempt to download an additional file (Win32 Application Updates) from the user's home directory. If it exists, it will include the INF file information that defines the set of Windows registry changes that need to be made due to the specific applications that have changed. If changes have been made, the Windows client machine will issue Windows programming calls to merge the application specific registry values into the user's unique Windows registry (which is part of the user's Windows user profile). The user's Windows user profile will then be written back to the user's home directory at the server, and the Win32 Application Updates file will be deleted from the user's home directory. Following that, the System Policy file is processed by the Windows client machine which results in an additional set of changes to the Windows registry.

#### **4.5.2 Deleting an application from a user**

This activity is done on the Workspace On-Demand Domain Controller where the users are defined and where the application is provided. For packages that have been added to the domain controller, the application can be removed from users which updates their user profile to remove access to it.

The application packages must be managed from a primary domain controller. Administrator authority is required for command execution. To delete an application from a user, complete the following steps:

1. The domain controller is powered on, and the administrator logs on using an account with system administrator authority.
2. The domain controller must have the Workspace On-Demand Win32 Application Management feature installed. During the install of this feature, the Win32 Application Datastore will have been created on the domain controller. The Win32 Application Datastore is simply a file used to keep details about the Win32 applications that exist on the domain controller. Each time an application is added to the domain by the administrator, a Datastore entry is added. The Application Package

directory and Client Application Information directory are also created during the install and updated as the CRTPKG utility is run.

3. The administrator executes the command specifying the userid and appid of the desired application:

```
NETWIN USER userid /DELETE /APP:appid
```

4. The Win32 Application Datastore is read to determine if the package exists and to get the needed information to update the user. The Application Package directory is also be accessed to get appid specific information.
5. The user account must have been previously defined in the domain and is used to get the user information.  
The User Datastore is accessed to determine if this user has the application assigned to their desktop.
6. The user's home directory is accessed to get the Windows user profile information.
7. Within the user profile, the Desktop folder is updated to remove the link(s) to the icon(s) for the application. The icon information is from the Application Package directory for the specified appid.
8. The Start Menu folder is updated to remove the entry(s) for the application. The Start menu is also part of the Windows user profile. The entry information is from the Application Package directory for the specified appid.
9. The Programs folder is updated to remove the entry(s) for the application. The Programs folder is part of the Windows user profile. The entry information is from the Application Package directory for the specified appid.
10. The updated user profile is written back to the home directory for the user.  
The User Datastore record for the user is updated to remove the application from the set for the desktop.
11. If a logon assignment exists for the application, the logon assignment to the Win32 shared directory is removed. The user may have other Windows applications assigned to them on the same share.

#### **4.5.3 Enumerating an application for a user**

This activity is done on the Workspace On-Demand Domain Controller where the users are defined and the application(s) are provided. The command enumerates all Windows applications that have been assigned to a given

user on a given platform (Windows 9X, Windows NT). Information that will be enumerated includes: APPID, REMARK, and TYPE.

The application packages must be managed from a primary domain controller. Administrator authority is required for command execution. To enumerate an application for a user, complete the following steps:

1. The domain controller is powered on and the administrator logs on using an account with system administrator authority.
2. The domain controller must have the WorkSpace On-Demand Win32 Application Management feature installed. During the install of this feature, the Win32 Application Datastore will have been created on the domain controller. The Win32 Application Datastore is simply a file used to keep details about the Windows applications that exist on the domain controller.

3. The administrator executes the command:

```
NETWIN USER userid
```

To enumerate all applications for each type of desktop they have been assigned (WIN9X or WINNT).

Or

```
NETWIN USER userid /TYPE:{WIN9X | WINNT}
```

To enumerate all applications assigned to the user for the specified platform type (WIN9X or WINNT).

4. The user account must have been previously defined in the domain and is used to get the user information.
5. The User Datastore is accessed to get the set of applications assigned to the specified desktop type for the user.
6. Each of the APPIDs retrieved from the user record in the User Datastore is used to get the specific information for each appid. The application information (APPID, REMARK, and TYPE) is displayed for each icon from the Desktop folder of the user profile.

---

## 4.6 Application management commands

Application management is done on the primary domain controller for the WorkSpace On-Demand domain. Administrator authority is required for all command execution. Commands can either be executed directly at the

primary domain controller, or remotely using NetAdmin to direct the individual commands to a primary domain controller.

Table 11. Application management command summary

Routine	Description
Adding an Application Package	This activity is done on the primary domain controller for the WorkSpace On-Demand domain from which the application is to be managed. After the package has been created on the sandbox machine, this command will make the application available for the system administrator to assign to users.
Deleting an Application Package	This activity is done on the WorkSpace On-Demand Domain Controller from which the application is provided. This command removes the package from the selected domain controller.
Enumerating an Application Package	This activity is done on the WorkSpace On-Demand Domain Controller from which applications are provided. The command enumerates the Win32 applications available for a platform.
Querying an Application Package	This activity is done on the WorkSpace On-Demand Domain Controller from which applications are provided. The command queries the details of a specific Win32 application available for a platform.
Modifying an Application Package	This activity is done on the WorkSpace On-Demand Domain Controller from which applications are provided. The command modifies selected details of a specific Win32 application available for a platform.

User Application Management includes the initial assignment of a restricted desktop to a user(s), and then all management of the applications assigned to a specified user's desktop. Whether the application is server-based or client-based, User Application Management allows the management of access to the application through control of the desktops that receive an icon for the application.

### 4.6.1 Adding an application package

This activity is done on the primary domain controller for the WorkSpace On-Demand domain from which the application is to be managed. After the package has been created on the sandbox machine, this command makes the application available for the system administrator to assign to users.

The application packages must be managed from a primary domain controller. Administrator authority is required for command execution. To add an application package, complete the following steps:

1. The administrator logs on to the domain controller using an account with system administrator authority.
2. The domain controller must have the WorkSpace On-Demand Windows Application Management feature installed. During the install of this feature, the Windows Application Datastore will have been created on the domain controller. The Windows Application Datastore is simply a file (appstore.ini) used to keep details about the Windows applications that exist on the domain controller. For applications that are server-based, the Windows Application Share(s) must have been created on a server in the domain by the system administrator (and an alias must exist for the share). The Windows shared directory is a standard Warp Server shared directory in which users have read-only access.
3. The administrator executes:

```
NETWIN APP appid /ADD /REMARK:"text"
```

The Windows Application Datastore is checked to determine if the package is currently on the system. If not already on the system, a new entry is added to the Windows Application Datastore. The command accesses the Application Package directory to retrieve and specify the APPDRIVE, APPDIR, and TYPE values for the application (these will exist in the appid subdirectory within the file appid.dat where appid is the unique name for the application). For server-based applications, the drive and alias of the location containing the read-only files (specified when the application was created using the `CRTPKG` utility) is specified in APPDRIVE and APPDIR. For applications that are client-based, the location of the executable files (drive and directory) must be specified in APPDRIVE and APPDIR.

### 4.6.2 Deleting an application package

This activity is done on the WorkSpace On-Demand Domain Controller from which the application is provided. This command will remove the package from the selected domain controller.

The application packages must be managed from a primary domain controller. Administrator authority will be required for command execution. To delete an application package, complete the following steps:

1. The administrator logs on to the domain controller using an account with system administrator authority.
2. The administrator executes:

```
NETWIN APP appid /DELETE
```

The Windows Application Datastore is checked to determine if the package is currently on the system. If it is on the system, all entries for the application are removed (from the Windows Application Datastore, Application Package directory, and Client Application Information directory).

3. When an application is deleted from the server domain, each user desktop of that application type (WIN9X or WINNT) is processed to remove the application from the desktop.

#### Removing Applications

As with WorkSpace On-Demand 2.0, the system administrator is responsible for the deletion of the application files on the servers.

### 4.6.3 Enumerating an application package

This activity is done on the WorkSpace On-Demand Domain Controller from which applications are provided. The command will enumerate the Windows applications available for a platform.

The application packages must be managed from a primary domain controller. Administrator authority will be required for command execution. To enumerate an application package, complete the following steps:

1. The administrator logs on to the domain controller using an account with system administrator authority.
2. The administrator executes:

```
NETWIN APP or NETWIN APP /TYPE:{WIN9X | WINNT}
```

When invoked with a `/TYPE` option, the applications for the specified type are enumerated. When invoked without a `/TYPE` option, all Windows applications are enumerated. The Win32 Application Datastore is read to determine the packages currently on the system for the platform type(s). The application information is enumerated for each application of the

correct type(s). The following information is enumerated for each: APPID, REMARK, and TYPE.

#### 4.6.4 Querying an application package

This activity is done on the WorkSpace On-Demand Domain Controller from which applications are provided. The command queries the details of a specific Windows application available for a platform.

The application packages must be managed from a primary domain controller. Administrator authority will be required for command execution. To query an application package, complete the following steps:

1. The administrator logs on to the domain controller using an account with system administrator authority.
2. The administrator executes the following command for the application desired:

```
NETWIN APP appid
```

The Windows Application Datastore is checked to determine if the application exists. If it exists, the full information held in the Datastore is displayed.

#### 4.6.5 Modifying an application package

This activity is done on the WorkSpace On-Demand Domain Controller from which applications are provided. The command modifies selected details of a specific Windows application available for a platform.

The application packages must be managed from a primary domain controller. Administrator authority is required for command execution. To modify an application package, complete the following steps:

1. The domain controller is powered on, and the administrator logs on using an account with system administrator authority.
2. The administrator executes:

```
NETWIN APP appid /REMARK:"text"
```

Since the `/REMARK` is the only value that can be modified for an application, that is the only parameter allowed by the command. The Windows Application Datastore is checked to determine if the specified application exists. If the application exists, the Windows Application Datastore entry for the application is updated with the changed `/REMARK` information.

---

## 4.7 Application commands

The two key application related commands and their syntax are discussed here: the `CRTPKG` and the `NETWIN APP` commands.

### 4.7.1 The `CRTPKG` command

The `CRTPKG` command is executed by the system administrator on the platform type the application is intended to be used on. The Windows system is a sandbox system created from a clean install on the system.

#### Command Syntax:

```
CRTPKG appid {/START | /FINISH} /APPDRIVE:drive /APPDIR:pathname
```

#### Options:

<b>appid</b>	Specifies the unique application ID for the application. Appid can be a maximum of 8 bytes in length. An appid cannot contain imbedded blanks or any of the following characters ' / \ [ ] :   < > + = ; , . ? *. appid is only allowed when the /FINISH switch is specified.
<b>/START</b>	Specifies that an initial snapshot of the sandbox system is taken. This is executed prior to the application installation on the sandbox system.
<b>/FINISH</b>	Specifies that a final snapshot of the sandbox system is taken. A DIFF operation is performed to determine changes that occurred due to application installation, and the appropriate files associated with the application package are transported to the primary domain controller.
<b>/APPDRIVE:drive</b>	Specifies the drive letter of the location where the application program will reside. For applications installed on a server, a <code>NET USE</code> to the location specified by the APPDRIVE and APPDIR values must be done prior to execution of this utility. /APPDRIVE is a required parameter when the /FINISH switch is specified.
<b>/APPDIR:pathname</b>	Specifies the fully-qualified path name where the application program will reside. If the application is to be installed on a server, a <code>NET USE</code> to the fully-qualified path name must be done prior to execution of this utility. Since an alias will be used at the server to ease assignment of the application access to users, the



fully-qualified path name used must be equal to the alias (the fully-qualified pathname specified by the alias).

**Note**

The alias at the server must have been created prior to execution of the CRTPKG utility. /APPDIR is a required parameter when the /FINISH switch is specified.

For Application Package Management and User Application Management, a set of files containing information on Windows registry changes, ini file changes, common folder changes, and local file changes is required. This set of files is stored in the Application Package directory and Client Application Information locations on the primary domain controller. The files associated with a particular application will be stored based on the appid, where appid is the unique application identifier defined during package creation.

The files created at the primary domain controller along with the location they will be stored at are:

PKG\_DIRvalue\WIN32APP\SERVER\appid\

**appiduser.inf or appiduser.reg** Contains the user registry changes.

**appid.dat** Contains the values specified for APPDRIVE, APPDIR, and TYPE.

**[Desktop]** Folder containing the icon(s) targeted for the user desktop.

**[Start Menu]** Folder containing the entry(s) targeted for the user Start menu.

**[Programs]** Folder containing the entry(s) targeted only for the user Programs menu.

PKG\_DIRvalue\WIN32APP\CLIENT\appid\

**appidsys.inf or appidsys.reg** Contains the system registry changes.

**appidiniX.inf** Contains the system ini file changes.

**local files** Contains the directory of local files installed for the application.

The PKG\_DIR value is a fully qualified pathname that will be retrieved from \BMLAN\DCDB\WIN32APP.INI.

To view how to build application packages, see Figure 23 on page 84.

#### Size Limitations on Windows 9x

Due to a limit of 64 K of information per INF file, there may be multiple INF files required for the registry and system ini information. Therefore, an X is part of the name of each INF file where X represents a number.

The `CRTPKG` command first takes a snapshot of a system prior to changes being made (such as installing applications). After changes are completed, it is then run again to capture the different of the old state against a new system state. The output is a package file containing a description of ini file changes, registry changes, and file changes that, when applied on another Windows 95/98 or Windows NT installation, duplicates the changes made to the system.

The utility creates a set of files for the registry: .REG files for Windows 9x and .INF files for Windows NT as well as INF file's for both operating systems. It also creates a directory structure of files on the client machine.

#### 4.7.2 The `NETWIN APP` command

The `NETWIN APP` command creates, deletes, changes, and displays information about Windows application definitions for a given domain.

The `NETWIN APP` command must be invoked at a primary domain controller.

When used without parameters, the `NETWIN APP` command enumerates all existing Windows applications and the applications' types. The following different options are available using the `NETWIN APP` command:

<b>appid</b>	Specifies the unique application ID for the application. Appid can be a maximum of 8 bytes in length. An appid cannot contain imbedded blanks or any of the following characters: ' / \ [ ] :   < > + = ; , . ? *
<b>/ADD</b>	Indicates that a Windows application definition will be created for the domain. An application package must have been previously created by the utility on the sandbox Windows system.
<b>/DELETE</b>	Indicates that a Windows application definition will be removed from the domain.
<b>/REMARK:"text"</b>	Specifies an optional comment that describes the application. "text" can be a maximum of 256 characters in length and must be enclosed within

double-quotation marks if it includes any spaces. This will be displayed when applications are enumerated. This option is not required. /REMARK is modifiable.

**/TYPE:{WIN9X | WINNT}** Indicates whether the application information requested is for a Windows 95, Windows NT, or Windows 98 desktop. /TYPE is optional and used to display the application definitions of a particular type. The /TYPE value cannot be modified.

The command processor is responsible for determining and providing the following values during the /ADD option:

- /APPDRIVE:drive

Specifies the drive letter where the application program resides. Together with the APPDIR value, the path to the application program will be specified. If the application is installed on a server, the APPDRIVE must be an available value that can be used for the logon assignment that is created when users are given access to the application. When a local path is used to install the application, the drive value represents the drive in which the application will reside on each client system. The command processor will access the appid.dat file within the PKG\_DIRvalue\WIN32APP\SERVER\appid\ directory and determine the APPDRIVE value. The PKG\_DIR value is a fully qualified pathname that will be retrieved from \BMLAN\DCDB\WIN32APP.INI.

- /APPDIR: {alias[\rempath] | localpath}

Specifies the directory where the application program resides. If the application is installed on a server, its location must be specified by an alias, and the server must be a member of the primary domain controller's domain. When a local path is specified, the maximum length is 256 characters. The command processor will access the appid.dat file within the PKG\_DIRvalue\WIN32APP\SERVER\appid\ directory and determine the APPDIR value. The PKG\_DIR value is a fully qualified pathname that will be retrieved from \BMLAN\DCDB\WIN32APP.INI:

- alias

Specifies an existing files alias definition pointing to a directory containing the application or to a directory containing the application in one of its subdirectories.

- rempath

Specifies the remaining path, beyond alias, to the subdirectory containing the application.

- `localpath`

Specifies a fully-qualified path to a directory that is local to each client system (requester). For example, if an application is defined with `/APPDIR:c:\windows\apps` (where `c:` is a local drive), the application must be installed in the `c:\windows\apps` directory of every client system that will use it.

- `/TYPE:{WIN9X | WINNT}`

Indicates whether the application is for a Windows 95/98 or Windows NT. `/TYPE` is a required parameter when `/ADD` is specified. The `/TYPE` value cannot be modified. The command processor will access the `appid.dat` file within the `PKG_DIRvalue\WIN32APP\SERVER\appid\` directory and determine the `TYPE` value. The `PKG_DIR` value is a fully qualified pathname that will be retrieved from `\BMLAN\DCDB\WIN32APP.INI`.

The `NETWIN APP` command must ensure that the following criteria are met before attempting any Windows application management:

- The server service is started.
- The local machine is the domain controller.
- A user is logged on with administrator privileges.

---

## 4.8 Application control files and data stores

There are many files that are stored in many locations to hold all the data that is used for both the user administration and the application distribution. This section lists those files, their purposes, and their locations (Figure 31 on page 130 also shows file locations):

- `PKG_DIR`

The `PKG_DIR` value is the directory and drive value you entered when you installed the Feature for Windows Clients software. Most of the Feature for Windows Clients files are stored relative to this location. During installation, you were prompted for the location of the Feature for Windows Clients database. This is the location. The location is stored in the `\BMLAN\DCDB\WIN32APP.INI` file. The default value for this variable is `x:\BMLAN\DCDB\`, where `x` is your boot drive. The following is a listing of the default `WIN32APP.INI`:

```
[Install]
PKG_DIR=C:\BMLAN\DCDB
```

- Windows Application Datastore

The Windows Application Datastore is located on the primary domain controller in the PKG\_DIR\WIN32APP\APPSTORE.INI file. This file contains entries for applications and this file is updated as application packages are added or deleted. The datastore also keeps application package details that are used when the applications are assigned or removed from users.

- User Datastore

The User Datastore is located on the primary domain controller in the PKG\_DIR\WIN32APP\USRSTORE.INI file. Records are created in the datastore for each user assigned a restricted desktop. An entry is added for each Win32 application assigned to the user's given desktop (with separate records kept for the Windows 9X desktop and the Windows NT desktop).

- Application Package Directory

The Application Package directory is located on the primary domain controller in the PKG\_DIR\WIN32APP\SERVER directory. The directory is created during the Feature for Windows Clients install process. Within this directory, a subdirectory is added when an application package is defined. This directory contains profile changes for the userid used at the sandbox.

- Client Application Information

The Client Application Information is located on the primary domain controller in the PKG\_DIR\WIN32APP\CLIENT directory. This directory is created during the Feature for Windows Clients install process. Within this directory, a subdirectory is added when an application package is defined. The subdirectory contains a number of files including the system registry and ini file changes related to the application and a directory of local file changes that were made during the installation of the application. For more information, see Table 10 on page 105.

- Windows Application Share or Local Drive

The Windows Application Share is located either at a server in the domain or on the local client system. This is the location in which the application files that are installed during application package creation are stored. This share/drive is *not* created during the install process. The system administrator is responsible for creation and maintenance of this location.

- Windows Registry Updates File

The Windows Registry Updates file is located in a user home directory on a server within the domain. This file is created or added to when an application is added on a specific user's desktop for a Windows platform type.

- Windows User Profile

The Windows user profile is located at a server in the domain. A default Windows user profile for Windows 95/98 and a default Windows user profile for Windows NT is added to the domain controller as part of the install process. These default user profiles define restricted (mandatory) desktops for users.

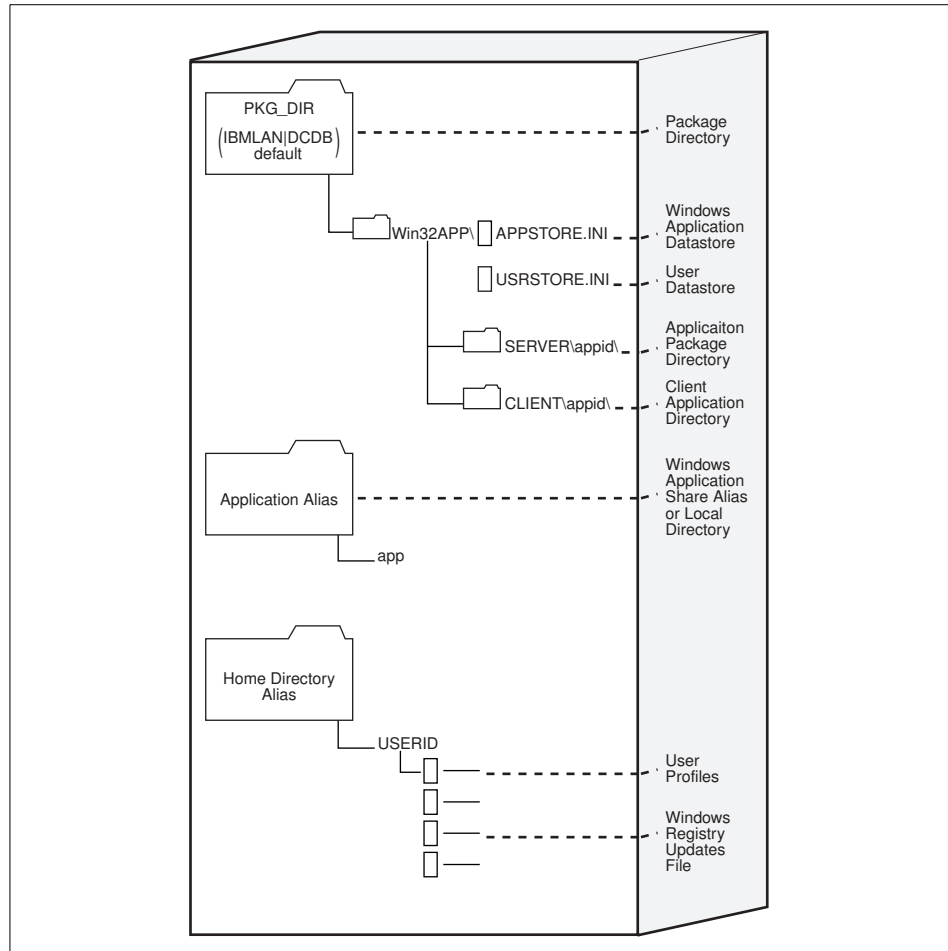


Figure 31. File locations

---

## Chapter 5. Workstation customization and administration

This chapter describes the client installation and configuration process. The command line syntax to define a workstation is described in some detail. This chapter also includes information on the client install process for Windows 9x and Windows NT Workstation. Various methods of adding hardware support are also discussed.

Information on some tools and utilities that aid the installation and configuration process is also addressed.

---

### 5.1 About defining workstations

The command used to define workstations is `NETWIN RIPLMACH`. You define the client by specifying values for the `NETWIN RIPLMACH` parameters. These parameters are then merged into the selected response file creating a unique response file for that particular workstation. This response file is stored in the `c:\IBMLAN\RPLUSER\workstation` directory.

During the workstation installation process, the response file is copied, together with the operating system image, to the workstation's local hard drive and used for the installation of the selected operating system.

The response file is selected using the `/RSP` parameter.

There are two ways to define a client: Using the command line to define each individual client or using the command line, together with a data file, to define multiple clients all at once.

The format of the command to define, change, or delete a workstation is as follows:

```
NETWIN RIPLMACH [machinename] [options]
```

Where `[machinename]` specifies the name of the client to be created, configured, deleted, or viewed. The name can be 1 to 8 bytes in length and cannot contain spaces or any of the following characters: " / \ [ ] : | < > + = ; , ? \*

The options ( `[options]` ) are as follows:

<b>/ADD</b>	Indicates that a client definition is created. If <code>/ADD</code> is specified without <code>/DATAFILE</code> , the
-------------	---

	following parameters must also be specified: /MAC, /RECORDID, /RELEASE, and /TYPE.
<b>/DELETE</b>	Indicates that a client definition is removed. /DELETE should not be specified with any other parameters.
<b>/CDKEY{:cdkey}</b>	Specifies the product ID for the operating system that is installed. If /CDKEY is not specified when a client definition is created, the installation process for the client prompts for a product ID. A CDKEY value is found on the CD-ROM sleeve or in the manual provided with the operating system you installed. It may also be provided as part of a license pack.
<b>/CONFIGATLOGON{:Y   N}</b>	Specifies if video settings are configurable at logon. The default value is N (video settings are <i>not</i> configurable at logon).
<b>/DATAFILE{:FileName}</b>	Indicates that configuration information for one or more client definitions is contained in the file specified by FileName. For information about the format of this file, please refer to the Administrator's Guide for this product.
<b>/DHCP{:Y   N}</b>	Indicates if the client should be configured for Dynamic Host Configuration Protocol.
<b>/ENABLED{:Y   N}</b>	Indicates if the client definition is enabled or disabled. The default value is Y (enabled).
<b>/HELP</b>	Displays task-oriented syntax for this command. /HELP should not be specified with any other parameters.
<b>/IMAGE{:ImageFileName}</b>	Specifies the name of the image definition that is used to initially start the client before the operating system installation process begins. Valid values are the names of any .IMG files found in the \IBMLAN\DCDB\IMAGES directory. The default value is W32DOSSB.IMG.
<b>/IP{:TCPIPAddress}</b>	Specifies the TCP/IP address of the client. The TCPIPAddress must use the standard TCP/IP dotted decimal address format, nnn.nnn.nnn.nnn. When /IP is specified with no value, TCP/IP configuration is removed from the



client definition. If /DHCP:Y is specified, the value for /IP is ignored.

- /MAC{:AdapterAddress}** Specifies the MAC address of the client. The MAC address consists of twelve hexadecimal characters. This parameter is required when creating a client definition.
- /NIP{:NameServerTCPIPAddress[,NameServerTCPIPAddress]}** Specifies the TCP/IP addresses of TCP domain servers (nameservers). Up to three TCP/IP addresses, separated by commas, may be specified with the /NIP parameter. If /DHCP:Y is specified, the value for /NIP is ignored.
- /OPTIONS** Displays detailed information on command parameters. /OPTIONS should not be specified with any other parameters.
- /PARTITION{:PrimaryPartitionSize | ALL}** Specifies the size of the primary disk partition on the client. The default value is 250 MB. The maximum value is 2 GB. The minimum value for all client types is 250 MB. The value may be specified in terms of megabytes (MB) or gigabytes (GB). If no size designation is specified, MB is assumed. Specify /PARTITION:ALL if you want the entire disk to be the client's primary disk partition.
- /PRINTER{: "PrinterName,DriverModel,Port" |PrinterName=delete}** Specifies a printer used by a Microsoft Windows 95 or Microsoft Windows 98 client. This parameter is not valid for Microsoft Windows NT clients. PrinterName specifies the queue name of the printer as it is known on the client. This value cannot be longer than 31 bytes and cannot contain the following characters: \ , ; = - DriverModel indicates the device driver associated with the printer. Port indicates one of the following: the port (LPT1-LPT9) to which the printer is physically attached if PrinterName specifies a local printer or a UNC name indicating a shared network printer. To disassociate a printer with a client definition, specify the name of the printer followed by

=delete. For example:  
/PRINTER:printer1=delete

**/RECORDID{:ServerRecordID}**

Specifies the name of the identifier of the server record that services the client's RIPL request when the client is started. Valid values can be found in the ServerRecord section of RPL.MAP, as the twelfth field per line. This parameter must be specified when creating a client definition.

**/REGUSER{"RegisteredUser"}**

Specifies the name of the user (such as "John Doe") who is registered to use the client's copy of the operating system. If /REGUSER is not specified when creating a client definition, the installation process for the client prompts the user for this information. The value specified with /REGUSER can be a maximum of 256 bytes in length and must be enclosed in double-quotation marks if it includes any spaces.

**/RELEASE{:DirectoryName}**

Specifies the version of the operating system installed on the client. This parameter is required when creating a client definition. Valid values for this parameter are the names of any subdirectory of the \BMLAN\RPL directory.

**/REMARK[:"text"]**

Specifies an optional comment that describes the client definition. The "text" can be a maximum of 48 bytes in length and must be enclosed within double-quotation marks if any spaces are included. When /REMARK is specified without a value, the existing remark (if any) is deleted.

**/RESET**

Indicates that the specified client has its operating system re-installed the next time it is restarted. Many modifications made to the client require the operating system to be reinstalled. Make these modifications to the client definition before the `NETWIN RIPLMACH machinename /RESET` command is issued. Do not specify /RESET with /ADD or /DELETE.

**/RESOLUTION{:VideoResolution}**

Specifies the video resolution for the client. VideoResolution must be specified in the format hhhXvvvXccc@rrr where: hhh is the horizontal resolution, vv is the vertical resolution, and ccc is the number of colors supported. This value is converted to a bit depth value. rrr is the vertical refresh rate.

**/RIP{:RouterTCPIPAAddress}**

Specifies the TCP/IP address of the TCP/IP Router. RouterTCPIPAAddress must use the standard TCP/IP dotted decimal address format, nnn.nnn.nnn.nnn. If /DHCP:Y is specified, /RIP is ignored.

**/RSP:ResponseFileName**

Indicates the name of a user-supplied response file that is used to install the selected operating system on a particular machine. When /RSP is specified, ResponseFileName is used instead of the default response file for the operating system. ResponseFileName is ignored if /ADD is not specified.

**/SANDBOX**

Indicates that the client is configured to be a sandbox. This means that the client will have an unrestricted Windows desktop and can be used for application package creation. When /SANDBOX is specified, /ADD must also be specified.

**/SUBM{:SubnetMask}**

Specifies the subnet mask. 'SubnetMask' must use the standard TCP/IP dotted decimal address format, nnn.nnn.nnn.nnn. If /DHCP:Y is specified, /SUBM is ignored.

**/TCPDN{:TCPIPDomainName}**

Specifies the TCP/IP domain name, for example, companyname.com.

**/TCPNAME{:TCPIPHostName}**

Specifies the TCP/IP host name of the Windows 95 client or Windows 98 client. /TCPNAME is ignored if specified for a Windows NT client. If /DHCP:Y is specified, /TCPNAME is ignored.

**/TYPE{:WIN95 | WIN98 | WINNT}**

Specifies the operating system that is installed on the client. This parameter is required when creating a client definition.

### 5.1.1 Defining workstations individually

**Note: BIOS**

For a number of reasons, before you install a Windows operating system on a client, verify that the BIOS is 4/15/98 or later. Check your PC vendor's documentation on how to do this.

Usually, when you create a single client for testing, or ad hoc client definition, you would probably want to use the command line. However, when you are defining multiple clients, it is best to use the data file option. Using data files is explained later in this chapter.

When you define a client using the command line, you need to specify at least the following parameters together with the clients name:

```
/ADD  
/TYPE{:WIN95 | WIN98 | WINNT}  
/RECORDID{:ServerRecordID  
/RELEASE{:DirectoryName}  
/MAC{:AdapterAddress}
```

If you do not specify a response file, it uses the default response files to define the client. Table 12 lists the default response files for each of the operating systems.

Table 12. Directory path for default response file

Operating system	Directory path and default response file
Windows NT Client Feature	IBMLAN\RPL\NT40\RSP\UNATTEND.TXT
Windows 95 Client Feature	IBMLAN\RPL\W95\RSP\MSBATCH.INF
Windows 98 Client Feature	IBMLAN\RPL\W98\RSP\MSBATCH.INF

For example, if you use the command line:

```
netwin ripmach sample01 /add /mac:8884ac7763ec /release:w98 /type:win98  
/dhcp:n /enabled:y /image:w32dossb.img /recordid:r_dtk_ndis /partition:600  
/cdkey:010-1578717 /REGUSER:"ITSO User" /rsp:c:\rsp\ibm300gl.inf
```

The above command line produces the following unattend.txt file in the c:\ibmlan\rpluser\sample01 directory:

```
[Setup]
Express=1
InstallDir="c:\windows"
InstallType=1
EBD=0
ChangeDir=0
OptionalComponents=1
CleanBoot=1
CCP=0
DevicePath=0
NoDirWarn=1
TimeZone="Central" ;Central Timezone
USA and Canada
Uninstall=0
NoPrompt2Boot=1
ShowEula=0
ProductKey="010-1578717"

[System]
Locale=L0409
Display=Driver_65.Install, PCI\VEN_1013&DEV_00D6 ;CIRRUS LOGIC
VIDEO
DisplChar=16,800,600

[CirrusVideo] ;CIRRUS LOGIC VIDEO
546X.INF

[DestinationDirs] ;CIRRUS LOGIC VIDEO
CirrusVideo=10,INF

[SourceDisksNames] ;CIRRUS LOGIC VIDEO
24,"Cirrus",CirrusVideo,0

[SourceDisksFiles] ;CIRRUS LOGIC VIDEO
546X.INF=24

[NameAndOrg]
Name="ITSO User"
Org=""
Display=0

[InstallLocationsMRU]
c:\windows=mrul
```

```

[OptionalComponents]
"Paint"=0
"Communications"=0
"Disk compression tools"=1
"Accessibility Tools"=1
"Audio Compression"=0
"Video Compression"=0
"Sound Recorder"=0
"Volume Control"=0
"Media Player"=0
"Personal Web Server"=0
"Document Templates"=1
"Dial-Up Networking"=0
"Quick View"=0
"Internet Tools"=0
"Phone Dialer"=0
"Microsoft Outlook Express"=0
"Microsoft FrontPage Express"=0
"Imaging"=0
"Multimedia"=0
"CD Player"=0
"Microsoft NetMeeting"=0
"Online Services"=0
"America Online"=0
"AT&T WorldNet Service"=0
"CompuServe"=0
"Prodigy Internet"=0
"The Microsoft Network"=0
"Macromedia Shockwave Director"=0
"Macromedia Shockwave Flash"=0
"Microsoft VRML 2.0 Viewer"=1

[Network]
Clients=VRedir
IgnoreDetectedNetCards=0
Protocols=NETBEUI
ComputerName=SAMPLE01
Workgroup=Workgroup
Security=SHARE
Display=0
ValidateNetCardResources=0

[MSTCP]

[VRedir]
LogonDomain="GA_DOM"

```

```

[Printers]

[Install]
AddReg=LogonClnt.Reg
DelReg=

.Welcome
Copyfiles=CirrusVideo                                ;CIRRUS LOGIC VIDEO

[Registry.Welcome]
HKLM, SOFTWARE\Microsoft\Windows\CurrentVersion\Run, Welcome, ,

[LOGONCLT.REG]
HKLM, "Software\Microsoft\Windows\CurrentVersion\Network\Real Mode
Net", AutoLogon, 1, 0
;HKLM, "Software\Microsoft\Windows\CurrentVersion\RunOnce", TMA, , "c:\$TIVOLI
\TIVOLI.bat"
;This is the 300gl file I specified and moved...
HKLM, "Software\Microsoft\Windows\CurrentVersion\RunOnce", JVM, , "c:\$IBM JVM\
IBM JVM.bat"
HKLM, "Software\Microsoft\Windows\CurrentVersion\RunOnce", Logonclnt, , "C:\LOG
ON\LOGONCLT.BAT"

[Strings]

```

### 5.1.2 Defining multiple workstations

You can use a data file to define multiple clients. A data file stores all of the client information in one file that you specify using the `NETWIN RIPLMACH` command to set all your network clients.

#### Data File

We strongly recommend that the administrator uses a data file when creating clients. Data files give you the ability of deploying a client operating system to many clients without too much rekeying.

To define multiple clients using a data file, complete the following steps:

1. Create a data file. The data file format:
  - Include an entry for each client to be created or modified.
  - Enclose each client name within square brackets.
  - Specify only one command line parameter per line.
  - Enclose within double quotation marks all parameter values that contain spaces.

- Parameter values may not contain a forward slash (/) or square bracket characters ([]).

The following is an example of using a data file to create two clients:

```
[client1]
/IMAGE:W32DOSSB.IMG
/MAC:123456789012
/TYPE:WIN95
/RECORDID:R_DTK_NDIS
/RELEASE:W95
*optional parameters*
[client2]
/IMAGE:VERS2.IMG
/MAC:212345678901
/TYPE:WINNT
/RECORDID:R_DTK_NDIS
/RELEASE:NT40
/RSP:GROUP1.TXT
```

2. To run the data file and define multiple clients, type:

```
NETWIN RIPLMACH /ADD /DATAFILE:path_to_file
```

When specified together, /ADD and /DATAFILE allow no other parameters. All other parameters must be specified in the data file. For more information, see Section 5.1, “About defining workstations” on page 131.

3. Turn on the client and allow it to install the operating system on the client.

---

## 5.2 The install scripts

The install process for both Windows 9x and Windows NT make use of the Microsoft unattended install process. These processes are not identical in both the operating systems. Both operating system use a response file that looks similar, but differs greatly in most respects.

Figure 32 on page 141 show the install process that is used for the different operating system setup processes.



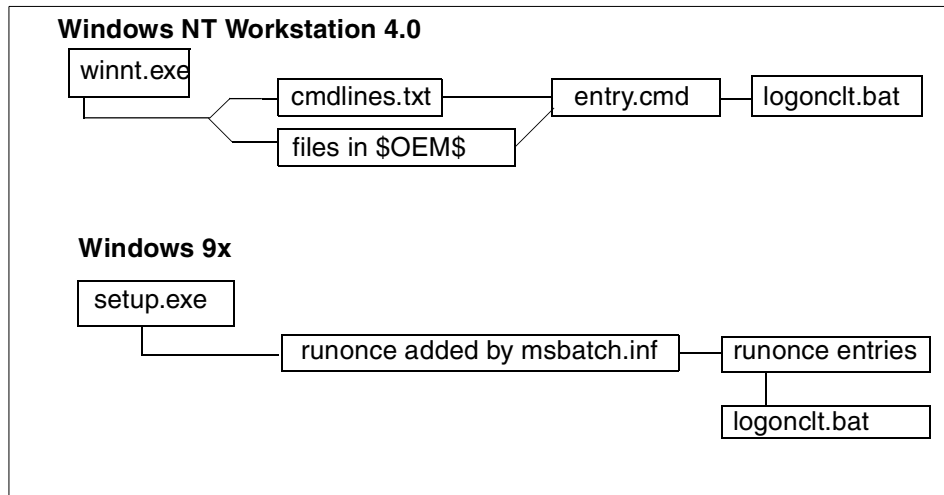


Figure 32. Installation and configuration process for Windows client setup.

#### Editing response files

Do not use an OS/2 editor on the server to modify any of the client response files, such as `MSBATCH.INF` or `UNATTEND.TXT`. The code pages for the OS/2 and Windows operating systems are not compatible. Language-specific characters (such as accented characters) that display in the client response files do not display correctly when they are created with an OS/2 editor. To update any of the response files, open the response file on a Windows system by using a Windows editor to edit and save the response file. Although, the language-specific characters might not display as expected when you view them on the OS/2 server, they will function correctly on the client.

### 5.2.1 Windows NT Workstation install files

Windows NT Workstation uses the `winnt` command. This command uses the network connection to access the installation files on the boot server. The `winnt` command copies all the files needed to complete the installation over the network into a temporary directory and then continues by performing the installation from the local hard disk, first going through the text mode setup and then GUI mode setup.

The syntax of the `winnt` command is as follows:

```
winnt [/s:sourcepath] [/i:inf_file] [/t:drive_letter] [/x] [/b] [/o[x]]
[/u:answer_file] [/udf:id, [UDF_file]]
```

Table 13 describes the parameters of the `winnt` command.

Table 13. The `winnt` parameters

Parameter	Description
/s:sourcepath	Specifies the location of the Windows NT files.
/i:inf_file	Specifies the file name (no path) of the setup information file. The default is DOSNET.INF.
/t:drive_letter	Forces setup to place temporary files on the specified drive.
/x	Prevents setup from creating setup boot floppies. Use this when you already have setup boot floppies (from your administrator, for example).
/b	Causes the boot files to be loaded on the system's hard drive rather than on floppy disks so that floppy disks do not need to be loaded or removed by the user.
/o	Specifies that setup only create boot floppies.
/ox	Specifies that setup create boot floppies for CD-ROM or floppy-based installation.
/u:answer_file	Specifies the location of an answer file that provides answers the user would otherwise be prompted for during setup.
/udf:id [,UDF_file]	Specifies the identifier that is to be used by the setup program to apply sections of the UDF_file in place of the same section in the answer file. If no UDF is specified, the setup program prompts the user to insert a disk that contains a file called \$UNIQUE\$.UDF. If a UDF is specified, setup looks for the identifier in that file.

Because of the way WorkSpace operates, many of the command line options are not used. The command issued by a WorkSpace On-Demand client provides the response file (answer file) and the location of the source:

```
winnt.exe /u:c:\IBMWIN32\Uattend.txt /s:x:\nt40\i386
```

#### 5.2.1.1 Unattend.txt

The unattend.txt response file or answer file is a file containing the information required to automate the install process. The format of the file consists of section headers, parameters, and values for those parameters. The section headers are predefined and some may be user-defined. It is not necessary to specify all the parameters and keys. The file format is as follows:

```
[section1]
; Section contains keys and the corresponding
; values for those keys/parameters.
; keys and values are separated by "=" signs
; Values usually require double quotes "" around them
;
key = value
.
.
[section2]
key = value
.
```

Information on the sections and keys can be found in the Microsoft Resource Kit or at:

<http://support.microsoft.com/support/kb/articles/Q155/1/97.asp>

#### 5.2.1.2 Copying the \$OEM\$ directory and its subdirectories

Once the operating system is installed, any additional components, files, or applications that are not included with the Windows NT Workstation retail product that you wish to have with the product must be added to subdirectories of the \$OEM\$ directory on the server.

The \$OEM\$ directory includes the following subdirectories:

- \$OEM\$\Textmode

The \$OEM\$\Textmode directory contains all of the files that the setup program needs to load and that text mode setup needs to copy to the target computer so that the system can boot into GUI setup.

If you are installing SCSI, keyboard, video, or pointer device drivers, or HALs, that are not included or have been updated since the release of the

Windows NT Workstation 4.0 retail version, this directory should also contain a `txtsetup.oem` file. This file contains pointers to all the files required by the setup program and the text mode setup to load and install these components. `Txtsetup.oem` and all files listed in it must also be listed in the `[OEMBootFiles]` section of the answer file.

- `$OEM$\$$`

This directory contains system files (either new files or replacements to files included in the retail product) that need to be copied to the various subdirectories in the Windows NT system directory. You need to maintain the directory structure used in the retail product and place in each subdirectory the files that need to be copied to the corresponding system directory on the destination computer. If some of the files use long filenames, add the file `$OEM$\$$\$$Rename.txt`. This file lists all files that have long names and their corresponding short names.

- `$OEM$\NET`

This directory contains only subdirectories. You need to place in each subdirectory the files for a particular network component, such as network cards, network services, and network protocols. Files in this directory are used by the network setup module.

- `$OEM$\drive_letter`

The `$OEM$\drive_letter` directory holds files that are to be copied by the setup program to corresponding drives on the destination computers. For example, files in the `$OEM$\C` directory will be copied to drive C on the destination computer during text mode setup.

Place files for applications that you want to install along with the Windows NT Workstation in subdirectories of the `$OEM$\drive_letter` directories on the distribution sharepoint. The files in these subdirectories must have short filenames. To rename them with long filenames after they have been copied to the destination computer, list them in a file named `$$rename.txt` in the same directory.

Applications included in the `$OEM$\drive_letter` directories must support a scripted (silent) installation; to include applications that require interactive installation, use the `sysdiff` utility.

Setup uses commands specified in the `$OEM$\Cmdlines.txt` file to install applications or copy files that are available in subdirectories of `$OEM$\drive_letter` on the distribution sharepoint. You can use these commands to invoke a Win95-style INF file, run the `sysdiff` command, or perform other actions.

For example, to install MS Office on the C drive, in the \MSOffice directory, place the files in \$OEM\$\C\MSOffice. Then specify the command to install the application in the \$OEM\$\Cmdlines.txt file.

You can also place files that you only want copied (but not installed) in subdirectories of \$OEM\$\drive\_letter. For example, if you have written help files that cover policies and procedures used in your organization, you can place them in a subdirectory of \$OEM\$\C. Then use a command in the \$OEM\$\cmdlines.txt file to copy them to the destination computers.

#### 5.2.1.3 The cmdlines.txt and other files

Once the `winnt` command completes, and the files are copied across, the system processes the `cmdlines.txt` file. You can install files located in the subdirectories of \$OEM\$ by listing the installation commands in a text file named `cmdlines.txt`. The `cmdlines.txt` file is located in the `x:\IBMLAN\rp\nt40\i386\`\$OEM\$ directory. All workstations share the same `cmdlines.txt` file. The syntax is as follows:

```
[Commands]
"command 1"
"command 2"
"command 3"
```

Enter the entire command line in double quotation marks. The following is an example of a `cmdlines.txt` file for Windows NT Workstation for WorkSpace On-Demand.

```
[Commands]
".\service\sp4i386.exe -U -Z"
"C:\logon\entry.cmd"
"C:\$IBM JVM\SETJVM.cmd"
"C:\logon\setlogon.cmd"
```

The `cmdlines.txt` file has all the comments removed. The file first installs Service Pack 4 and then runs `entry.cmd`, installs the JVM and runs another command file `setlogon.cmd`.

#### ***entry.cmd***

The `entry.cmd` file has only one line:

```
RUNDLL32 setupapi,InstallHinfSection DefaultInstall 128 c:\logon\ntclean.inf
```

This runs `ntclean.inf`, which is listed below:

```

[version]
signature="$Windows NT$"
[DefaultInstall]
AddReg=InstallRunOnce
[InstallRunOnce]
;KLM,%RunOnceKey%,%RunOnceEvent%,,%RunOnceProg%
HKLM,%AutoLogon%,%AutoAdmin%,,%Val1%
HKLM,%AutoLogon%,%Pass%,,%Val2%

[DeleteAutoAdminLogon]
DelReg=DelAutoLogon
[DelAutoLogon]
HKLM,%AutoLogon%,%AutoAdmin%

[Strings]
RunOnceKey="SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce"
RunOnceEvent="LogonClt"
RunOnceProg="c:\logon\logonclt.bat"
AutoLogon="SOFTWARE\Microsoft\Windows NT\CurrentVersion\WinLogon"
AutoAdmin="AutoAdminLogon"
Val1="1"
Pass="DefaultPassword"
Val2=""

```

ntclean.inf adds registry entries that allow the workstation to autologon and then run the batch file logonclt.bat.

### ***logonclt.bat and setlogon.cmd***

```

rem =====
rem This file contains setup commands that are executed on
rem Windows NT clients when the client image is installed. Do
rem not change this file.
rem =====
@echo off
CD \LOGON
md c:\temp 2>nul
RUNDLL32 setupapi,InstallHinfSection DeleteAutoAdminLogon 128 c:\logon\ntclean.i
nf
call misc.bat
if not "%miscres%"=="2" del misc.bat
start /wait prninst.exe -Ic:\ibmwin32\unattend.txt
set cleanupf=%cleanup%
start /wait SETUP domain=%DOMAIN% sandbox=%SANDBOX% -s
chkini /ini=setup.log /section=ResponseResult /key=ResultCode /del=300 /errorlev
el /val=0
if not errorlevel 0 goto leave_files
:Success
if not '%cleanupf%'=='Yes' goto leave_files
chkini /ini=c:\$ibmjvm\setup.log /section=ResponseResult /key=ResultCode /del=2
/val=0 /errorlevel
if not errorlevel 0 goto DelIBMWin32

```

```

deltree /Y c:\$ibmjvm
:DelIBMWin32
deltree /Y c:\ibmwin32
copy dellogon.bat c:\temp
copy deltree.exe c:\temp
CD \temp
dellogon
:leave_files
if exist C:\WINNT\system32\wshutdwn.exe c:\winnt\system32\wshutdwn /Q /R
exit

```

The file setlogon.cmd uses the regedit utility to import a text based file called a registration file, or .REG file. The file setlogon.cmd contains just one line:

```
\WINNT\REGEDIT /s c:\logon\logonclt.reg
```

The contents of the logonclt.reg file is as follows:

```

REGEDIT4

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce]
"LogonClt"="C:\\LOGON\\LOGONCLT.BAT"
[HKEY_CURRENT_USER\Control Panel\Desktop]
"Wallpaper"="wsoddtbg.bmp"
[HKEY_CURRENT_USER\Control Panel\Colors]
"Background"="0 0 128"

```

Essentially, the logonclt.bat first installs any printers using the prininst utility and then cleans up the system, deleting any unnecessary install files.

The setlogon file changes the background bitmap with the WorkSpace On-Demand bitmap and resets the system.

### 5.2.2 The \$\$Rename.txt files

To map short filenames used in subdirectories of \$OEM\$ to long filenames that should be assigned to those files after they are copied to the destination computers, list them in a \$\$Rename.txt file. Each subdirectory of \$OEM\$ requires its own \$\$Rename.txt file if the files in that directory need to be assigned long filenames on the destination computer. For example, if some of the system files in \$OEM\$\\$\$ use long filenames, they must be mapped in the file \$OEM\$\\$\$\\$\$Rename.txt. If some of the application files in \$OEM\$\C use long filenames, they must be mapped in the file \$OEM\$\C\\$\$Rename.txt. The format for the \$\$Rename.txt file is as follows:

```

[section name]
short name 1 = "long name 1"
short name 2 = "long name 2"

```

Table 14 describes the parameters.

Table 14. The rename.txt file description

Parameter	Description
section name	This parameter is the path to the directory that contains the files. To indicate that the section contains the names of files or subdirectories that are on the root of the drive, use a backslash [\] as the section name.
short name	This parameter is the name of the file or subdirectory in this directory to be renamed.
"long name"	This parameter is the new name of the file or subdirectory. This name must be in double quotes.

### 5.2.3 Windows 9x installation flow

Windows 9x works slightly differently in that it will copy all files into a temporary directory and then run setup.exe to start the installation locally. The copying of the files to the local drive is performed by the WorkSpace On-Demand installation process and not by the operating system install process as is the case with Windows NT Workstation.

The setup command that is run is:

```
setup.exe /is /iw MsBatch.inf
```

It is not very important to know all the switches. However, they are described in Table 15.

Some of the switches can be used with both Windows 95 and Windows 98, and some can be used only with Windows 98.

Table 15. Windows 9x setup switches

Switch	Description
<b>For Windows 98 Only</b>	
/m	This switch bypasses the playing of the setup sound (.wav) files.



Switch	Description
/na	This switch bypasses the program check and can use the following values: 0: default 1: No Windows-based program check, but MS-DOS-based program are blocked. 2: No MS-DOS-based program check, but Windows-based programs are blocked. 3: No Windows-based or MS-DOS-based program check.
/nd	This switch ignores the presence of a Migration.dll file and is used to force Windows 98 to overwrite newer files. NOTE: Files that use the ".,,32" flag in the .inf file still force Windows 98 Setup to keep the newer files.
/nf	Do not prompt to remove the floppy disk from the drive (for bootable CD-ROMs), same as if there is a file named BOOTCD in the cabinet folder.  or if there is a "BootCD=1" line in the Msbatch.inf file.
/nh	This switch bypasses running the Hwinfo.exe program at zero percent files and RunOnce.
/nx	Do not check the version of setup that is running.
/ie	This switch bypasses the Windows 98 Startup Disk Wizard screens. If this switch is used, the Windows\Command\EBD folder is not created.
/iv	This switch bypasses displaying the Setup screens during an upgrade within Windows.
<b>Windows 98 and Windows 95</b>	
/?	This switch provides a brief summary of the available setup switches and the correct command-line syntax to use them.
/c	This switch bypasses running SMARTDrive.
/d	This switch bypasses using your existing Windows configuration (such as your current Win.ini and System.ini files).
/l	Use this switch if you have a Logitech mouse and want it enabled during setup.
/n	This switch causes Setup to run without a mouse.
-s	Use this switch to use an alternate Setup.inf file.
/t:<dir>	This switch lets you specify where Setup copies its temporary files. <i>Note:</i> Any existing files in this folder are deleted.

Switch	Description
/ig	Allows setup to run on some older Gateway and Micron computers with an early BIOS.
/ih	This switch causes setup to run ScanDisk in the foreground.
/im	Causes setup to ignore the conventional memory check.
/iq	If you use the /is switch to bypass ScanDisk, or ScanDisk fails, setup checks your drive for cross-linked files. The /iq switch prevents setup from doing this.
/is	This switch causes setup not to run ScanDisk.
/iw	This switch causes setup to bypass licensing.
/it	This switch bypasses checking for the presence of <i>dirty</i> or <i>deadly</i> terminate-and-stay-resident programs (TSRs) that are known to cause problems with Windows setup.
/p	This switch causes setup to pass string(s) directly to Detection Manager (or Sysdetmg.dll). Setup does not interpret the content of the string. The string can contain one or more detection options.

The /p switch is not used by itself. The string can contain one or more detection switches separated by a semicolon (;). For example, if you want to use /p f and /p i, you type: `setup /p f;i`

Some switches are simply on/off switches. The absence of the switch implies Off; the presence of the switch turns it on. A minus sign (-) appended immediately after a switch turns it off.

Some switches take parameters in the form of <c>=<params>. If there is more than one parameter to a switch, the parameters are separated by a comma (.). There must not be any spaces in the detection option string.

Valid detection switches are available from the Microsoft Web site.

#### 5.2.3.1 The msbatch.inf file

Once the operating systems are installed, additional components, such as service packs, the TMA client, and other software, can be installed. For the different versions of Windows, they are started from different places.

For Windows 9X type clients, it is best to start the install by adding a line to the msbatch.inf file that contains all the install responses.

The msbatch.inf file is the response file for Windows 9x type clients. This is the equivalent of the unattend.txt file for Windows NT Workstation. Unlike the Windows NT Workstation install and configuration process, Windows 9x does not have as many additional files.

A part of the msbatch.inf file is reproduced below. The install section shows the additional items to be executed after setup. In the figure, the product registration and the logonclt.reg section are going to run after install. The logonclt.reg section has a number of RunOnce items. First, it does an automatic logon, installs the Tivoli client, then installs the JVM, and finally runs a file called LOGONCLT.BAT.

```
:
:
[Install]
AddReg=ProductID.Reg,LogonClnt.Reg
DelReg=Registry.Welcome

[Registry.Welcome]
HKLM,SOFTWARE\Microsoft\Windows\CurrentVersion\Run,Welcome,,

[PRODUCTID.REG]
HKLM,"Software\Microsoft\Windows\CurrentVersion","ProductId",,%ProductID%

[LOGONCLT.REG]
HKLM,"Software\Microsoft\Windows\CurrentVersion\Network\Real Mode Net",AutoLogon,1,0
HKLM,"Software\Microsoft\Windows\CurrentVersion\RunOnce",TMA,,"c:\$TIVOLI\TIVOLI.bat"
HKLM,"Software\Microsoft\Windows\CurrentVersion\RunOnce",JVM,,"c:\$IBM JVM\IBM JVM.bat"
HKLM,"Software\Microsoft\Windows\CurrentVersion\RunOnce",Logonclnt,,"C:\LOGON\LOGONCLT.BAT"
:
:
:
```

### **LOGONCLT.BAT**

The logonclt.bat file is the file that runs last on the client before configuration is complete. Below is an extract of the logonclt.bat file for a standard Windows 95 client. Some of the remarks have been taken out and the lines are wrapped.

```
@echo off
rem =====
rem This file contains commands that are executed on
rem Windows 95 clients when the client image is installed.
rem =====
rem In this section, place commands to rename files that are in use
rem when XCOPY executes. rem attrib c:\windows\tasks\sa.dat -h
rem attrib c:\windows\help\windows.gid -h
rem attrib c:\windows\ttfcache -h
```

```

rem rename c:\windows\tasks\sa.dat sa.bak
rem rename c:\windows\help\windows.gid windows.bak
rem rename c:\windows\ttfcache ttfcache.bak

rem =====
rem In this section, place a command to XCOPY application package files
rem from a client's $WINTMP directory to a client's WINDOWS directory
rem
rem xcopy C:\$WINTMP\*. * C:\WINDOWS /s /e /h /k > c:\xcopy.log
rem =====
rem This section contains the log on executable .INF file for Windows 95
rem clients.
rem Do not change this section.

rundll setupx.dll,InstallHinfSection DefaultInstall 132
c:\logon\autooff.inf

rem =====
rem In this section place commands to run any .INF files contained
rem in Windows 95 or Windows 98 application packages.
rem rundll32 setupx.dll,InstallHinfSection DefaultInstall 128
c:\appidini1.inf

rem =====
rem This this section, place a command to run the DELICN95.INF file. This
rem file
rem removes registry entries for icons on Windows 95 clients.

rundll32 setupx.dll,InstallHinfSection DefaultInstall 128
c:\logon\delicn95.inf

rem =====
rem In this section place commands to run any .REG files contained
rem in Windows 95 or Windows 98 application packages.
rem start /wait regedit /s c:\appidsys.reg
rem cd\

rem =====
rem In this section place a command to run the RENFILE.BAT program that
rem is contained in Windows 95 and Windows 98 application packages.
rem This program runs on Windows 95 clients.

rem The RENFILE.BAT program contains the long file names for the files
rem in your application package that were given short names when
rem the package was created. When this program runs on a client, the
rem files listed in the program are renamed to their original long file
rem names.

```

```

rem call c:\renfile.bat

rem =====
rem This section includes set up commands for Windows 95 clients.
rem Do not change this section.

set cleanupf=%cleanup%
CD \LOGON
setkeys
chkini /ini=c:\logon\setup.iss /section=SdFinishReboot-0 /key=BootOption
/del=0 /val=3 /wr
if not '%cleanupf%'=='Yes' goto CallSetup
chkini /ini=c:\$ibmjvm\setup.log /section=ResponseResult /key=ResultCode
/del=1 /val=0 /errorlevel
if not errorlevel 0 goto DelIBMWin32
deltree /Y c:\$ibmjvm
:DelIBMWin32
deltree /Y c:\ibmwin32
:CallSetup
start SETUP domain=%DOMAIN% sandbox=%SANDBOX% -s
dosexit

```

This file is used to clean up the code on the client if requested to do so. The details on how to change this is described below. Other processes can also be added to this file for customization.

#### 5.2.4 Operating system files on the client

When you install a client image on a client, files that are used to install the client image remain on the hard disk of that client. To remove those files, do the following before you install the operating system:

1. Log on to the boot server as an administrator.
2. Change to the \IBMLAN\RPL\W95\LOGON directory for Windows 95 clients, to the \IBMLAN\RPL\W98\LOGON directory for Windows 98 clients, or to the \IBMLAN\RPL\NT40\LOGON directory for Windows NT clients.
3. From any ASCII text editor, open the LOGONCLT.BAT file.
4. At the bottom of the file, find the following entry:

```
Set cleanupf=%cleanup%
```

Change the entry to:

```
Set cleanupf=Yes
```

You can choose to save the files that are used to install the client image on one client and delete the files from the rest of the clients. To save those files on only one client:

1. Log on the boot server as an administrator.
2. Change to the \BMLAN\RPLUSER\client\_name directory (client\_name represents the client where the setup files will remain).
3. Open the STATE.FIL file on the client where the setup files will reside.
4. Find one of the following entries:

```
DeleteSetupFiles=DefaultNo  
DeleteSetupFiles=Yes
```

Change the entry to:

```
DeleteSetupFiles=No
```

---

### 5.3 Summary chart

Table 16 is a summary table of the steps and files that would normally be used to complete the tasks described on the left. In many cases, there is more than one way to do many of the tasks. The path you select is dependent on your operating environment.

Table 16. Summary table of steps

Task	Windows 9x	Windows NT Workstation
Add hardware support for a device, where drivers are included in the base operating system.	modify msbatch.inf	modify unattend.txt
Add hardware support for a device, where drivers are not included in the base operating system.	modify msbatch.inf update image	modify unattend.txt update image
Add printer support.	modify msbatch.inf update image	modify unattend.txt update image
Add printer support for printers not included in the base operating system.	Not supported	Not supported
Add network protocol support.	modify msbatch.inf	modify unattend.txt

Task	Windows 9x	Windows NT Workstation
Add service pack.	modify cmdlines.txt update image	modify logonclt.bat update image
Add TMA support.	modify cmdlines.txt update image	modify logonclt.bat update image
Add IBM JVM support.	modify cmdlines.txt update image	modify logonclt.bat update image
Change the TimeZone.	modify msbatch.inf	modify unattend.txt
Create a new machine class.	modify msbatch.inf update image	modify unattend.txt update image

## 5.4 Machine classes and response files

With WorkSpace On-Demand, the concept of defining a machine based on a class was introduced. This meant that we could define a base class say, a microchannel machine, and if we defined a machine as belonging to that class all the physical properties and drivers associated with that class will be used by that machine to boot.

With the Windows feature the response file can be used to define a machine class. Several default response files are included with the base product. The Windows 95 response files are located in the c:\IBMLAN\RPL\W95\RSP directory. The Windows 98 response files are located in the c:\IBMLAN\RPL\W98\RSP directory. The Windows NT response files are located in the c:\IBMLAN\RPL\NT40\RSP directory. Two machine specific or machine class response files are included for each of the supported operating systems.

### 5.4.1 IBM PC 300GL (Model 6561-54U)

The files IBM300GL.INF for Windows 9x and IBM300GL.TXT for Windows NT are response files that have been customized for the IBM PC 300 GL that have the following video and network interface adapter:

- Video Adapter: Cirrus Logic 546x 1.71a Graphics Adapter
- Network Interface Card: IBM EtherJet 10/100 Ethernet Adapter

### 5.4.2 IBM PC: 350, 730, and 750

The files IBMPC.INF for Windows 9x and IBMPC.TXT for Windows NT are response files that have been customized for the IBM PC 350, IBM PC730, or

IBM PC750 machines, The machine contains the following video adapter and network interface cards:

- Video Adapter: S3 Trio3D
- Network Interface Card: IBM Auto 16/4 Token-Ring Adapter

### 5.4.3 Creating your own machine classes

To create your own *machine classes*, you need to do the following:

1. Modify the default response file and save it with a name that describes your machine class in the response file directory.
2. If a device is not shipped with the base operating system code, add the device driver support. This is explained in the sections that follow.
3. Use the response file when you define the client by specifying the file name on the /RSP parameter.

#### 5.4.3.1 Adding a machine class

As an example, we will define a new machine class. The class that we are going to define is a PC 300 GL class that has the IBM Auto 16/4 Token-Ring drivers instead of the Ethernet that is defined in the original class. We will do the example for Windows NT only to describe the process. Windows 95 and Windows 98 are very similar. To add a new machine class, do the following:

1. Use an ASCII editor and open the response file  
c:\IBMLAN\RPL\NT40\RSP\IBM300.TXT.
2. Modify the driver information. Since the base class has the Ethernet adapter already defined, change the following lines that define the Ethernet adapter:

```
[SelectedAdaptersSection]
IBMFE = IBMFEParamSection, \${OEM$}\NET\IBMFE
```

```
[IBMFEParamSection]
```

This section should be changed to:

```
[SelectedAdaptersSection]
TRSR0001=TRSRParamSection, \${OEM$}\NET\IBMTRSR
```

```
[TRSRParamSection]
```

TRSR0001 represents the unique device ID. This ID is used to find the driver details in the oemsetup.inf file. This file is contained in the \\${OEM\$}\NET\IBMTRSR directory. The TRSRParamSection can be used to override the default parameters for the particular driver. \\${OEM\$}\NET\IBMTRSR represents the directory name where the driver is contained. This driver



was available by default, so no further work is needed to be done. An example is given later on how to add support for a driver that is not, by default, included with the base operating system.

3. Save the response file with a different name, for our example, c:\IBMLAN\RPL\NT40\RSP\IBM300TR.TXT.

4. To define the sample client we used above, we used the command:

```
netwin riplmach sample01 /add /mac:8884ac7763ec /release:nt40
/type:wimnt /dhcp:n /enabled:y /image:w32dossb.img /recordid:r_dtk_ndis
/partition:600 /cdkey:010-1578717 /REGUSER:"ITSO User"
/rsp:c:\rsp\ibm300TR.TXT
```

---

## 5.5 Video adapters

Some video adapters may have only become available after the operating system was shipped, so they are not included in the operating system image. To add support for video adapters, you have to perform a few steps for each operating system:

1. Ensure that the driver works with a stand-alone version of the operating system.
2. Add the video device driver files to the client image.
3. Update the response files to include the new driver.

### 5.5.1 Installing support for video adapters on Windows NT

There are a few ways to add support for an unsupported video adapter. One of them is described in the on-line Administrator's Guide. This example uses the textmode description on how to install the Matrox Video Drivers on Windows NT:

1. In the directory \IBMLAN\RPL\NT40\I386\%OEM% create a directory called Textmode.
2. Copy the OEM Video Drivers to the directory created in the previous step.
3. Create a Txtsetup.oem file in the Textmode directory. The file should look like this:

```
[Disks]
d1 = "OEM Video Disk", \OEMVideo.tag,\
[Defaults]
Display = MGA64
[Display]
MGA64 = " Matrox MGA64 Driver - OEM"
[Files.Display.mga64]
```

```

driver = dl,mga64.sys,mga64
dll = dl,mga64.dll
[config.mga64]
value=device0,InstalledDisplayDrivers,REG_MULTI_SZ,mga64
value=device0,VgaCompatible,REG_DWORD,0

```

4. Edit the Unattended.txt file located in the \IBMLAN\RPL\NT40\RSP directory and check or change the following:

```

[Unattended]
OEMPreinstall = Yes
[DisplayDrivers]
" Matrox MGA64 Driver - OEM" = "OEM"
[Display]
BitsPerPel = 16
XResolution = 800
YResolution = 600
VRefresh = 60
AutoConfirm = 1
[OEMBootfiles]
MGA64.SYS
MGA64.DLL
TXTSETUP.OEM

```

This method of installing video drivers only installs the video driver. Additional utilities or applications provided by the manufacturer have to be installed using other methods. Contact the video driver vendor for information on automation options for their products.

For more information on the unattended install of display drivers, go to:

<http://support.microsoft.com/support/kb/articles/Q166/0/28.asp>

### 5.5.2 Installing support for video adapters on Windows 9x

We completed the following steps to add support for the Matrox Video Drivers on Windows 9x. Check the online Administrator's Guide and readme file for clearer, more generic steps for adding device driver support. In the steps that follow, substitute 5 or 8 for X depending on the operating system:

1. In the directory \IBMLAN\RPL\W9x\c, create a directory called \$Display.
2. Copy the Matrox drivers to this directory.
3. Go to \$Display directory and edit MGA.INI.
4. Find the following lines and change them to the following:

```

Reboot = no
Final_Box = no

```

```
Install_Type = Typical  
[AVAILLANG]
```

5. Remove the language files you don't need.
6. Close and save document.
7. Go to \IBMLAN\RPL\W9x\RSP and edit the MSBATCH.INF.
8. Find [LOGONCLT.REG] and insert the following line:

```
HKLM, "Software\Microsoft\Windows\CurrentVersion\RunOnce", MGA, , "c:\$display\setup.exe"
```

---

## 5.6 Enabling network adapter support

The current list of supported adapters is available in Appendix B, "Supported network adapter list" on page 227. This is a list of adapters that have been tested to work with the respective operating systems.

Enabling support for an adapter describes the steps you would need to take to get the adapter installed. These steps do not include how to create the initial boot file that are used to boot the machine and copy the images to the server. Check the online documentation or the redbook, *WorkSpace On-Demand Handbook*, SG24-2028 for more information on how that is done.

### 5.6.1 Enabling network adapters on Windows NT

When installing device drivers for network adapters on Windows NT, there are three main steps:

1. Find the device ID for the driver (most network interface card manufacturers use the OEMSETUP.INF file).
2. Copy the driver to a subdirectory under  
  \IBMLAN\RPL\NT40\I386\SOEM\$\NET\ or  
  \IBMLAN\RPL\NT40\I386\DRVLIB.NIC directories.
3. Update the response file, UNATTEND.TXT.

The following example contains specific instructions for installing IBM Turbo 16/4 Token-Ring ISA Adapter on Windows NT:

1. Locate the newest Windows NT device driver diskettes from the adapter manufacturer. The adapter must support unattended installation. Refer to the manufacturer documentation for details about unattended installation.
2. List the files and directories on the device driver diskette.

3. Find the device ID for the driver. Most network interface card manufacturers use the OEMSETUP.INF file. For example, the device ID for IBM Turbo 16/4 Token-Ring ISA Adapter is under the [Options] section of the OEMSETUP.INF file. The value is TRSR1080.
4. If the driver is not included with this product, copy the driver to a subdirectory under \IBMLAN\RPL\NT40\I386\%OEM%\NET\. A subdirectory is provided for this driver:  
\IBMLAN\RPL\NT40\I386\%OEM%\NET\IBMTRSR.
5. In any ASCII text editor, open the Windows NT response file (UNATTEND.TXT) located at IBMLAN\RPL\NT40\RSP\UNATTEND.TXT.
6. Add or modify the [SelectedAdaptersSection]. For example, to add the turbo IBM Turbo 16/4 Token-Ring ISA Adapter, type:

```
[SelectedAdaptersSection]
TRSR1080=TRSRParamSection, \%OEM%\NET\IBMTRSR

[TRSRParamSection]

TRSR1080 represents the option name. TRSRParamSection points to the
[TRSRParamSection] that stores additional information about the device
driver upon installation. \%OEM%\NET\IBMTRSR represents the location where
the driver will be installed. IBMTRSR represents the unique subdirectory that
you make for each driver.
```
7. Save the UNATTEND.TXT file under a different file name in the IBMLAN\RPL\NT40\RSP directory. The new file you saved is the response file you use when you define clients. The filename is the value for the /RSP parameter for the NETWIN RIPLMACH command.

UNATTENDED.TXT:

```
[unattended]
...
...
[Network]
InstallAdapters = SelectedAdaptersSection

[SelectedAdaptersSection]
TRSR1080 =TRSRParamSection, \%OEM%\NET\IBMTRSR
```

### 5.6.2 Enabling network adapters on Windows 9x

Before installing network adapters on Windows 95 or Windows 98, consider that you have two options for managing your installation. If you have a number of different adapters in your organization for the same operating system, create a separate subdirectory under the \IBMLAN\RPL\W95\NET or \IBMLAN\RPL\W98\NET directory for each network adapter. Apply access

controls to any subdirectory you create. You can then reference each adapter using the [NetCardsDirs] section of the MSBATCH.INF response file.

With this method, you will limit any confusion if adapters have similar file names. If you have only one adapter type, you can copy the drivers directly to the \IBMLAN\RPL\W95\OS or \IBMLAN\RPL\W98\OS directory so that it will be distributed to the client with the operating system.

Because the instructions are so similar for Windows 95 and Windows 98, the example describes how to install one type of adapter for Windows 95 and a second example shows how to install multiple adapters for Windows 98. Both these examples use the IBM Turbo16/4 Adapter.

#### **Driver Location**

Do not download and install device drivers for the token-ring adapter from the Web.

For Windows 98 clients, the device driver for the IBM Turbo 16/4 Token-Ring ISA Adapter is packaged on the Windows 98 Microsoft CD. The device driver is included with the CD image when you install Windows 98 on the WorkSpace On-Demand server.

For Window 95 and Windows NT clients, the supported device driver is packaged on the WorkSpace On-Demand Feature for Windows Clients CD. You must modify MSBATCH.INF or UNATTEND.TXT, respectively, to ensure that the turbo driver installs on the appropriate client.

#### **5.6.2.1 Using the operating system directory on Windows 95**

The following instructions are for enabling the IBM Turbo 16/4 Token-Ring ISA network adapter for Windows 95. These instructions distribute the driver with the operating system image. In our lab setup, we did the following:

1. Locate the newest Windows 95 device driver diskettes from the adapter manufacturer. The adapter must support unattended installation. Refer to the manufacturer documentation for details about unattended installation.
2. Update the machine BIOS to the latest level as well as the network adapter. Most network adapters have flash memory that contains operational information that may need to be updated.

#### Note

The machine setup may not list the network adapter as a boot device unless the remote boot function has been enabled on the adapter. With this adapter, an adapter setup program has to be run to enable this feature.

3. Copy all files, including subdirectories, from the device driver diskette to the IBMLAN\RPL\W95\OS directory. In our example for the IBM Turbo 16/4 Token-Ring ISA network adapter, you copy diskette two (the device driver diskette) to IBMLAN\RPL\W95\OS on the boot server.

#### Caution

Before you copy the device driver diskette, verify that the file names on the diskette do not already exist in the \OS directory.

4. Next, update the response file. List the files in the device driver diskette and open the .INF file to obtain the value for the unique device ID of the driver. In the case of the IBM Turbo 16/4 Token-Ring ISA network adapter, you open NETIBM.INF. The device ID for this driver is \*IBM1080.
5. Using any ASCII editor, open the MSBATCH.INF file in the IBMLAN\RPL\W95\RSP directory.
6. Under the [Network] section, fill in the value for Netcards= with the device ID you found in the .INF file. For example, in the case of the IBM Turbo 16/4 Token-Ring ISA Adapter, the device ID is \*IBM1080.
7. MSBATCH.INF

```
[Network]
netcards=*IBM1080
IgnoreDetectedCards=1

[NetCardsDirs]
*IBM1080=NET\IBMTRSR
```

8. Save the changes to the MSBATCH.INF file in the IBMLAN\RPL\W95\RSP directory. You may want to save the response file using a different name. For example, you can save this file as TURBO.INF to show that this file includes changes for an IBM Turbo 16/4 Token-Ring ISA adapter. The file you saved is the response file you use when you define clients. The filename is the value for the /RSP parameter for the NETWIN RIPLMACH command.

### 5.6.2.2 Using separate directories on Windows 95

The following instructions are for enabling the IBM Turbo 16/4 Token-Ring ISA network adapter for Windows 95. These instructions distribute the driver within their own directories. In our lab setup, we did the following:

1. Locate the newest Windows 98 device driver diskettes from the adapter manufacturer. The adapter must support unattended installation. Refer to the manufacturer documentation for details about unattended installation.
2. Create a separate directory for each adapter under the IBMLAN\RPL\W98\NET directory.
3. Copy the diskettes for each adapter to the separate subdirectory you created for each adapter in step 1.
4. Next, you update the response file. List the files in the device driver diskette and open the .INF file to obtain the value for the unique device ID of the driver. Write down the name so that you can use it in a later step.
5. Using any ASCII editor, open the MSBATCH.INF file in the IBMLAN\RPL\W98\RSP directory.
6. Under the [Network] section, fill in the value for Netcards= with the device ID you found in the .INF file. For example, in the case of the IBM Auto 16/4 Token-Ring ISA Adapter, the device ID is \*IBM1080. For this example, the second device ID is \*XYZ123.
7. Because you are installing more than one network adapter, you need to add a section [NetCardsDirs] to the MSBATCH.INF file to indicate the corresponding directory where you installed each driver. For example:

#### MSBATCH.INF

```
[Network]
netcards=*IBM1080,*XYZ123
IgnoreDetectedCards=1

[NetCardsDirs]
*IBM1080=NET\IBMTRSR
*XYZ123=NET\XYZ
```

8. Save the MSBATCH.INF file in the IBMLAN\RPL\W98\RSP directory. The file you saved is the response file you use when you define clients. The filename is the value for the /RSP parameter for the NETWIN RIPLMACH command.

### 5.6.2.3 Installing the EtherJet 100/10 for Windows 95 OSR2

The following is a description on how to enable the IBM EtherJet 100/10 Ethernet adapter on Windows 95 OSR2.

1. The driver files for IBM EtherJet 100/10 are already included in the IBMLAN\RPL\W95\NET directory.
2. Using any ASCII text editor, open the MSBATCH.INF file for Windows 95 (located in IBMLAN\RPL\W95\RSP\MSBATCH.INF).
3. At the bottom of the [Setup] section, add the following line:  
OptionalComponents=1
4. In the [Network] section, set IgnoreDetectedNetCards=0.  
IgnoreDetectedNetCards=0
5. Under the [Network] section, fill in the value for Netcards= with the device ID you found in the .INF file. For example, in the case of the IBM EtherJet 100/10 Adapter, the device ID is IBMFE. For example:

```
[Network]
netcards=IBMFE
IgnoreDetectedCards=0
```

6. Add a section [NetCardsDirs] to the MSBATCH.INF file to indicate the corresponding directory where each driver is installed. For the EtherJet 100/10 adapter, the driver is already included in the IBMLAN\RPL\W95\NET directory:

```
[NetCardsDirs]
IBMFE=NET\IBMFE
```

7. Immediately after the [Network] section, add an [OptionalComponents] section. Include the following lines:

```
[OptionalComponents]
"Dial-Up-Networking"=1
"Direct Cable Connection"=1
```

8. In the [Network] section, comment out the Netcards= statement:

```
;Netcards=
```

9. In the [LOGONCLT.REG] section, add the following line:

```
HKLM, "System\CurrentControlSet\Services\VxD\DHCP", "PopupFlag",
65537,00,00,00,00
```

10. In the [LOGONCLT.REG] section, uncomment the following line:

```
HKLM, "Software\Microsoft\Windows\CurrentVersion\RunOnce", "ipfix", , "c:\I
BMWIN32\ipfix.exe"
```



### 5.6.3 IBM Auto, Turbo, and 3COM

After you install Windows 95, Windows 98, or Windows NT on the client system, you might encounter the following error during the Windows startup process:

Your network adapter IBM Auto or Turbo 16\4 Token-Ring ISA Adapter(0001) is not working properly. You may need to set it up again. For more information, see the Network Troubleshooting Windows Help.

This error may be caused by a down-level version of the token-ring adapter microcode. Download the latest version on the microcode from:

<http://www.networking.ibm.com>

During the Windows client installation process, some clients with 3COM 3C90x adapters stop responding after the HDBOOT executable loads. This may be caused by a down-level version of the LanWorks BootWare ROM. To request the current version of the MBA Flash Update for the

BootWare ROM, contact 3COM at:

<http://www.3comm.com>

To continue with the installation process after you update the BootWare ROM, reset the client system (using the `NETWIN RIPLMACH/RESET` command) or run `FDISK /MBR`.

---

## 5.7 Printing

The IBM WorkSpace On-Demand 2.0 Feature for Windows Clients provides a mechanism to configure local printers on a client machine. This feature uses the support built into Windows 9x. However, Windows NT does not provide any mechanism to install printer drivers and to configure local printer queues.

### Note

To see a list of supported printer models, open the Add Printer Wizard in a client (such as a sandbox for the same operating system). Printer drivers not shipped with the Windows operating systems cannot be installed.

### 5.7.1 Windows 9x printing

For Windows 9x, printer support is configured through the `NETWIN RIPLMACH` command. You can install a single printer or multiple printers using the `NETWIN RIPLMACH` command. The parameter to configure a printer is:

```
/PRINTER:="PrinterName, ModelName, Port"
```

The result of adding this parameter is a section in the `MSBATCH.INF` file is:

```
[Printers]
PrinterName = PrinterModel, Port
```

If more than one printer is required for a client, the configuration information should be added to a template response file.

The following is an example for a Windows 95 machine:

```
NETWIN RIPLMACH w95cl1 /MAC:000152365478 /RELEASE:W95 /TYPE:WIN95
/IMAGE:W32DOSSB.IMG /RECORID:R_DTK_NDIS /SANDBOX /ADD
/PRINTER:"LOCALQ1,LASERJET.HP PaintJet300,LPT2"
/PRINTER:"NETWORK1,LASERJET.HP Paintjet300,\\server\printershare"
```

`LOCALQ1` and `NETWORK1` represent the printer names. `LASERJET.HP PaintJet300` represents the driver model for both printers. `LPT2` represents the port and `\\printserver\share2` represents the UNC name.

#### 5.7.1.1 Using a response file

You can also modify the `MSBATCH.INF` file to install multiple printers for Windows 95 and Windows 98 clients. The `MSBATCH.INF` file resides in the operating system directory (`IBMLAN\RPL\W95\MSBATCH.INF` or `IBMLAN\RPL\W98\MSBATCH.INF`).

Under the `[Printers]` section, make a separate entry for each printer. Use the following format:

```
PrinterName=ModelName,Port (for a local printer) or
PrinterName=ModelName,UNCname (for a network printer).
```

The following example installs a local printer and a network printer:

```
[Printers]
"Printer1"="Canon Bubble-Jet BJC-600",LPT1
"Printer2"="HP Laserjet IIIsi",\\printserver\share2
```

`Printer1` and `Printer2` represent printer names. `Canon Bubble-Jet BJC-600` and `HP Laserjet IIIsi` represent driver models. `LPT1` represents a local printer port. `\\printserver\share2` represents a UNC name for a shared network printer.

## 5.7.2 Windows NT printing

As mentioned previously, the Windows NT Workstation does not provide an unattended way to install printer drivers and configure local printer queues. With the restricted desktop of WorkSpace On-Demand clients, an end user cannot configure a local printer device after the machine has been booted. To address this issue, a utility was created that installs a printer driver and configures the queue. This utility can run in an unattended mode. The format of the command is:

```
PRNINST
-N:name
-M:model
-P:port
-R:driver
-D:data
-C:config
```

Where `name` is the printer name, `model` is the printer model, `port` is the local port where the printer is attached, `driver` is the driver file, `data` is the data file, and `config` is the configuration file.

Since printer information can vary between clients, the `unattended.txt` file is used to provide the client with printer information. To enable this support, the `NETWIN RIPLMACH` command accepts the additional parameters on the printer string that NT requires, recognized by the `/TYPE=WINNT` parameter.

Therefore:

```
NETWIN RIPLMACH... /TYPE:WINNT /PRINTER:="PrinterName, ModelName, Port,
Driver, DataFile, ConfigFile"
```

Results in the following section in the `UNATTENDED.TXT` file:

```
[Printers]
PrinterName = ModelName, Port, Driver, DataFile, ConfigFile
```

### 5.7.2.1 PRNINST

The `PRNINST` utility parses the `UNATTENDED.TXT` file looking for the `[Printers]` section. The `UNATTENDED.TXT` file is copied to the `C:\IBMWIN32` subdirectory during installation.

If the section is blank or not found, the utility exits without defining a printer. A printer is configured for each entry found in the `[Printers]` section. If duplicate names exist, the utility errors on the duplicate name(s) and continues through the list.

NT expects to find the printer driver(s) in C:\\$\system32\spool\drivers\w32x86, where \$\$ is defined by TargetPath= in the [Unattended] section of the response file, UNATTENDED.TXT. These drivers must be in an uncompressed format. It is up to the system administrator to ensure the appropriate driver(s) gets installed into the specified location. This can be accomplished by uncompressing the drivers on a sandbox client and copying the required driver(s) to the \\${OEM}\\$\system32\spool\drivers\w32x86 directory on the boot server.

The PRNINST utility runs once the spooler has been initialized by the RunOnce facility in Windows NT. In addition, the utility runs under the administrator context.

Prininst.exe is executed from the LOGONCLT.BAT file that runs as a RunOnce program under the context of Administrator. PRNINST.EXE is installed into the \RPL\NT40\LOGON directory.

#### 5.7.2.2 Using the NETWIN RIPLMACH command

The prininst utility makes it somewhat transparent to the administrator. It hides Windows NT's lack of remote print install support.

To define a printer using the NETWIN RIPLMACH command you need to do the following:

1. Before assigning the printer to a client, copy the printer driver to the \IBMLAN\RPL\NT40\i386\\${OEM}\C\WINNT\SYSTEM32\SPOOL\DRIVERS\W32X86 subdirectory.
2. Use one line when typing the NETWIN RIPLMACH command and all of the parameters. The following example uses more than one line due to the width of the page.

```
/PRINTER:"LOCALQ1,LASERJET.HP  
PaintJet300,LPT1,driver,datafile,configfile"  
/PRINTER:"LOCALQ2,LASERJET.HP  
Paintjet300,LPT2,driver,datafile,configfile"
```

LOCALQ1 and LOCALQ2 represent the printer names. LASERJET.HP PaintJet300 represents the driver model for both printers. LPT1 and LPT2 represent the ports.

#### Note

To define multiple printers using the `NETWIN RIPLMACH` command, use `/PRINTER` for each printer. Install the printer driver on a sandbox client and print a test page to see the valid values for the driver, `DataFile`, and `ConfigFile` that are particular to your printer.

#### 5.7.2.3 Using a response file

To use a response file to install and configure printers, do the following:

1. Before modifying the response file, copy the printer driver to the `\IBMLAN\RPL\NT40\I386\%OEM%\C\WINNT\SYSTEM32\SPOOL\DRIVER\SW32X86` subdirectory.
2. The `UNATTEND.TXT` file resides in the operating system directory (`\IBMLAN\RPL\NT40\UNATTEND.TXT`). Under the `[Printers]` section, make a separate entry for each printer. Use the following format:

```
PrinterName=DriverModel,Port (for a local printer) or
PrinterName=DriverModel,UNCname (for a network printer).
```

The following example installs a local printer and a network printer:

```
[Printers]
"Printer1"="Canon Bubble-Jet BJC-600",LPT1
"Printer2"="HP Laserjet IIIsi",\\printserver\share2

Printer1 and Printer2 represent printer names. Canon Bubble-Jet BJC-600
and HP Laserjet IIIsi represent driver models. LPT1 represents a local
printer port. \\printserver\share2 represents a UNC name for a shared
network printer.
```

3. To see a list of supported printer models, open the Add Printer Wizard in a client (such as a sandbox for the same operating system). Printer drivers not shipped with the Windows operating systems cannot be installed.

---

## 5.8 Network protocol support

The network protocols that are installed are defined within the `Protocols` Section of the `Unattended.txt` for Windows NT or the or the `MSBATCH.INF` for Windows 9x.

For Windows NT, the default `Unattended.txt` file contains the following abstract for protocols:

```
[ProtocolsSection]
NBF = NBFParamSection
```

```
TC = TCPIPParameters
```

```
[TCPIPParameters]
```

```
DHCP = yes
```

```
[NBParamSection]
```

The above abstract is made up of two sections: the ProtocolsSection and a section for each of the protocols listed below the protocol section header. In the above example, it is TCP/IP and Netbeui.

NetBEUI has to be selected, since the client is only supported on a local area network. If a client needs to access files or printers on remote servers, support for NetBIOS over TCP/IP is required. The following is a list of network protocols that can be specified. Documentation on the configuration parameters for each of the protocols is available from Microsoft.

- NBF
- NWLNKIPX
- TC
- DLC
- RASPPPTP
- STREAMS
- ATALK

---

## 5.9 The Microsoft ScriptIt utility

The Microsoft ScriptIt is a tool that can be used for automating interactive software installations or configuration tasks. For applications or tasks that cannot be automated using response files, ScriptIt is one of the few alternatives to visiting every workstation. This tool can also be used for tasks that can be automated using the response files, although we recommend that you use response files wherever possible.

### Note

Microsoft warns that it does not provide technical support for scripts prepared with the ScriptIt tool. If you plan to use such scripts in a production roll out, ensure that they work correctly and also ensure that you have a feedback mechanism that reports the failure and success of tasks performed with this tool.

The ScriptIt tool allows you to start a process. The tool can then monitor the window titles of the active processes, and when needed, send a set of keystrokes to the correct window. ScriptIt can differentiate between windows with similar titles. These keystrokes can be predetermined or they can be determined at run time.

ScriptIt is not a traditional scripting language and does not offer many of the traditional programming language constructs. Because the tools act as though there were a user at the workstation, it needs to be started with sufficient security privileges to perform the installation or configuration task.

Documentation and the tool is available for download from the Microsoft Web site at:

<http://www.microsoft.com/NTServer/nts/deployment/custguide/scriptit3.asp>

There is no licensing restriction; however, ScriptIt and ScriptIt scripts are unsupported. The ScriptIt readme file is included in the download.

### 5.9.1 Using ScriptIt

The ScriptIt tool is very easy to use. The ScriptIt tool, or processing engine (scriptit.exe), is initiated from the run command in the following way:

```
scriptit script file
```

Where script file is a text file that is in .ini format. Briefly, the script file uses .ini file keys to identify the window titles and specifies the keystrokes that ScriptIt should send to these windows as .ini key values. For example, the script file entries:

```
[SCRIPT]
run=calc.exe
Calculator=9*12=
```

The above script will launch the calculator application and calculate the product of 9 times 12. The basic ScriptIt script file can have the following format and line types:

```
[SCRIPT]
run=command
runwait=command
WindowTitle=keystrokes
mkfile filename.ext=fileline
REM comment

[ADLIB]
WindowTitle=keystrokes
```

WindowTitle<+text>=keystrokes

The documentation referenced above contains a full description of each of the sections. Table 17 is a summary of those descriptions.

Table 17. ScriptIt file options

Keyword	Description
The [SCRIPT] Section	The [SCRIPT] header tells ScriptIt where to begin processing the script file. Any text above this tag is ignored and can be used as remarks. Lines in this section are processed sequentially. Within this section, four types of lines are recognized: run and runwait, title and title+text, mkfile, and rem.
Run lines(run and runwait)	These run lines start executables, run valid MS-DOS operating system commands, and interpret valid WinBatch commands.
Window title lines (title and title+text)	Window title lines identify an instance of a window and send the designated keystrokes.
Make file lines (mkfile)	Make file lines cause the script to open a new file as specified by filename and write lines of text to it as specified by the file line.
Remarked lines (REM)	Remarked lines allow script lines to be deactivated and documented.
[ADLIB]	This optional section provides instructions for ScriptIt to use when the setup program displays a window or dialog box that is not part of a normal scripted process and therefore not specified in the [SCRIPT] section. Entries in the [ADLIB] section allow ScriptIt to respond appropriately to errors or exceptions that occur during the processing of the script section. Lines that appear in this section are processed only if a matching window is displayed; otherwise, they are ignored.

The ScriptIt tool is useful. However, by not having any of the sophisticated features of traditional scripting languages, such as Perl or Rexx, limits its



usage. However, the mkfile line, together with the [ADLIB] section, allows you to enhance some of the tool's abilities. The mkfile line allows ScriptIt to write a Perl or other script at run time. This script can then be executed from a title line. By using a left quotation mark ('), the output of the external script can be redirected to the designated window title.

### 5.9.2 Multimedia support using ScriptIt

In our example below, we used the ScriptIt tool to install Crystal Audio drivers. In order to do that, we completed the following steps:

1. Install the driver on a standalone machine and record all the keystrokes that are required for the installation.
2. Create a script file that contains all of these keystrokes. The audio.scr file is shown in Figure 33 on page 173.

```
[SCRIPT]
run=rundll32.exe shell32.dll,Control_RunDLL mmsys.cpl
Multimedia Properties={RIGHT 4}{TAB 2}{ENTER}
Add={ENTER}
Install Driver=C:\CRYSTAL{ENTER}
Add Unlisted or Updated Driver={ENTER}
Driver Exists={RIGHT}{ENTER}
CrystalWare(TM) Audio Driver={ENTER}
System Setting Change={ENTER}
Multimedia Properties={TAB 3}{ENTER}
```

Figure 33. Script file, audio.scr

3. Next, we change the Logonclt.bat file to include the step to install the Crystal audio driver. This step is highlighted in the Logonclt.bat file in Figure 34 on page 174.

```

C:\CRYSTAL\SCRIPTIT\SCRIPTIT.EXE c:\CRYSTAL\SCRIPTIT\audio.scr
C:\WINNT\system32\spool\drivers\w32x86\prninst.exe -N:"Demo Printer"
-M:"Lexmark Optra S 1255" -P:LPT1: -R:RASDD.DLL -D:LMPCLN41.DLL
-C:RASDDUI.DLL
C:
CD \LOGON
RUNDLL32 setupapi,InstallHinfSection DeleteAutoAdminLogon 128
c:\logon\ntclean.inf
SETUP domain=%DOMAIN% sandbox=%SANDBOX% -s

```

Figure 34. Updated logonclt.bat file adding support for the Crystal audio drivers

4. Ensure that the appropriate files for the Crystal audio drivers are available for installation. Figure 35 is a listing of all the files and their sizes. These files were copied into the x:\IBMLAN\rp\nt40\i386\%OEM%\c\crystal directory.

```

18,944 BNT190RN.DOC
21,505 CSPNP.INF
236,032 CWB3DSND.EXE
8,625 CWBAUDIO.BIN
281,664 CWBAUDIO.SYS
126,976 CWBAUDLL.DLL
41,606 MIDIMAP.CFG
340 OEMSETUP.INF

```

Figure 35. File list for the Crystal drivers

## 5.10 Other client commands

The following sections list commands that can be used to perform a number of client tasks.

### 5.10.1 Changing the client operating system

Changing the client operating system means that the client needs to be redefined. What happens is that, effectively, the client definition is deleted and then redefined with the new operating system. When the client is rebooted, the new operating system is installed.

To change the client operating system, you must redefine the client. You can change the operating system using the `NETWIN RIPLMACH` command. You must

verify that your client operating system is copied to the right directory. To do that, complete the following steps:

1. Logon on to the boot server where the client is defined
2. Change the client definition. At command prompt, type:

```
NETWIN RIPLMACH machinename /TYPE: WINNT or WIN95 or WIN98  
/CDKEY:productid /RELEASE:version /RESET
```

(machinename represents name of the client whose operating system is changing.)

3. Restart the client for the changes to take place.

### 5.10.2 Resetting a client.

If the client's operating system is damaged, or if you need to redistribute your client image or reinstall, there are two ways to do that. You can reset or redefine the clients.

To reset a client:

1. Log on to the server where the client is defined.
2. From command prompt, type:

```
NETWIN RIPLMACH machinename /RESET
```

(machinename represents the name of the client whose operating you are reinstalling.)

3. Restart your client for the changes to take place.

### 5.10.3 Redefining a client

To redefine a client:

1. Log on to the server where the client is defined.
2. From command prompt, type:

```
NETWIN RIPLMACH machinename /DELETE
```

(machinename represents the client whose operating you are reinstalling.)

### 5.10.4 Deleting a client.

You can use the `NETWIN RIPLMACH` to delete Feature for Windows Clients clients from your server, by specifying the `/DELETE` parameter.

From command prompt, type:

```
NETWIN RIPLMACH machinename /DELETE
```

(machinename represents the client you are deleting.) If you delete a client definition while the client is running, or if the command prompt is set to a path inside the client definition (for example, \BMLAN\RPLUSER\ machinename) when you delete the client definition, you will receive an error message. In both instances, the client definition will be only partially deleted. You must delete the remainder of the client definition manually.

#### 5.10.5 Querying client information

You can also use the `NETWIN RIPLMACH` command to gather the information about a defined Feature for Windows Clients client. Using the `NETWIN RIPLMACH` command without specifying the client name, lists the defined clients with their associated client operating systems.

The `NETWIN RIPLMACH` command with the machinename provides you with output of the client. From the command prompt, type:

```
NETWIN RIPLMACH machinename
```

(machinename represents your client name.)

---

## Chapter 6. User and desktop administration

This chapter describes the methods and tools that can be used to customize a user's environment. It provides details on the Windows User Profiles and the System Policy files and how they can be changed to create a custom and easy to manage environment.

---

### 6.1 Windows Clients

Workstations with the Feature for Windows Clients need to log on to a WorkSpace On-Demand server to ensure that the images on the local drives are kept current. All of the network services provided by clients currently connecting to Windows NT servers can be provided by the OS/2 Warp Server.

Support for both Windows 9x and Windows NT Workstations for functions, such as home directories, logon assignments, network applications, logon scripts, and more, is available from the server.

In addition to these services, management of policy files and user profiles is also enforced. These are described in the sections that follow.

Command line interfaces are the sole method of administering some of the user options. The commands need to be executed by system administrators.

#### Remote Commands

Being able to remotely execute the `NETWIN APP` and `NETWIN USER` commands is possible through the usage of existing mechanisms, such as `NET ADMIN` and `Telnet`. In all cases, the command must be directed to a primary domain controller to be executed with system administrator privilege.

#### 6.1.1 The `NETWIN USER` command

The `NETWIN USER` command adds, deletes, and displays information about Windows applications assigned to a user's Windows desktop. It is also used to assign a restricted desktop to a selected user. If the specified user does not have a home directory, one will be added as a side effect of assigning a restricted desktop to the user. Adding a home directory changes the user accounts subsystem (`NET.ACC`) for the specified user. Another side effect of assigning a Windows 32-bit application to a user is that a logon assignment is made if the application is server-based. The logon assignment is used to give

the user runtime access to the application. Adding a logon assignment changes the user's Domain Control Database (DCDB) information.

The `NETWIN USER` command must be invoked by an administrator at a primary domain controller. It will provide the following options:

<b>userID</b>	Specifies an existing user account.
<b>/ADD</b>	Indicates that a Windows application will be added to the specified user's desktop. When /ADD is specified, /APP must also be specified.
<b>/DELETE</b>	Indicates that a Windows application will be removed from the user's desktop. When /DELETE is specified, /APP must also be specified.
<b>/DESKTOP:pathname</b>	Indicates that a restricted desktop is to be assigned to the specified user. When /DESKTOP is specified, /TYPE must also be specified. Specifies a fully-qualified path (on the domain controller) to the default desktop of the type that will be assigned to the specified user.
<b>/APP:appid</b>	Specifies the unique application ID of an existing Windows application definition that is to be added or removed from a user's desktop. /APP is valid only when used with /ADD or /DELETE.
<b>/TYPE:{WIN9X   WINNT}</b>	Indicates the type of the restricted desktop that will be assigned to the specified user. /TYPE should not be specified with /ADD or /DELETE.

The `NETWIN USER` command must ensure that the following criteria are met before attempting any user management:

- The Server service is started.
- The local machine is the domain controller.
- A user is logged on with administrator privileges.

When adding a restricted desktop to a user account, a home directory will be created for the user if the user does not have one. The home directory will be created in `PKG_DIRvalue\users\userid`, where `PKG_DIRvalue` defaults to `IBMLAN\DCDB\` and the `userid` is specified on the command line.

The home directory information will be added to the users account on the server.

#### Note

When you define a desktop, you have to do it either for Windows 9x or Windows NT. If a user uses both systems, and you want to assign a desktop to both operating systems, you have to do it twice, one for NT on a NT SANDBOX and another for Win9x on a Win9x SANDBOX.

---

## 6.2 User profiles and system policy files review

Windows User Profiles and system policy files are described in some detail in Section 2.5.1.1, “The user profile” on page 44 and Section 2.5.1.2, “System policy files” on page 47. What follows is a review of the concepts that were already covered.

A user profile defines the Windows environment that is loaded when a user logs on. The user profile contains all user-defined settings for the computer, including network connections, display settings, printer connections, and so on. The primary goal of user profiles is to provide a user with a common environment between computers, that is, allowing the user to roam with common settings.

System policies are used mainly to restrict what users can do. For example, you can use system policies to restrict what users can do from their desktops, such as restricting certain Control panel options or configuring network settings. Using a policy editor, you can create a file that contain registry settings. These registry settings will be written to the user or the local computer portion of the registry database of the client computer to generate a desired environment for the user.

From the above summary, it is possible, at times, to use either of the two mechanisms to control the user environment. As a rule of thumb, it is always best to try to use the system policy file rather than the user profile to control the user’s environment. This is because the system policy file is contained in one central location and is easier to update.

In the sections that follow, we use both methods to do different tasks.

---

## 6.3 Customizing the desktop or user profile

Since the desktop in the Workspace On-Demand documentation refers to the customizing the user profile, these two words can be used interchangeably. In

the text that follows, we use the word desktop to ensure consistency with the product documentation.

The Feature for Windows Clients comes with a default desktop for both Windows 9x and Windows NT. The location of the default desktop is shown in Figure 12 on page 46. The process that occurs when a desktop is assigned is also described in Section 2.5.1, “Building the desktop” on page 44.

In this section, we describe, using examples, how to create, change, and assign desktops.

### **6.3.1 Planning your desktop**

Before starting to create a desktop, you should consider what the user environment should look like. These are some of the questions to help you better prepare your implementation:

- What folders should be on the desktop?
- What should the Start menu look like?
- How should shortcuts and links be displayed for the user?
- How much of the users environment should be controlled?
- What should the background look like; is there a corporate standard?
- What will the desktop look like on different hardware platforms?

When deciding what a user should be presented with, it is wise to consult with the user community and understand their requirements. As a rule of thumb, users who perform very specific, repetitive tasks daily prefer a simple desktop; whereas, users who perform a variety of different tasks are able to navigate their way around, and the layout of the desktop is not very significant but needs to be less restricted.

The more restricted the desktop is made, the easier it is to manage, and this lowers the overall cost of support. However, you need to ensure that this does not hamper the overall productivity of the individual user.

For task-based users, you would want to create a profile geared toward enhancing the work environment for the set of applications that the users need. You would hide functionality from the user for two reasons:

- To simplify the user interface so that the user has easy access to the functions required.
- To focus the user's attention on what he or she must do to get his or her job done.



Try to create a few common desktop types. Do this by grouping users into the tasks they perform. Create desktops associated with those groups.

### 6.3.2 Defining a new desktop

Defining a new desktop is a simple but fairly long process. As mentioned above, try to create standard desktops for groups of people. Customizing individual desktops takes the value away from managing the environment.

In short, to create a new desktop, you set up a sandbox machine, modify the desktop as you like it, and save the desktop. This saved desktop now can be assigned to Windows users. The detailed steps to define a new desktop are as follows:

1. Create a user account with administrator authority on the WorkSpace On-Demand Domain controller. Do not assign the user a desktop.
2. Log on to the sandbox by using the user account you created.
3. Verify that user profiles are enabled if you are using a Windows 95 or Windows 98 sandbox. To verify that user profiles are enabled:
  - Click **Start->Settings->Control Panel**. The screen is shown in Figure 36.

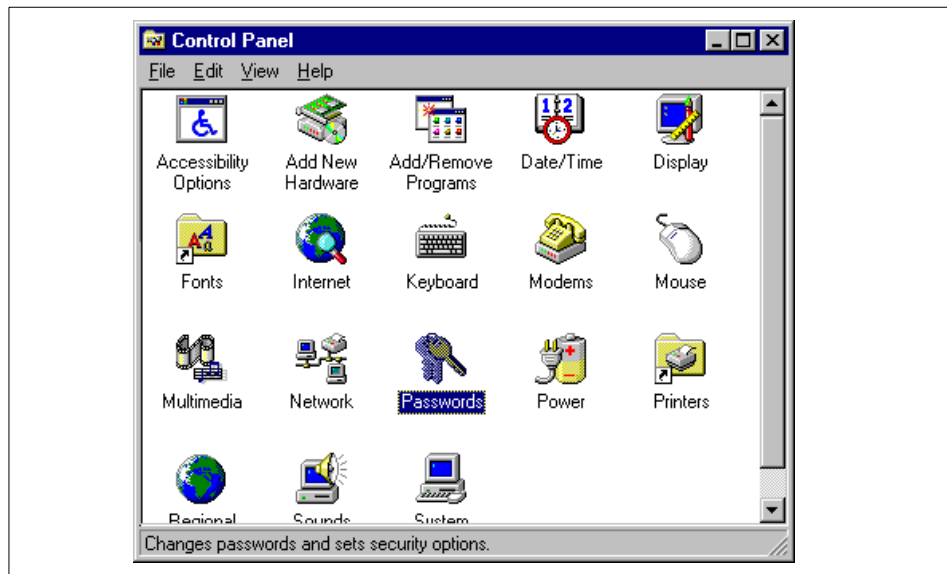


Figure 36. Windows 95, Control panel

- Double-click **Passwords**, then select the **User Profiles** tab.

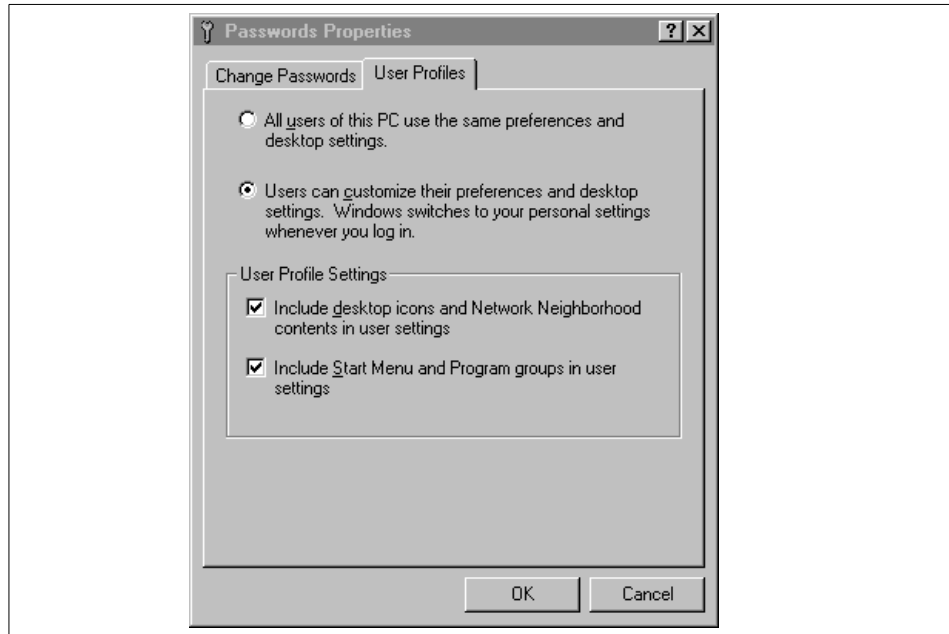


Figure 37. Windows 95, Password Properties

- As shown in Figure 37, verify that the **Users can customize their preferences and desktop settings** button and both **User Profile Settings** boxes are selected.
  - Click **OK**.
4. Customize the desktop to reflect the appearance you want your new desktop to have. The way you customize the links and Start menu entries is the way that those links and entries will display on the user's desktop.
  5. There are various attributes that you might want to change on the desktop to reflect either a department or corporate standard. Most of these can be changed by selecting properties from the desktop context menu.

To change the wallpaper, select the **Background** tab as shown in Figure 38 on page 183.

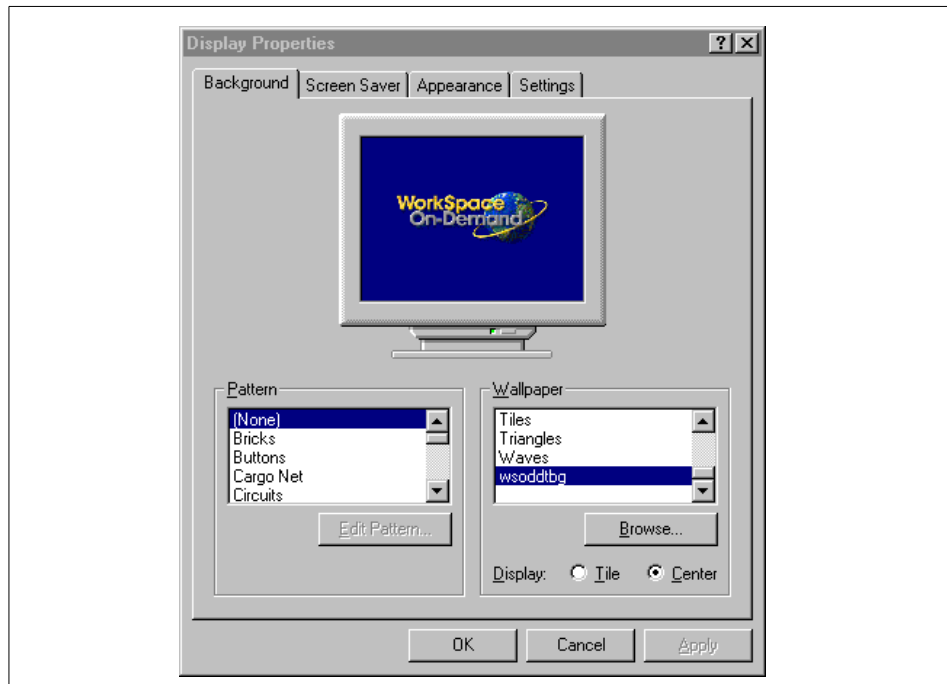


Figure 38. Windows 95 Display Properties, Background tab

The pattern you select is applied to either a wallpaper or a color. The wallpaper selection is on the right side of the box. You can also select to tile the wallpaper across the screen. The wallpaper is contained in the c:\Windows directory for Windows 9x and in the c:\Winnt directory for Windows NT. If you place a custom bitmap in that directory, it will be selectable as a wallpaper. Custom bitmaps need to be distributed with the operating system image to be selectable.

6. To change the color scheme, select the **Appearance** tab from the Display Properties window as shown in Figure 39 on page 184. Before you change the color scheme, be sure to consult the user community.

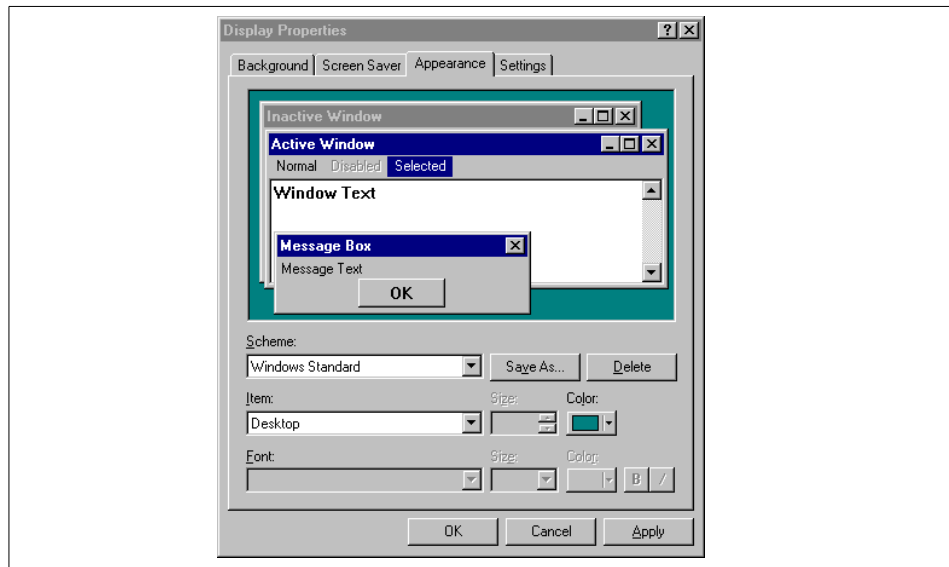


Figure 39. Windows 95 Display Properties, Appearance tab

7. To change the screen saver, select the **Screen Saver** tab as shown in Figure 40.

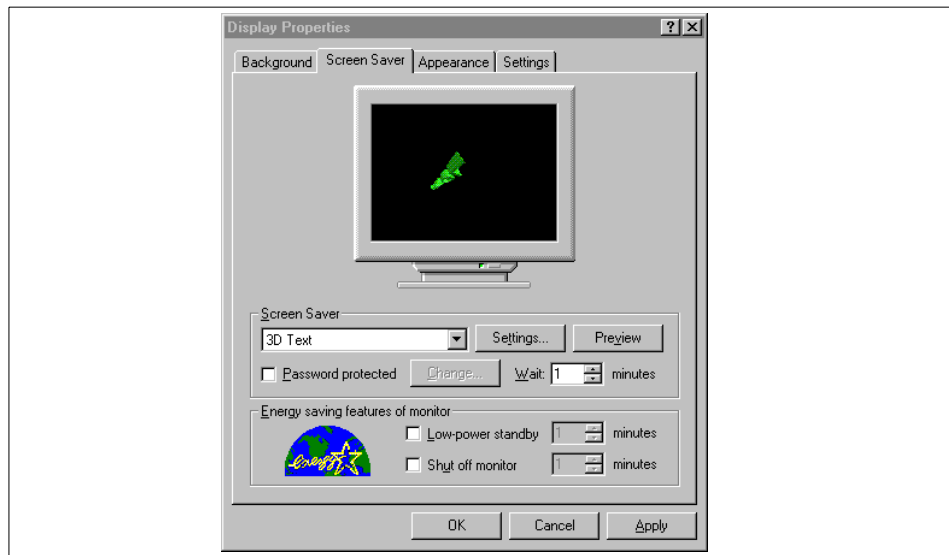


Figure 40. Windows 95 Display Properties, Screen Saver tab

8. To change the Start menu, do the following:

**For Windows 95:** Update the Start menu on a Windows 95 sandbox by changing the contents of the following directories:

- %SYSTEMROOT%\PROFILES\userMENU directory (user represents the name of the sandbox user account).
- %SYSTEMROOT%\PROFILES\ALL USERS\START MENU directory.

%SYSTEMROOT% is the location where the selected Windows operating system is installed.

Figure 41 shows a Windows 95 Start menu before customization. We are going to remove some of the items from the Programs tab. To do this, you need a command prompt. Change to the C:\WINDOWS\Profiles\indrann\Start Menu\Programs and delete the items you do not want on the menu. indrann is the userid that is being used at this workstation.



Figure 41. Windows 95 Program menu, before customization

Figure 42 on page 186 shows a listing of the directory that corresponds with the latter Start menu.

```

C:\WINDOWS\Profiles\indrann\Start Menu\Programs>dir
Volume in drive C is WIN95
Volume Serial Number is 2750-16FD
Directory of C:\WINDOWS\Profiles\indrann\Start Menu\Programs
.                <DIR>          07-03-99 11:20a .
..               <DIR>          07-03-99 11:20a ..
ONLINE~1         <DIR>          07-03-99 11:20a Online Services
ACCESS~1         <DIR>          07-03-99 11:20a Accessories
STARTUP          <DIR>          07-03-99 11:20a StartUp
INTERN~1 LNK      327  07-01-99  4:00p Internet Explorer.lnk
INTERN~2 LNK      374  07-01-99  4:00p Internet Mail.lnk
MICROS~1 LNK      315  07-01-99  4:00p Microsoft NetMeeting.lnk
MS-DOS~1 PIF      967  07-03-99 11:14a MS-DOS Prompt.pif
WINDOW~1 LNK      296  07-01-99  4:00p Windows Explorer.lnk
PIXWIZ~1         <DIR>          07-03-99 11:25a PixWizard
                5 file(s)          2,279 bytes
                6 dir(s)        321,404,928 bytes free

```

Figure 42. File listing for the Start menu, Programs tab

In order to remove items from the Programs tab, we deleted one directory and all the files we did not require. Each file refers to a link, and each directory refers to a set of links.

```

C:\WINDOWS\Profiles\indrann\Start Menu\Programs>dir

Volume in drive C is WIN95
Volume Serial Number is 2750-16FD
Directory of C:\WINDOWS\Profiles\indrann\Start Menu\Programs

.                <DIR>          07-03-99 11:20a .
..               <DIR>          07-03-99 11:20a ..
ACCESS~1         <DIR>          07-03-99 11:20a Accessories
STARTUP          <DIR>          07-03-99 11:20a StartUp
MS-DOS~1 PIF      967  07-03-99 11:14a MS-DOS Prompt.pif
PIXWIZ~1         <DIR>          07-03-99 11:25a PixWizard
                1 file(s)          967 bytes
                5 dir(s)        321,478,656 bytes free

```

Figure 43. File listing for the Start menu, Programs tab, after modification

Once we deleted all the files, the Start menu was immediately modified and displayed as shown in Figure 44.



Figure 44. Windows 95 Program menu, after customization

Customizing the Start menu for Windows NT and Windows 98 is very similar.

**For Windows NT Workstation:** Update the Start menu on a Windows NT sandbox by changing the contents of the:

- %SYSTEMROOT%\PROFILES\ userMENU directory (user represents the name of the sandbox user account).
- %SYSTEMROOT%\PROFILES\ALL USERS\START MENU directory

**For Windows 98:** Update the Start menu on a Windows 98 sandbox by changing the contents of the:

- %SYSTEMROOT%\PROFILES\ userMENU directory (user represents the name of the sandbox user account).
- %SYSTEMROOT%\ALL USERS\START MENU directory.

#### Note

To limit user access, verify that the command prompt is not available from the desktop you create. A command prompt provides direct access to local directories. In the previous example, the command prompt was not excluded.

9. Empty the **recycle bin** and clear any documents from the Taskbar.
10. Log off the sandbox. This saves the updated desktop to the user-account home directory.
11. Create a new directory in the <IBMLAN\DCDB>\WIN32APP\DESKTOP\WINNT directory for Windows NT desktops or in the <IBMLAN\DCDB>\WIN32APP\DESKTOP\WIN9X directory for Windows 9X desktops. This location is where you will store your new desktop. For example, to create a subdirectory for your desktop in the WINNT directory, change to the WINNT directory and type:
 

```
MD NEWDESK
```
12. Change to the home directory of the user account you created.
13. Change to the PROFILES.W95 subdirectory for Windows 9X desktops or the PROFILES subdirectory for Windows NT desktops. These are hidden subdirectories. You can use the `DIR /AH` command to display the directories. Copy the contents of the subdirectory to the new subdirectory you created in step 7. For example, to copy the contents of a Windows NT desktop to a new directory that is named NEWDESK, at a command prompt, type:
 

```
XCOPY *.* X:\IBMLAN\DCDB\WIN32APP\DESKTOP\WIN9X\NEWDESK /S /E /T /H
```
14. You can now assign the new desktop to other user accounts.

### 6.3.3 Changing the appearance of an existing desktop

Changing the appearance of an existing desktop is very similar to creating a new desktop. With a new desktop you are starting with a system default. Now, you are starting with an initial desktop. To change an existing desktop, do the following:

1. Log on to the primary domain controller as an administrator. Create a user with administrator authority and assign the user a home directory.
2. Assign the desktop you are changing to the user account you created. To assign a desktop to the user account, at a command prompt, type:
 

```
NETWIN USER username /DESKTOP:desktop_name /TYPE:WIN9X or WINNT
```

 (desktop\_name represents the name of the subdirectory in the <IBMLAN\DCDB>\WIN32APP\DESKTOP\WINNT directory or the <IBMLAN\DCDB>\WIN32APP\DESKTOP\WIN9X directory where the desktop resides. You can also specify a fully qualified path to the desktop subdirectory.)
3. Change to the home directory assigned to the user account you created above.



4. Change to the `PROFILES` subdirectory for a Windows NT desktop or the `PROFILES.W95` for a Windows 9X desktop. These are hidden subdirectories that you can access from a command line. You can use the `DIR /AH` command to display the directories.
5. If the subdirectory contains a `.MAN` file, rename the `.MAN` file to a `.DAT` file. This enables the server to update the Windows User Profile when you change the desktop appearance. If the file already contains a `.DAT` file, continue to the next step. If you are editing a Windows NT desktop, type:
 

```
RENAM NTUSER.MAN NTUSER.DAT
```

 If you are editing a Windows 9X desktop, type:
 

```
RENAM USER.MAN USER.DAT
```
6. Log on to the sandbox by using the user account you created and complete the steps as described above for creating a new desktop.

#### 6.3.4 Deleting the Connect to the Internet icon

All users receive the Connect to the Internet icon on Windows 98. To delete the Connect To The Internet icon from Windows 98 desktops, complete the following steps:

1. Log on to a Windows 98 sandbox with a user account that has local administrator authority and administrator authority on the server.
2. `NET USE` the drive on the boot server where the Windows 98 operating system resides. (If there are multiple boot servers in your environment, remove the icon from every boot server.)
 

For example, if the Windows 98 operating system image resides on the C drive of the boot server, type the following at a command prompt:

```
C: NET USE X: \\<boot server name>\C$
```
3. Change to the `X:\IBMLAN\RPL\W98\OS` directory (this directory resides on the boot server).
4. Insert the Windows 98 CD into the CD-ROM drive.
5. Open Windows 98 Explorer. Change to the `E:\WIN98` directory (E represents the CD-ROM drive).
6. Double-click **PRECOPY2.CAB** to view the contents of the cab file.
7. Right-click **ICW97.INF** and click **Extract**.
8. Select the root directory on the C drive (`C:\`) as the target location for the `ICW97.INF` file.
9. Click **OK**.

10. Change directories to `C:\`.
11. Open **ICW97.INF** in any ASCII text editor.
12. Type a semicolon before the entry under the [BaseWinOptions] section.
13. Type a semicolon before all the entries under the [msicw.reg] section.
14. Save the changes. Refer to the following ICW97.INF file fragment as an example:

```

;=====
; ICW97.INF
;
; Copyright (c) 1994, 1998 Microsoft Corporation
;
; This is the Setup information file to install
; The Internet Connection Wizard version 1.1
; for Windows98 (IE4) .
;
;=====

[version]
LayoutFile = layout.inf,layout1.inf, layout2.inf
signature = "$CHICAGO$"
SetupClass = BASE

[BaseWinOptions]
;msicw.reg ;this line was commented out
;
[msicw.reg]
;all the statements in this section need to be commented out
;DelFiles=DeleteICW
;CopyFiles=CopyINF,CopySYS,CopyICW,CopyHELP
;UpdateInis=UPD.Links
;PerUserInstall=ICW.links.pui
;AddReg=MSICW.RegEntries,MSICW.RegOLEObjects

;no changes below this line

;=====

```

15. From Windows 98 Explorer, change the CD directory to `WIN98/TOOLS/RESKIT/INFINST`.
16. Click **INFINST.EXE**. This displays a dialog box.
17. Type `C:\icw97.inf` in the Inf to add to Windows 98 Setup field.

18. Next to the Windows 98 setup.exe field, click the **browse** button and select the drive letter you used to connect to the boot server in step 2.
19. Change to the IBMLAN\RPL\W98\OS directory and click **SETUP.EXE**.
20. Click **OK**. This displays the following message:  

```
"This Windows 98 Setup already contains an inf named "icw97.inf". Adding this inf will remove the functionality of current inf. Do you want to continue?"
```
21. Click **Yes**. This displays the following message:  

```
"No PnP ID was found in icw97.inf. This inf will not install any PnP hardware. Do you want to continue?"
```
22. Click **Yes**.
23. When the INF finishes installing, exit the INFINST program.

---

## 6.4 System policies

System policies are described in Section 2.5.1, "Building the desktop" on page 44. A system policy is a set of registry settings that defines the computer resources available to an individual or to a group of users. Policies can be used to restrict what users can do from their desktops, such as restricting certain Control panel options or configuring network settings. System policies can be implemented for specific users, groups, computers, or for all users. System policies are created with the System Policy Editor (poledit.exe).

Most settings affect all users that log on to the domain. Care should be taken to ensure that these changes and restrictions are required for all users. While individual user restrictions can be implemented in the system policy file, it makes the file more difficult to manage. Make every attempt to restrict the file to generic computer and user settings. This is more of a problem when managing large distributed environments, such as remote branches offices.

System policies and user profiles can be used to configure, in many cases, the same settings. System policies, however, address a larger portion of the registry and are usually easier to customize and manage. The following flow chart in Figure 45 on page 192 describes how the system policies are implemented.

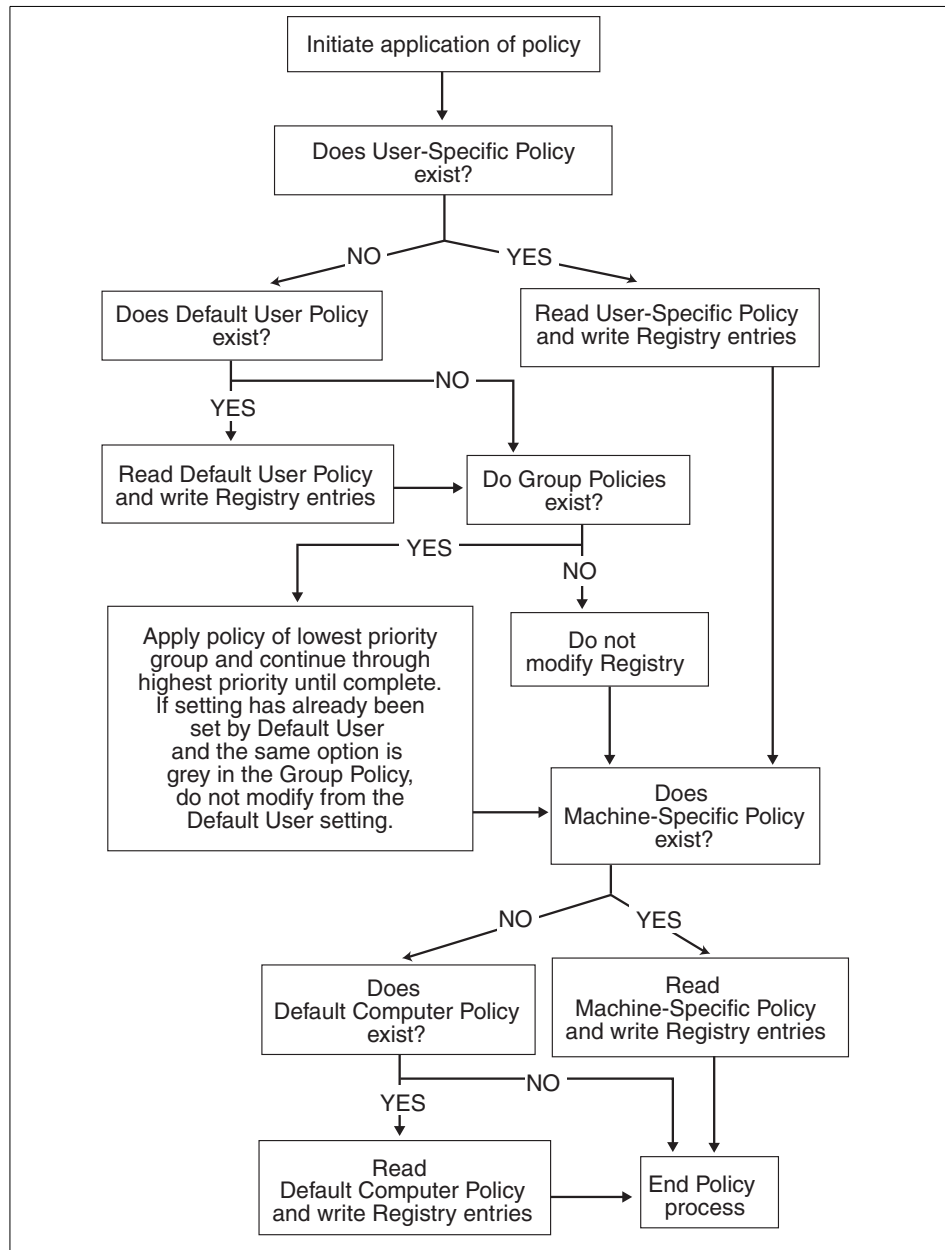


Figure 45. Implementing the system policy

### 6.4.1 The System Policy Editor

The System Policy Editor is a graphical tool provided with Windows NT Server 4.0 that allows you to generate registry settings that create a custom environment for a particular user or group of users. The editor creates a file that contains registry settings that are then written to the user or local machine portion of the registry database.

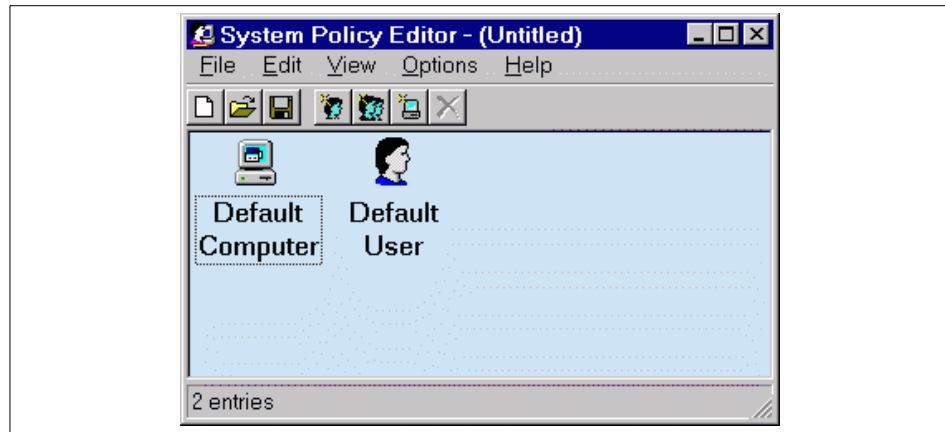


Figure 46. System Policy Editor

Using the System Policy Editor (poledit.exe), you can create a file that contains registry settings. The registry settings you define here are written to the user or local computer portion of the Registry database on the client computer to generate the required environment.

You can find a copy of Poledit in the following locations:

- Windows 95 OSR2 OEM CD in the \ADMIN\APPTOOLS\POLEDIT directory
- Windows NT Server 4.0 CD in the \CLIENTS\SRVTOOLS\WINNT\I386 directory
- Windows 98 CD in the \TOOLS\RESKIT\NETADMIN\POLEDIT directory
- Windows NT 4.0 Workstation Service Pack 3 CD in the \I386 directory

### Policy Files

A policy file is valid only for the platform on which it was created. For example, if you run Poedit.exe on a Windows 95-based machine, and you save the policy file, the file will be written in a format that can be interpreted by Windows 95-based machines only. The same is true when you create policy files on Windows NT-based machines.

Windows 95 and Windows NT policy files are not interchangeable.

The policy editor is automatically installed as part of Windows NT server. It is contained in the administrative tools group. To install the System Policy Editor on a Windows NT Workstation, you have two options:

1. Copy the System Policy Editor (poedit.exe) and the template files to the workstation. The template files have an .adm extension and are located in the x:\winnt\inf directory, which is hidden by default.
2. Run the Setup.bat file from the Windows NT 4.0 CD-ROM. This file is located in the \Clients\Srvtools\Winnt directory.

Installation of the editor on a Windows 95 Workstation.

When you install the System Policy Editor on a Windows 95-based computer, the installation process modifies the Windows 95 registry to allow system policies to function correctly. You can install the policy editor from the Windows 95 Upgrade or Retail CD, or you can install the editor that ships with Windows NT Server 4.0.

To install the System Policy Editor from the Windows 95 CD:

1. Insert the Windows 95 Upgrade CD in your CD-ROM drive.
2. Open **Control panel**, and click **Add/Remove Programs**.
3. Click the **Windows Setup** tab, and select **Have Disk**.
4. Browse to locate the directory x:\Admin\Apptools\Poedit\ (where x is drive A through Z) on the Windows 95 CD.
5. Select both **Group Policies** and the **System Policy Editor**, and then click **OK** to install.
6. It is important that you run the setup program as described above. Undesirable results will occur if you merely copy the System Policy Editor and related files to a Windows 95-based computer.

To install the System Policy Editor from a Windows NT 4.0 server:

1. Copy the Poedit.exe file from the Windows NT 4.0 server to the \windows directory of the Windows 95-based machine.
2. Copy the Common.adm and Windows.adm files from the Windows NT 4.0-based server to the \windows directory of the Windows 95-based machine.
3. Create a shortcut to the System Policy Editor executable (Poedit.exe, located in the \windows directory of the Windows 95-based computer).

After you complete the installation, the administrative tools group includes the System Policy Editor.

#### 6.4.2 System policy file considerations

When implementing system policies, you should consider the following questions. There may be different groups of users in your environment and if the following questions show different requirements for each group, you may have to implement groups in your system policy. Alternatively, you could use the User Profile feature to give groups their unique settings. Consider the following questions:

- What common desktop attributes would you like to have for all users?
- What restrictions do you want to imposed?
- Which computer settings do you want to affect?
- Which user settings do you want to affect?

#### 6.4.3 System policy templates

The System Policy Editor uses administrative (.adm) files to determine which registry settings can be modified. An .adm file is a text file. It is a hierarchical template and consists of categories and subcategories that dictate which settings are available through the System Policy Editor user interface. The .adm file contains the registry locations where changes should be made for a particular selection, additional options for a particular selection, restrictions, and, in some cases, the default value for a selection.

When you run the System Policy Editor, select the **Policy Template** from the Options menu, as shown in Figure 47 on page 196. The policy template options lists the .adm files that are currently being used. If you have a custom application, you can create your own .adm file and add it to this list. WorkSpace On-Demand comes with a custom .adm file that needs to be used when updating the system policy file that is shipped with the product. The .adm files for WorkSpace On-Demand are contained in the <IBMLAN\DCDB>\WIN32APP\POLICY directory and the file is called WORKSNT.ADM.

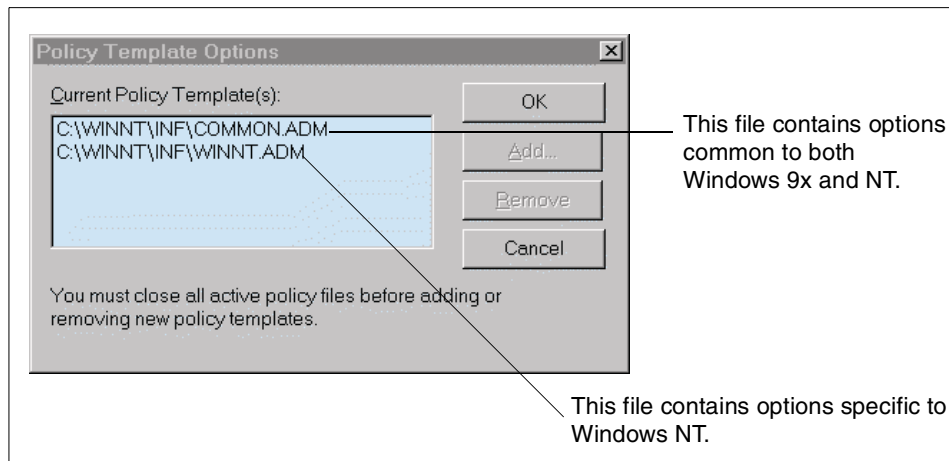


Figure 47. Policy template options

As described in Figure 47, the COMMON.ADM file contains options that are common to both Windows 9x and Windows NT. The WINNT.ADM file contains options valid for Windows NT only. Therefore, instead of having one large template file, you can have multiple custom templates that apply to applications.

A policy template file consists of a hierarchy of categories and sub-categories that define:

- How policies are displayed through the administrative interface
- Registry locations where the changes should be made
- Options that accompany the policy
- Restrictions to values that can be selected
- Default value that should be used if the option is selected

#### 6.4.4 Configuring policy settings

Policies in the policy file are described in a hierarchical tree structure. This structure is determined by the .adm file. An open policy file is shown in Figure 48 on page 197.



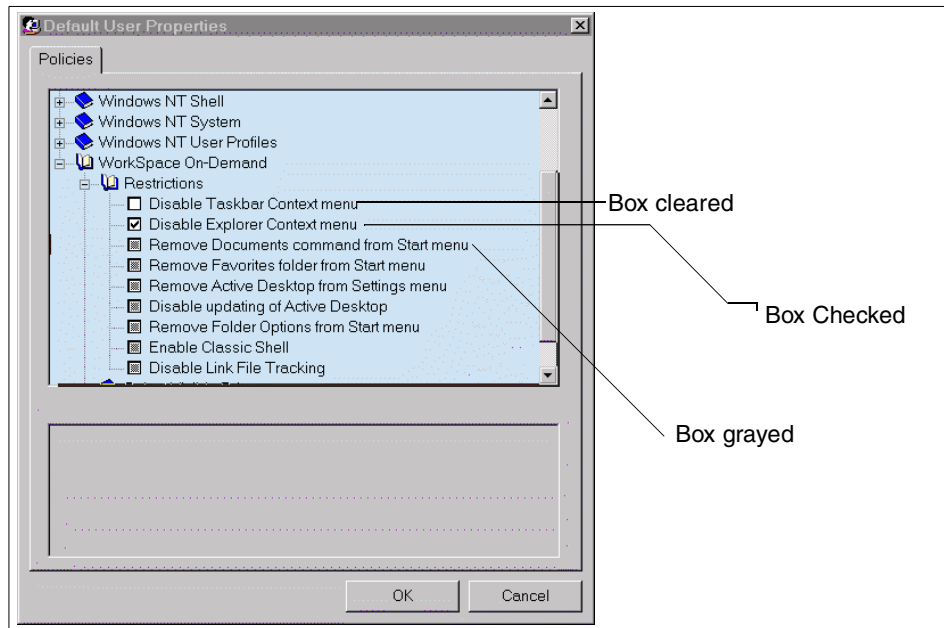


Figure 48. Editing the policy file

Since the policy file contains registry settings that may already have a value, when you edit a policy, the state of the box value is as shown in Table 18.

Table 18. Policy status

Box	Implication
Checked: Policy is implemented	When the user next logs on, the user's computer conforms to the policy. If the option was checked the last time the user logged on, Windows NT makes no changes.
Cleared: Policy is not implemented	If the settings were previously implemented, they are removed from the registry.
Grayed: Setting is ignored	The setting is ignored and unchanged from the last time the user logged on.

When you select an option, the pane below it contains other configurable items that relate to the setting you modified as well as information about the option you selected.

When administering System Policies, if you specify paths for options, such as wallpaper, ensure that the paths are consistent across all workstations that will receive that policy.

## 6.4.5 WorkSpace On-Demand .adm files

The WorkSpace On-Demand .adm files contain a number of options. These are described briefly in Table 19.

### 6.4.5.1 For Windows NT Workstations

The Windows NT Workstation .adm file is located in the x:\IBMLAN\DCDB\win32app\policy directory. It can be modified but care should be taken. Most of the policies are self-descriptive.

Table 19. worksnt.adm file options

Options
<b>Default Computer Section: WorkSpace On-Demand: CD Auto Run</b>
Disable CD Auto Run
<b>Default User: Restrictions</b>
Disable Taskbar Context menu
Disable Explorer Context menu
Remove Documents command from Start menu
Remove Favorites folder from Start menu
Remove Active Desktop from Settings menu
Disable updating of Active Desktop
Remove Folder Options from Start menu
Enable Classic Shell
Disable Link File Tracking
<b>Default User: Select Visible Drives</b>
Choose drives to make visible
<b>Default User: Windows Task Manager</b>
Disable Task Manager

#### 6.4.5.2 For Windows 9x Workstations

The Windows 9x Workstation .adm file is also located in the x:\IBMLAN\DCDB\win32app\policy directory. It can be modified but care should be taken. Most of the policies are self-descriptive.

Table 20. worksnt.adm file options

Option
<b>Default Computer Section: WorkSpace On-Demand: CD Auto Run</b>
Disable CD Auto Run
<b>Default Computer Section: Restrictions</b>
Show Windows 98 welcome tips at logon
Remove Windows Update from Settings menu
Remove Windows Update from Update Device Driver wizard
<b>Default User: Restrictions</b>
Disable Taskbar Context menu
Disable Explorer Context menu
Remove Documents command from Start menu
Remove Favorites folder from Start menu
Remove Active Desktop from Settings menu
Disable updating of Active Desktop
Remove Folder Options from Start menu
Enable Classic Shell
Disable Link File Tracking
Remove Internet Explorer Channel Band from desktop
Show Windows 95 welcome tips at logon
<b>Default User: Select Visible Drives</b>
Choose drives to make visible
<b>Default User: Windows Task Manager</b>
Disable Task Manager

#### 6.4.6 Changing the Windows NT system policy file

To change the system policy file, complete the following steps:

1. Decide what desktop settings you want to restrict.
2. Log on to the sandbox as an administrator.
3. Connect to the NETLOGON share on the primary domain controller. This enables you to remotely save your policy file onto your server. At a DOS prompt, type:

```
NET USE X: \\servername\netlogon
```

x represents any unused drive letter, and `servername` represents the name of the primary domain controller. `netlogon` represents the path to the netlogon share.

To find the path to the NETLOGON share:

- a) Log on to the primary domain controller as an administrator.
- b) At a command prompt, type:

```
NET SHARE
```

- c) The path to the NETLOGON share is displayed.

4. Connect to the <IBMLAN\DCDB>\WIN32APP\POLICY directory on the primary domain controller. At a DOS prompt, type:

```
NET USE Z: \\servername\directory
```

(z represents any unused drive letter, and `servername` represents the name of the primary domain controller.)

5. Start **Microsoft Poledit**.
6. In Poledit, click **Options** and then click **Policy Template**.
7. In the Policy Template box, add three policy templates:  
COMMON.ADM: This file is on the CD with POLEDIT.  
WINNT.ADM: This file is on the CD with POLEDIT.  
WORKSNT.ADM: This file is on the primary domain controller in the <IBMLAN\DCDB>\WIN32APP\POLICY directory.
8. Close the **Policy Template** box.
9. Click **File** and then click **Open Policy**.
10. Open the NTCONFIG.POL file from the Workspace On-Demand directory in the NETLOGON share.

11. Customize the policy file. For example, the H: and U: drives are visible if you use the default policy files. To select or change which drives are visible to users:

- Double-click **Default User**.
- Click **WorkSpace On-Demand**.
- Click in the **Select Visible Drives** box until a check mark is displayed.
- In the Select Visible Drives drop-down list, click any of the settings and then click **OK**.

Select the drives you want to be accessible to users.

12. Save the updated policy file as NTCONFIG.POL in the WorkSpace On-Demand directory (for general users) or in the ADMIN directory (for users with administrator privileges) on the NETLOGON share. To make further changes to the file before it restricts Windows NT desktops, save it as a different name, such as NTTEST.POL.

**Note**

The system policy file must be saved on a server that can process client log on requests. Place the NTCONFIG.POL file in the appropriate subdirectory (WSOD, ADMIN, or SANDBOX) on every backup domain controller.

#### 6.4.7 Changing the Windows 95 and Windows 98 system policy file

To change the system policy file, complete the following steps:

1. Decide what desktop settings you want to restrict.
2. Log on to the sandbox as an administrator.
3. Connect to the NETLOGON share on the primary domain controller. This enables you to remotely save your policy file onto your server. At a DOS prompt, type:

```
NET USE X: \\servername\NETLOGON
```

(x represents any unused drive letter, and `servername` represents the name of the primary domain controller.)

4. Connect to the <IBMLAN\DCDB>\WIN32APP\POLICY directory on the primary domain controller. At a DOS prompt, type:

```
NET USE Z: \\servername\directory
```

(z represents any unused drive letter, and `servername` represents the name of the primary domain controller.)

5. Start **Microsoft Poedit**.
6. In Poedit, click **Options** and then click **Policy Template**.
7. In the Policy Template box, add three policy templates:  
COMMON.ADM: This file is on the CD with POEDIT.  
WINDOWS.ADM: This file is on the CD with POEDIT.  
WORKS9X.ADM: This file is on the primary domain controller in the  
IBMLAN\DCDB>\WIN32APP\POLICY directory.
8. Close the **Policy Template** box.
9. Click **File** and then click **Open Policy**.
10. Open the CONFIG.POL file from the Workspace On-Demand directory in the NETLOGON share.
11. Customize the policy file. For example, the H: and U: drives are visible if you use the default policy files. To select or change which drives are visible to users:
  - Double-click **Default User**.
  - Click **Workspace On-Demand**.
  - Click in the **Select Visible Drives** box until a check mark is displayed.
  - In the Select Visible Drives drop-down list, click any of the settings and then click **OK**.
  - Select the drives you want to be accessible to users.
12. Save the updated policy file as CONFIG.POL in the Workspace On-Demand directory (for general users) or in the ADMIN directory (for users with administrator privileges) on the NETLOGON share. To make further changes to the file before it restricts Windows 95 and Windows 98 desktops, save it as a different name, such as TEST.POL.

**Note**

The system policy file must be saved on a server that can process client log on requests. Place the CONFIG.POL file in the appropriate subdirectory (WSOD, ADMIN, or SANDBOX) on every backup domain controller.

#### 6.4.8 Hiding the desktop

For many task-based users, the icons on the desktop can be confusing. Usually, these users are limited to two or three applications. These could be

setup to launch from the Start menu and the desktop could be cleared. The user will then focus on the start button. To clear the desktop, you could create a custom desktop with no icons. The other alternative is to use the system policy file.

In the system policy file, there is a Hide all items on the desktop. The Windows desktop is made up of three windows. Each of these windows is layered over the other. The lowest layer is the main desktop window. Above this is a transparent window that contains the desktop icons. The topmost window is the tray window (the Taskbar).

When you select the **Hide all items on the desktop** policy, the middle transparent window's visible status is turned off. The icons are still on this window but they are not visible to the user. According to Microsoft, the current design of the shell does not support hiding individual desktop icons on a per-user or per-group basis.

#### 6.4.9 The Start menu

The Start menu also can be configured by defining a user desktop. However, for more central control, you could also use the system policy to define the contents of a custom program menu. To do this, you would use two policies as shown in Figure 49 on page 204. The Remove common program groups from the Start menu policy and the Custom Programs folder policy. The first policy removes the programs group from the Start menu. The second policy points the users to a common set of policies. This could be done on a per-user basis, or it could be set to all point to a common location.

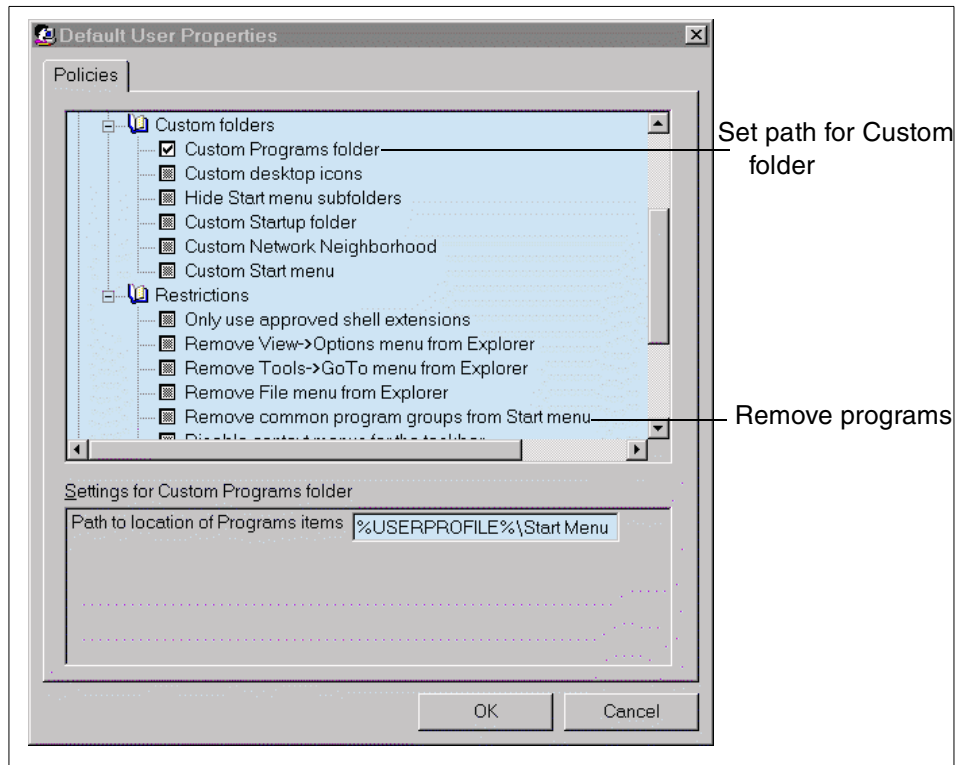


Figure 49. Setting up a custom program folder



---

## Chapter 7. The Tivoli TMA client

The IBM WorkSpace On-Demand 2.0 Feature for Windows Clients does not offer the same level of operating software or application software management that the OS/2 client does. Duplication of operating system and application files on client machines means that each time a critical file changes, each of the client machines have to be updated. When the entire operating system or a large application is distributed, redefining and refreshing the image on the client is warranted. However, for incremental updates, the refresh method used by WorkSpace On-Demand may prove to be drastic in practice.

This chapter discusses the Tivoli TMA client that is shipped with the IBM WorkSpace On-Demand 2.0 Feature for Windows Clients. This client is specifically intended to resolve the latter problem.

---

### 7.1 About Tivoli as a company

Tivoli was founded to ensure that enterprise-wide computing offers companies a strategic advantage. For customers working in enterprise computing environments, effective systems management is the final barrier to success. Tivoli management software breaks down this barrier by providing a unified, standard management approach to computing environments—one that hides the complexity and scale of these environments.

As a result, Tivoli management software has become the standard for managing network computing. Tivoli management software products are developed in cooperation with leading hardware and systems software vendors. They are used by leading organizations worldwide and are supported by dozens of third-party software products.

Tivoli now delivers the first true, end-to-end management solution that includes a single, standard, and unified method for managing everything from desktop PCs and laptops to data center mainframes.

Tivoli Enterprise provides management solutions that make it easier for large and small organizations worldwide to centrally manage all of their corporate computing resources. Tivoli Enterprise gives businesses the power to effectively manage the key software applications that drive business performance and profits.

---

## 7.2 The Tivoli Management Framework

Today's network computing enterprise requires an open, scalable, and cross-platform approach that is also truly integrated. The Tivoli Management Framework is the foundation for a suite of management applications that are making systems and network management easy.

The Tivoli Management Framework makes it possible to:

- Shield administrators from platform-specific details of day-to-day operations. Common operations, such as deploying applications and routine network maintenance, can be performed with a single action; administrators are no longer required to repeat the same operation for each platform on your enterprise.
- Deploy applications to literally thousands of machines with one operation, all the while ensuring the applications remain available.
- Integrate with third-party applications, because the Tivoli Management Framework is an open solution. Tivoli's Partner Association provides many of these third-party solutions to your enterprise management challenges, such as job scheduling, intrusion detection, and backup and restore, all of which snap into the Tivoli Management Framework.
- By providing a truly open and comprehensive foundation for network computing management, the Tivoli Management Framework simplifies today's most complex network computing challenges.

---

## 7.3 Introducing the TMA

The most visible new feature of the new Version 3.6 of the Tivoli Management Framework is the Tivoli Management Agent (TMA), previously called the Lightweight Client Framework (LCF) Endpoint. The TMA is an extension of the classic TME 10 Framework that increases scalability of TMRs, while reducing the hardware and software requirements on the managed systems.

The TMA-related extensions to the framework introduce three object types that represent system roles in a TMR:

1. Endpoint (TMA)
2. Endpoint Gateway
3. Endpoint Manager

Although each of the above systems logically represent a different system's role in the Tivoli environment, it should be noted that a single physical system

can contain more than one of the above object types. That is, one system could contain an endpoint manager, an endpoint gateway, and an endpoint. (In most of environments, endpoints gateways reside on different systems than endpoint managers.)

### **7.3.1 The endpoint (TMA)**

The endpoint is installed on the systems to be managed. The endpoint does not include any capability to perform management operations on other systems. These systems are managed and are not involved in the management of other nodes.

The endpoint function resides in the node to be managed. It runs as a small daemon, or background task. This daemon is called `lcfd`. It is responsible for executing methods at the request of a managing system. Its only connection to the rest of the Tivoli world is through an endpoint gateway.

When an endpoint is installed, a minimal number of files are installed on the managed system. Functionally, the only thing that is installed is the `lcfd` itself. When an application invokes a method to be executed on the managed system (endpoint), The method is automatically downloaded to the endpoint and executes the `lcfd`.

### **7.3.2 The endpoint gateway**

The endpoint gateway is a software component that runs on a full Tivoli managed node, enabling the managed node to operate as a gateway between a cluster of endpoints and the rest of the TMR. Each TMR can have multiple endpoint gateways. Currently, one TMR server can handle up to a approximately 200 endpoint gateways.

### **7.3.3 The endpoint manager**

The endpoint manager stores the association between the endpoint gateways and endpoints. It establishes and maintains the relationship between an endpoint and a gateway and is automatically created on the TMR server when you install the server.

An endpoint manager's primary role is to assign an endpoint to a gateway when the endpoint first logs in. If an endpoint's assigned gateway stops responding to the endpoint for some reason, the endpoint manager might again be involved in assigning the endpoint to a new gateway.

---

## 7.4 The role of the TMA

The Tivoli Management Agent (TMA), in conjunction with the Tivoli Framework and Management Applications, provides the customer with the framework necessary to perform management operations, such as software distribution, inventory, user administration, and distribution monitoring. TMA is already provided with IBM WorkSpace On-Demand 2.0 Feature for Windows Clients. The preloaded TMA allows the customer who purchases the product to immediately use the management features provided by Tivoli.

Windows clients with the Feature for Windows Clients installed have the option of having the TMA within their software package (it is not installed by default). This means the lcf daemon could already exist on the user's hard disk. However, by default, the preloaded TMA engine (the lcf daemon) does not start automatically, since it will only be of value if the Tivoli Framework is installed in the customer's environment. Therefore, when the client is installed, the TMA (lcf daemon) exists under the `..\Tivoli\lcf` directory in an inactive state.

---

## 7.5 TMA installation prerequisites

The following is a list of the management objects required to install and use endpoints:

**TMR Server** - The Tivoli Management Server component includes the libraries, binaries, data files, and graphical user interfaces needed to install and manage the Tivoli environment.

**Endpoint Manager** - The endpoint manager software runs on the TMR server. The endpoint manager maintains the information related to known endpoints and endpoint gateways.

**Endpoint Gateway** - The endpoint gateway provides the primary interface between a set of endpoints and the rest of the TMR. A single endpoint gateway can support communications with thousands of endpoints.

**Endpoints** - The endpoints are managed systems taking advantage of the Lightweight Client Framework. You can gather required management information from thousands of endpoint machines and remotely manage those machines with very little overhead. The endpoint is officially referred to as the Tivoli Management Agent (TMA) in Version 3.6 of Tivoli.

### 7.5.1 Enabling the TMA

This section describes the steps that need to be taken to enable the TMA. Figure 50 shows the various components within a Tivoli environment and their interaction.

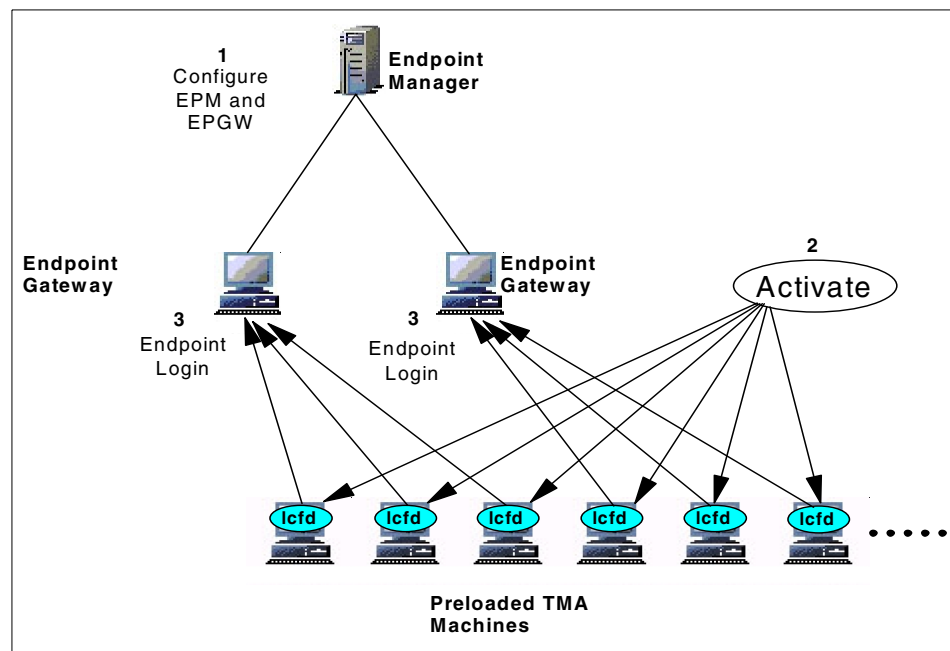


Figure 50. Steps to enable the TMA

To enable the TMA, you need to complete the following steps in your environment:

1. Create and configure the endpoint manager and endpoint gateways.
2. Activate the preloaded TMA.
3. Endpoint Login - The TMA logs in to the appropriate endpoint gateway that you configured during the activation process. Then, you can activate the preloaded TMA with the appropriate options. The active process of the preloaded TMA allows you to specify options for the `lcfd` daemon, so you can configure and control the endpoint login for all of the endpoints.

When the endpoint is started, it performs the endpoint login process to participate in the TMR. This is described in Figure 51 on page 210.

1. The endpoint establishes communications with the TMR.

2. The endpoint manager selects the endpoint gateway for the endpoint.
3. The endpoint receives its gateway assignment information and performs the normal login to the assigned gateway.

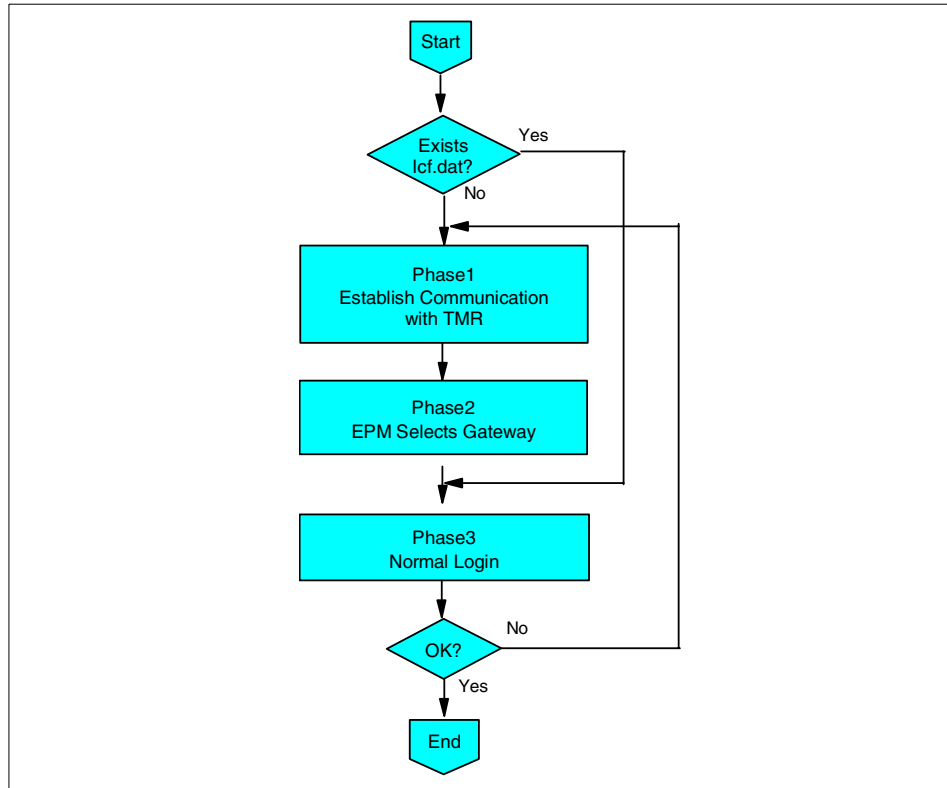


Figure 51. The endpoint login process

In the classic Tivoli management environment, the installation process implicitly determined the TMR to which a manager system belonged and there was no requirement to explicitly configure the managed nodes to communicate with their TMR server.

In the TMA environment, the endpoint must be configured to communicate with an appropriate endpoint gateway. To allow for both flexibility and availability, the association between the endpoint and the endpoint gateway can be dynamically determined and may change if a particular endpoint gateway is not available.

After receiving its gateway assignment from the intercepting gateway, the endpoint performs the normal login to its assigned gateway. At this time, the login policy is run for the first time, and the endpoint codeset is downloaded to the endpoint. The endpoint is now ready to participate in Tivoli management operations.

### 7.5.2 Installing the TMA

The Tivoli Management Agent (TMA) is included as part of the Feature for Windows Clients and improves the integration of WorkSpace On-Demand and the Feature for Windows Clients with the Tivoli Enterprise Management software.

WorkSpace On-Demand customers do not have to obtain the Tivoli Management Agent code separately from Tivoli, making it easier for customers to implement the Tivoli management software across their enterprise.

The IBM WorkSpace On-Demand 2.0 Feature for Windows Clients install program installs the TMA files in the \$TIVOLI directory for each operating system type on the boot server. The TMA install program would be added to the list of OEM software to be installed for each operating system response file provided. (TMA is installed on clients through commands in the default response files MSBATCH.INF for Win9X clients or CMDLINES.TXT for Windows NT clients.)

There are two sets of client agents:

- Client agent for Windows NT 4.0
- Client agent for Windows 9x (95 and 98)

Each of these client agents have their own installation program, setup.exe. These agents can be manually installed on the client by typing: `setup`

Alternatively, they are classically installed in an unattended mode by involving the setup program with the `-s` option:

`Setup -s <path to response file>`

The above command invokes this unattended install.

By default, the Feature for Windows Clients installs the Tivoli Management Agent Version 3.6. The directory trees for the Windows client TMA on the boot server are as follows:

IBMLAN\RPL\W95\C\TIVOLI for Windows 95  
\\IBMLAN\RPL\W98\C\TIVOLI for Windows 98  
\\IBMLAN\RPL\NT40\I386\SOEM\C\TIVOLI for Windows NT

By default, the TMA software is not installed on clients. If you want to use TME 10 products, you need to modify the default response file and uncomment a line on the MSBATCH.INF file on Windows 9X clients or the CMDLINES.TXT on Windows NT clients, and the TMA will be installed.

To install the TMA on Windows 95 clients:

1. Change to the X:\IBMLAN\RPL\W95\OS\RSP. (X represents the drive letter on the boot server where you installed your Win 95 operating system.)
2. Open the MSBATCH.INF file.
3. At the bottom of the file, uncomment (remove the ; from) the following line under section [LOGONCLT.REG]:

```
HKLM, "Software\Microsoft\Windows\CurrentVersion\RunOnce", TMA, , "C:\TIVOLI\TIVOLI.BAT"
```

To install the TMA on Windows 98 clients:

1. Change to the X:\IBMLAN\RPL\W98\OS\RSP. (X represents the drive letter on the boot server where you installed your Win 98 operating system.)
2. Open the MSBATCH.INF file.
3. At the bottom of the file, uncomment (remove the ; from) the following line under section [LOGONCLT.REG]:

```
HKLM, "Software\Microsoft\Windows\CurrentVersion\RunOnce", TMA, , "C:\TIVOLI\TIVOLI.BAT"
```

To install the TMA on Windows NT clients:

1. Change to the X:\IBMLAN\RPL\NT40\I386\SOEM\$. (X represents the drive letter on the boot server where you installed your Win NT operating system.)
2. Open the CMDLINES.TXT file.
3. At the bottom of the file, uncomment (remove the ; from) the following line under section [Commands]:

```
"C:\TIVOLI\SETTIVOLI.CMD"
```



The directory of the TMA agent for Windows NT is C:\Program Files\Tivoli\lcf, and the location of the lcf daemon is c:\Program Files\Tivoli\lcf\bin\w32-ix86\mrt

### 7.5.3 Configuration of the clients

To configure your clients with the correct port and address information to connect to Tivoli gateways, update the setup files for the Tivoli Management Agent (TMA). To configure the Tivoli Management Agent:

1. Log on to the boot server as an administrator.
2. Change directories to one of the following locations for:
  - Windows NT clients: \IBMLAN\RPL\NT40\I386\%OEM%\C\%TIVOLI
  - Windows 95 clients: \IBMLAN\RPL\W95\C\%TIVOLI
  - Windows 98 clients: \IBMLAN\RPL\W98\C\%TIVOLI

Open the file SETUP.ISS in any ASCII text editor.

3. Find the following section:

```
[SdShowDIgEdit3-0]
szEdit1=9494
szEdit2=9494
szEdit3=d1
Result=1
```

Change the settings to represent the port and IP address of your Tivoli gateway. For example, to configure a client with port 9999 and an IP address of 9.3.169.88, type:

```
[SdShowDIgEdit3-0]
szEdit1=9999
szEdit2=9999
szEdit3=-g 9.3.169.88+9999
Result=1
```

See the documentation for Tivoli Framework 3.6 for more information.

### 7.5.4 Stopping and starting the TMA

Occasionally, you may need to manually stop or restart an endpoint. These are the manual procedures:

For Windows NT:

- From the command line:

Use the `net start lcf` or `net stop lcf` command to start or stop NT endpoints.

- From the desktop:

Select **Control Panel -> Services**, then start or stop the Tivoli endpoint service.

To connect the TMA to a specific gateway:

- From the command line:

Use the `lcfcd start -g Gateway_IP_Address` command.

For example: `lcfcd start -g 9.12.12.42`

- From the desktop on Windows NT:

Select **Control Panel -> Services**, then start the Tivoli endpoint service with the -g Gateway\_IP\_Address parameter.

To remove an endpoint on a Windows NT/95/98 computer:

1. Stop and remove the Endpoint service by issuing the following command:

```
C:\Program Files\Tivoli\lcf\unist
```

2. Follow the instructions on screen.

---

## Appendix A. Comparing the Feature for Windows Clients

To position the Feature for Windows Clients, you need to understand the positioning of multiuser Windows application servers as well as the Windows Terminal Server concept. We compare these architectures with the Feature for Windows Clients so that, based on their individual merits, you can select a solution suitable to your environment.

This is not meant to be an in-depth document on any of the individual products or technologies. That would warrant a book of its own. However, we chose to take a corporate, administrative view of each of the architectures and look at the following features:

- Management of user environments
- Desktop security
- Application model characteristics

In the sections that follow, we describe the characteristics of some of the available technologies. We position WorkSpace On-Demand 1.0 and 2.0 against the Microsoft Windows Terminal Server product and the Citrix MetaFrame application server.

---

### A.1 Introducing the alternatives

The information provided in this section is extracted from different documentations available through sources on the Internet and provides only very basic information. For more information on individual topics, you should check the following Web sites:

- Microsoft Corporation  
[www.microsoft.com](http://www.microsoft.com)
- Wyse  
[www.wyse.com](http://www.wyse.com)
- Citrix  
[www.citrix.com](http://www.citrix.com)

#### A.1.1 Microsoft Windows Terminal Server

The Microsoft Windows NT Server, Terminal Server Edition, Version 4.0 gives the Windows NT Server operating system the capability to serve the 32-bit Microsoft Windows operating system-based applications to terminals and terminal emulators running on PC and non-PC desktops. The Terminal Server

environment is, by definition, a thin-client architecture where all application processing occurs centrally on the server.

The Microsoft Windows NT Server 4.0, Terminal Server Edition, adds Windows-based terminal support to the Windows NT server and a *super-thin client* to the Windows operating system family product line. This new technology provides enterprise customers with a new extension to the Windows-based computing environment that combines low total cost of ownership and the familiar 32-bit Windows user interface.

Terminal Server is an extension of the Windows NT Server 4.0 product line. With a super-thin client in the multiuser Windows NT operating system, users can experience the Windows NT desktop operating system and Windows-based applications completely off the server. Windows Terminal Server will provide users access to 16- or 32-bit Windows-based applications from any of the following types of desktops:

- A new class of low-cost hardware, commonly referred to as Windows-based terminals, that will be marketed by third-party hardware vendors.
- Any existing 32-bit Windows desktop operating system, such as Windows 95 or Microsoft Windows NT Workstation (running the 32-bit Terminal Server client as a window within the local desktop environment).
- Older 16-bit Windows-based desktops running the Windows 3.11 operating system (running the 16-bit Terminal Server client as a windows within the local desktop environment).
- X-based Terminals, Apple Macintosh, MS-DOS/Æ, networked computers, and UNIX-based desktops (by way of a third-party add-on product).

Microsoft Windows NT Server 4.0, Terminal Server Edition consists of three components—the Windows NT Server multiuser core, the Remote Desktop Protocol (RDP), and the super-thin Windows-based client software:

- Terminal Server  
A multiuser server core that provides the ability to host multiple, simultaneous client sessions on Windows NT Server 4.0. Terminal Server is capable of directly hosting compatible multiuser client desktops running on a variety of Windows-based and non-Windows-based hardware.
- Remote Desktop Protocol (RDP)  
A key component of the Terminal Server is the protocol that allows a super-thin client to communicate with the Terminal Server over the network. This protocol is based on International Telecommunications

Union's (ITU) T.120 protocol. It is tuned for high-bandwidth enterprise environments and also supports encrypted sessions.

#### Protocol Restriction

Although RDP is designed to support many different types of network topologies and many LAN protocols, such as IPX, Netbios, TCP/IP, and so forth, the current version of the RDP implementation in the Terminal Server only runs over TCP/IP.

- **Super-Thin Client**

The client software that presents, or displays, the familiar 32-bit Windows user interface on a range of desktop hardware is as follows:

- New Windows-based terminal devices
- Personal computers running Windows 95, Windows 98, or Windows NT Workstation
- Personal computers running Windows for Workgroups (Windows Version 3.11)

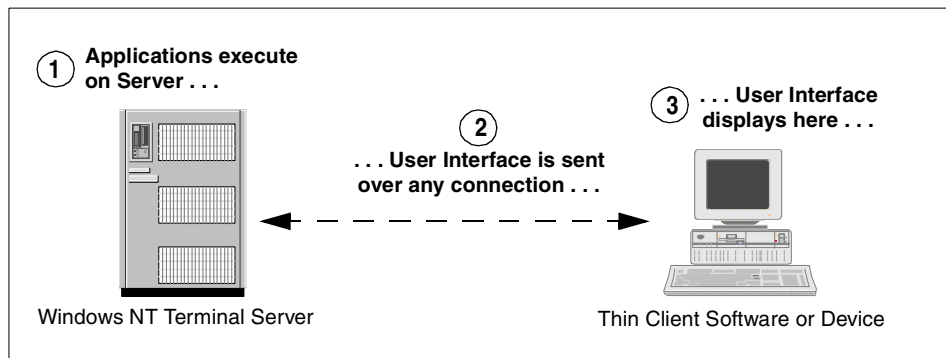


Figure 52. Data flow in the Terminal Server

If you need more technical information on Microsoft products, check their Internet home page.

### A.1.2 Citrix MetaFrame

With its WinFrame product, Citrix Systems had the first multiuser Windows NT technology-based solution, named MultiWin, available on the market. This solution was based on Windows NT 3.51 and is comparable to Microsoft Windows NT Server 4.0, Terminal Server Edition.

The second key technology Citrix has developed is the *Independent Computing Architecture* or *ICA*. On the server, ICA has the unique ability to separate application logic from the user interface. On the client, users see and work with the application's interface, but 100 percent of the application executes on the server. With ICA, applications consume as little as one-tenth of their normal network bandwidth.

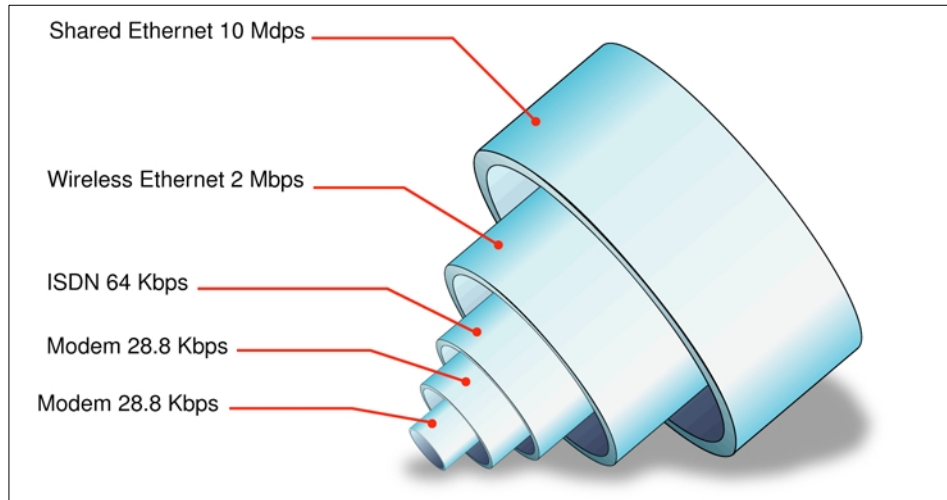


Figure 53. ICA protocol bandwidth

ICA is optimized for connections as low as 14.4 kbps. Only mouse clicks, keystrokes, and screen updates travel the network to generate exceptional performance.

Since Microsoft Windows NT Server 4.0, Terminal Server Edition only supports Microsoft Windows clients, you would choose to add Citrix MetaFrame if you want to add further client support:

- Windows NT, 9x, 3.x, WfW 3.x, WinCE
- DOS, OS/2
- JAVA
- Netscape Navigator (Plug-in)
- Microsoft Internet Explorer (Active X)
- Windows based Terminals
- Macintosh computers
- UNIX workstations

On May 12, 1997, Citrix and Microsoft signed a joint development and marketing agreement, whereby Microsoft licensed the MultiWin, multiuser NT

technology from Citrix and simultaneously endorsed the Citrix Independent Computing Architecture (ICA). The impact of this agreement is significant in that it recognizes the broad customer demand for the server-based computing model. The agreement also enables Microsoft and Citrix to work closely together to develop multiuser software solutions for Windows NT 4.0, Windows 2000 Server, and beyond. The jointly developed multiuser Terminal Server Edition for Windows NT 4.0 provides a platform for Citrix to offer MetaFrame software, a robust server-based system software that takes enterprise computing even further.

### **A.1.3 The MetaFrame architecture**

The MetaFrame product is the Citrix system software for Microsoft Windows NT Server 4.0, Terminal Server Edition. Citrix MetaFrame software incorporates Citrix Independent Computing Architecture and brings server-based computing to the entire enterprise, extending the Terminal Server Edition by providing access to any device, to any location, over any connection, and with any application. MetaFrame offers IT professionals a cost-effective way to deploy, manage, and support applications from a single point. It provides universal application access from virtually any type of client device.

It ensures bandwidth-independent performance with any type of network protocol or connection. It also offers unique features for enhanced application management and security.

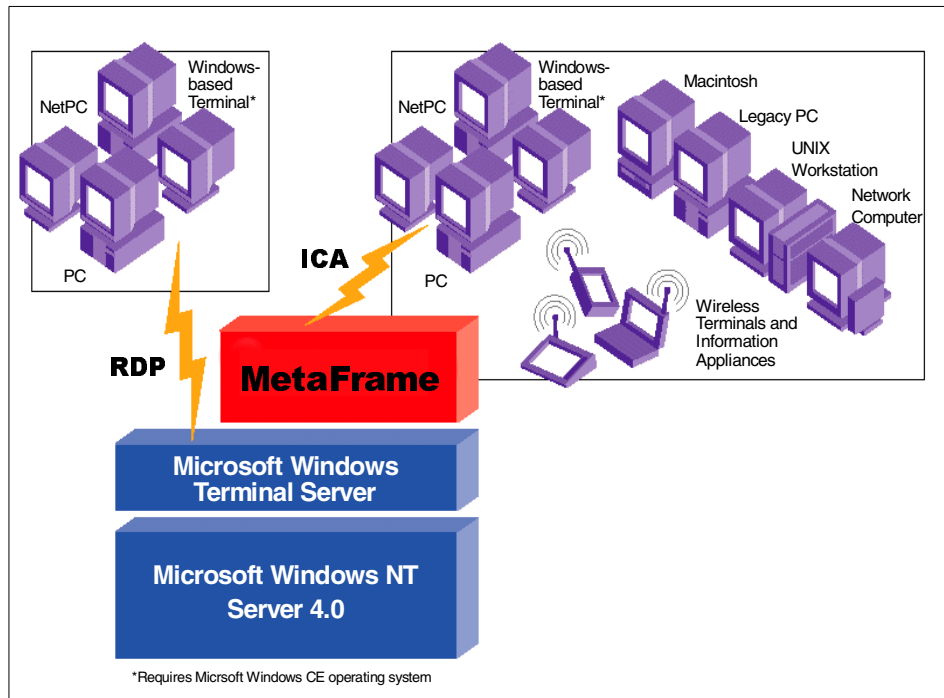


Figure 54. Citrix MetaFrame architecture

With this innovative software, enterprises are able to:

- Bring server-based computing to heterogeneous computing environments, providing access to Windows-based applications regardless of client hardware, operating platform, network connection, or LAN protocol. This enhances the TCP/IP-only protocol of Microsoft Windows NT Server 4.0, Terminal Server Edition that only serves Windows based terminals. For companies with multiple networks and file servers, MetaFrame software supports all popular LAN and WAN protocols, including TCP/IP, IPX, SPX, NetBIOS, and direct asynchronous connections for LAN as well as WAN links (T1, T3, 56Kb, X.25), broadband connections (ISDN, Frame Relay, ATM), wireless, and CDPD connections, making it a convenient and efficient solution for enterprise-wide application deployment.
- Offer enterprise-scale management tools, allowing IT professionals to scale, deploy, manage, and support applications from a single location. These tools let you add servers without having to reconfigure user systems, and applications can be administered across multiple servers from a single location. If you need a failsafe or most responsive server



environment, you can group multiple MetaFrame servers into a unified single-server image by using the Citrix Load Balancing Services.

- Combine seamless integration of the user's local and remote resources and applications with exceptional performance.

If you need more technical information on Citrix products, check their Internet home page.

---

## A.2 Computing architectures, a quick comparison

Workspace On-Demand is a product based on the network computing model. The aim is to implement a thin client environment to reduce the total cost of ownership as well as the end user complexity. However, it attempts to use as much of the local desktop's processor power as possible to reduce server and/or network load.

There are two other computing models: the client/server and server-based computing models. While all three computing models have a valid role in today's enterprises, it is important to note the differences between them.

As you know, in the traditional client/server architecture, processing is centered around local execution using fat, powerful hardware components.

In the network computing architecture, as defined by Sun, Oracle, Netscape, IBM, and Apple, components are dynamically downloaded from the network into the client device for execution by the client.

With the server-based computing approach, users are able to access business-critical applications, including the latest 32-bit Windows-based and Java applications, without requiring them to be downloaded to the client.

Table 21 gives an overview of the three architectures.

*Table 21. Comparing application model architectures*

	<b>Traditional Client/Server</b>	<b>Network Computing</b>	<b>Server-Based Computing</b>
<b>Processing Model</b>	Local Installation Local Execution	Download and Local Execution	Server Execution
<b>Client Hardware Footprint</b>	Fat	Fat or Thin	Fat or Thin

	<b>Traditional Client/Server</b>	<b>Network Computing</b>	<b>Server-Based Computing</b>
<b>Application Architecture</b>	Two or Three-Tier Client/Server	Components, Objects	Monolithic, Components, or Client/Server
<b>Target Application Type</b>	Windows, OS/2, proprietary	Windows, OS/2, proprietary, or JAVA	Windows, OS/2, proprietary

### **A.2.1 Why bother?**

Given the availability of high powered personal computers at extremely low prices, many may ask why even bother with remote boot mechanisms or remote application execution. The point is that the initial cost for buying hardware and software makes up only one third of a machine's total cost over one year. Most of the costs are based on ongoing hardware and software maintenance and user support.

Other factors may come into this. For example, in a client/server model, users have the ability to store data locally. This has an associated cost should the data be lost. When you enforce policies, you begin to change your computing model.

Selecting a model is highly dependent on the computing environment. In an environment where the volume of transactions is high and the income from each transaction is low, such as bank tellers, you need to keep the costs down. In other environments, you may have a low number of high value transactions. In this environment you could possibly afford to spend more. Generally, the more you can save, the lower your overhead, and the higher your profits.

In the following sections, we take the view that we will be working in a high volume, low value transaction environment.

---

## **A.3 The total cost of ownership**

Total Cost of Ownership (TCO) is based on a few simple principles. While the principles may be simple, turning them into actions is not always easy.

### **A.3.1 Centralize as many resources as possible**

You can maximize your savings by centralizing as many resources on as few a servers as possible. This is often called server consolidation.

Large site installations may have one or more domains with many users relying on access to the data and applications on these servers. In such cases, you may need to provide high availability by setting up more than one server for each function.

In smaller branch offices, you will find only one server, in most cases. Here, it is much easier to manage one remote server than to manage many locally installed client machines.

As a rule of thumb, data and applications should be the first items to be removed from the client machines and consolidated on the server. Workspace On-Demand has the complete operating system stored and administered on the server. This comes together with central user desktop management that gives you the roaming capability while still using the local processor and memory of your remote systems to keep the load profile on the server as low as possible.

Using Workspace On-Demand 1.0 or 2.0, you don't even need a local hard drive enhancing the availability of the client systems.

Microsoft Windows NT Server 4.0, Terminal Server Edition gives you two choices.

The first uses full-blown PCs with locally installed versions of Microsoft's operating systems on the hard drive that are operated on the local processor and memory. You can have central user profiles on the logon server to enable roaming users, but you need to locally install and maintain all the clients. User profiles also are dependent on the version of the OS that is installed on the client. For example, a user with a Windows 9x profile cannot roam to a machine with Windows NT installed unless he or she also has a user profile defined for NT.

The second possibility is to install new Windows-based terminals that are based on a small local OS, such as Windows CE, that is only used for communication services to establish connections to a server. All processing is done completely on the server. You need additional user profiles for those kinds of machines if you wish to integrate them in a mixed environment.

In both cases, applications and data are stored on the server. Furthermore, all applications are executed on the server. This may, depending on the application, require a significantly larger server. For example, if you have 200 branch offices with 15 to 20 users each, you will require a multiuser, multiprocessor server in each branch.

The Citrix MetaFrame solution has much the same architecture but also offers more than one application server for availability and failover

reasons as well as the need to support other clients than Microsoft Windows clients by using the ICA protocol.

If you rarely need access to Windows 32-bit applications, and your WAN environment gives you good bandwidth (64 Kbit min. / 2 Mbit and greater perfect), you can even consider a central Windows application server, such as Citrix MetaFrame, to allow users access to Windows applications. This is a solution for a large number of branch offices when you are not using Windows applications extensively.

### **A.3.2 Reduce end user complexity**

Reducing the complexity for the end user essentially reduces your support costs, which in turn, reduces your TCO. Seventy to eighty percent of transaction-oriented business users only care about their tasks. They don't want to be bothered with local administration of a user interface, attaching the new LAN printer, or configuring an application. They want to handle phone orders, transfer values between accounts, or calculate mortgages.

WorkSpace On-Demand is designed to provide and centrally manage simple interfaces for end users by default. Each logon is coupled with the appropriate end user's desktop on whatever machine they are using.

You can do similar things within the Windows environment depending on the components you are using. Windows NT provides a very secure way of forcing end users to log on to a domain server while preventing any local logons. With Windows 9x systems, you need additional components distributed to all clients to provide a similar security.

### **A.3.3 Exploit current resources**

Extending the life of existing hardware and software, without affecting functionality available, means that you can save money. Spending associated with development, initial capital costs, and so on can be delayed. With the current trend of more for your money later, you can always buy later.

With WorkSpace On-Demand, you can keep much of your local hardware and applications while making the transition to a new environment.

---

## **A.4 Application architecture models**

Today's client/server applications are generally programmed to a platform-specific API or use proprietary infrastructures. If you are planning to move away from this enclosure and adopt an open programming model, such as Java, you need to ensure that you invest in the right infrastructure.

With the Java Component Model, objects can reside any place in the network. These distribution and maintenance mechanisms are completely different from the static client/server applications of today.

Should you choose this way, you need to ensure that your applications are 100 percent pure Java. Only these applications will run on any client platform today and in the future. There are already different types of end user devices, such as PDAs (PSION, NOKIA, PalmPilot, and so forth), cellular telephones (NOLIA, Ericson, Philips, Nortel, and so forth), and WebTV devices (Philips, Sony, and so forth), that give users access to Java applications but cannot use platform proprietary Java code, such as Windows Foundation Classes (WFC) or ActiveX controls. Therefore, be aware and keep your applications as open as possible.

Once your mission critical applications are available in Java in your enterprise, you are free to choose any client device available.

---

## A.5 Summary

There are many different approaches and many emerging technologies and each have their own inherent strengths and weaknesses. Usually, a choice is made based on costs, applications, risks, and rewards.

WorkSpace On-Demand provides many benefits over the other choices. It gives you better control over your end users' desktop environment that reduces complexity for most users and results in a lower TCO.

WorkSpace On-Demand fully exploits your desktop PCs and often does not require any additional server hardware, which protects your current investments.

You can now flawlessly integrate Windows clients within your WorkSpace On-Demand environment. Servicing all your WorkSpace On-Demand clients from one server and providing the desktop operating system of choice helps you keep the TCO low since you don't need to deploy additional servers.

All platforms serviced within WorkSpace On-Demand give you full 100 percent pure Java client-side operating system environments. This helps you to begin implementing new enterprise application models.



---

## Appendix B. Supported network adapter list

This is a list of network adapters that are supported for the IBM WorkSpace On-Demand 2.0 Feature for Windows Clients. However, new network adapters are added to this list periodically. For the latest list of supported adapters for WorkSpace On-Demand, see

<http://www.software.ibm.com/enetwork/workspace/about/adapters.html>

---

### B.1 Windows NT Workstation supported adapter list

- IBM Turbo 16/4
- IBM Turbo 16/4 w/WOL
- IBM 16/4 Auto
- IBM 16/4 Auto w/WOL
- 3COM 3C900-Combo
- 3COM 3C905B-TX
- 3COM 3C905-TX
- 3COM 3C905B-TX-NM
- IBM 10/100 Etherjet PCI WOL
- Madge T/R 16/4 Plus
- 3COM3C905-T4
- 3COM3C509B-TP
- Intel EtherExpress PRO/100
- SMC 8216C
- SMC 9432TX
- IBM ETHERJET 10BASE-T ISA
- Kingston Technology KNE100TX
- 3COM3C509-TP
- 3COM3C509B-TP
- 3COM3C509-TPO
- IBM TokenRing PCI Adapter
- 3COM 3C619B Tokenlink
- IBM 100/16/4 TokenRing PCI

- 3COM3C509B-Combo
- IBM LanStreamer PCI
- Olicom OC3137
- 3COM 3C319 Tokenlink
- Madge Smart 16/4 ISA Client +

---

## **B.2 Windows 95 supported adapter list**

- IBM Turbo 16/4
- IBM Turbo 16/4 w/WOL
- IBM 16/4 Auto
- IBM 16/4 Auto w/WOL
- 3COM 3C900-Combo
- 3COM 3C905B-TX
- 3COM 3C905B-TX-NM
- IBM 10/100 Etherjet PCI WOL
- Madge T/R 16/4 Plus
- 3COM 3C905-TX
- 3COM 3C905-T4
- 3COM 3C509B-TP
- SMC 8216C
- Intel EtherExpress PRO/100
- SMC 9432TX
- IBM ETHERJET 10BASE-T ISA
- Kingston Technology KNE100TX
- 3COM3C509-TP
- 3COM3C509B-TP
- 3COM3C509-TPO
- Madge Smart 16/4 ISA Client +
- IBM TokenRing PCI Adapter 2
- 3COM 3C619B Tokenlink
- IBM 100/16/4 TokenRing PCI



- 3COM3C509B-Combo
- IBM LanStreamer PCI
- Olicom OC3137
- 3COM 3C319 Tokenlink
- Madge Smart 16/4 ISA Client +

---

### **B.3 Windows 98 supported adapter list**

- IBM Turbo 16/4
- IBM Turbo 16/4 w/WOL
- IBM 16/4 Auto
- IBM 16/4 Auto w/WOL
- 3COM 3C900-Combo
- 3COM 3C905B-TX
- 3COM 3C905B-TX-NM
- IBM 10/100 Etherjet PCI WOL
- 3COM 3C905-TX
- Madge T/R 16/4 Plus
- 3COM 3C905-T4
- 3COM 3C509B-TP
- SMC 8216C
- SMC 9432TX
- Intel EtherExpress PRO/100
- IBM ETHERJET 10BASE-T ISA
- Kingston Technology KNE100TX
- 3COM3C509-TP
- 3COM3C509B-TP
- 3COM3C509-TPO
- Madge Smart 16/4 ISA Client +
- IBM TokenRing PCI Adapter 2
- 3COM 3C619B Tokenlink
- IBM 100/16/4 TokenRing PCI

- 3COM3C509B-Combo
- IBM LanStreamer PCI
- Olicom OC3137
- 3COM 3C319 Tokenlink
- Madge Smart 16/4 ISA Client +

---

## Appendix C. IBM Object REXX interpreter

It's easy to get excited about REXX. REXX is a versatile, free-format language. Its simplicity makes it a good first language for beginners. For more experienced users and computer professionals, REXX offers powerful functions and the ability to issue commands to multiple environments.

The most vital role REXX plays is as a programming language for Windows. A REXX program can serve as a script for the Windows operating system to follow. Using REXX, you can reduce long, complex, or repetitious tasks to a single command or program.

Although we have not included any examples here, REXX can be used to customize your users desktops, automate their installs by running scripts in the background, during installation. This language provides intelligent scripting capabilities at no additional costs.

---

### C.1 Features

For those unfamiliar with REXX, the following aspects round out its versatility and function.

#### C.1.1 Object-oriented programming

Object-oriented extensions have been added to traditional REXX, but its existing functions and instructions have not changed. The Object REXX Interpreter is actually an enhanced version of its predecessor but with new support for:

- Classes, objects, and methods
- Messaging and polymorphism
- Inheritance and multiple inheritance

Object REXX supplies the user with a base set of classes. These are ALARM, CLASS, ARRAY, LIST, QUEUE, TABLE, SET, DIRECTORY, RELATION, BAG, MESSAGE, METHOD, MONITOR, STEM, STREAM, STRING, and SUPPLIER. Object REXX is fully compatible with earlier versions of REXX that were not object-oriented.

#### C.1.2 An English-like language

To make REXX easier to learn and use, many of its instructions are meaningful English words. Unlike some programming languages that use

abbreviations, REXX instructions are common words, such as SAY, PULL, IF...THEN...ELSE, DO...END, and EXIT.

### **C.1.3 Fewer rules**

REXX has relatively few rules about format. A single instruction can span many lines, and you can include multiple instructions on a single line. Instructions need not begin in a particular column and you can type them in uppercase, lowercase, or mixed case. You can skip spaces in a line or entire lines. There is no line numbering.

### **C.1.4 Interpreted, not compiled**

REXX is an interpreted language. When a REXX program runs, its language processor reads each statement from the source file and runs it, one statement at a time. Languages that are not interpreted must be compiled into object code before they can be run.

### **C.1.5 Built-in functions and methods**

REXX has built-in functions and methods that perform various processing, searching, and comparison operations for text and numbers and provide formatting capabilities and arithmetic calculations.

### **C.1.6 Typeless variables**

REXX regards all data as objects of various kinds. Variables can hold any kind of object, so you need not declare variables as strings or as numbers.

### **C.1.7 String handling**

REXX includes capabilities for manipulating character strings. This allows programs to read and separate characters, numbers, and mixed input. REXX performs arithmetic operations on any string that represents a valid number, including those in exponential formats.

### **C.1.8 Clear error messages and powerful debugging**

REXX displays messages with meaningful explanations when a REXX program encounters an error. In addition, the TRACE instruction provides a powerful debugging tool.

---

## C.2 Installation

A copy of the IBM Object REXX Interpreter Edition Version 1.0.2 resides on the IBM WorkSpace On-Demand 2.0 Feature for Windows Clients CD. To run REXX scripts from your clients, install the REXX Interpreter on your clients.

To install the REXX Interpreter, create an application package containing the IBM Object REXX Interpreter, as follows:

1. See the section "Creating and Managing Application Packages" in the *WorkSpace On-Demand 2.0 Feature for Windows Clients Administrator's Guide* for directions that explain how to create an application package. You can also refer to Chapter 4, "About application creation and management" on page 83.
2. Install REXX Interpreter by using ORXI1023.EXE. This program resides in the \OBJREXX directory of the IBM WorkSpace On-Demand 2.0 Feature for Windows Clients CD.
3. Delete any entries from the Start menu that are associated with REXX before you complete your application package.
4. Log on to the primary domain controller as an administrator.
5. Change to the \IBMLAN\DCDB\WIN32APP\SERVER directory. Delete the appid subdirectory (appid represents the name of the application package). Copy the client application files to the client image.
6. Do not assign the REXX application package to user accounts. This package runs automatically from clients when you distribute the application files to the client.



---

## Appendix D. Special notices

This publication is intended to help IBM technical personnel, Business Partners, and customers to plan, install, and configure the Feature for Windows Clients on WorkSpace On-Demand 2.0. The information in this publication is not intended as the specification of any programming interfaces that are provided by the Feature for Windows Clients or WorkSpace On-Demand. See the PUBLICATIONS section of the IBM Programming Announcement for the IBM WorkSpace On-Demand 2.0 Feature for Windows Clients for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM ("vendor") products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer

responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

Any performance data contained in this document was determined in a controlled environment, and therefore, the results that may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

The following document contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples contain the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

AS/400	AT
BookManager	CT
EtherJet	Home Director
IBM ®	Network Station
OS/2	PC 300
RS/6000	SP
Streamer	System/390
XT	400

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.



Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

PC Direct is a trademark of Ziff Communications Company in the United States and/or other countries and is used by IBM Corporation under license.

MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and/or other countries licensed exclusively through X/Open Company Limited.

SET and the SET logo are trademarks owned by SET Secure Electronic Transaction LLC.

Other company, product, and service names may be trademarks or service marks of others.



---

## Appendix E. Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

---

### E.1 International Technical Support Organization publications

For information on ordering these ITSO publications see “How to get ITSO redbooks” on page 241.

- *Inside OS/2 LAN Server 4.0*, SG24-4428
- *OS/2 Installation Techniques: The CID Guide*, SG24-4295
- *WorkSpace On-Demand Handbook*, SG24-2028
- *WorkSpace On-Demand Handbook Release 2.0*, SG24-5117
- *OS/2 Warp Server for e-business*, SG24-5393
- *OS/2 Warp CID Software Distribution Guide*, SG24-2010

---

### E.2 Redbooks on CD-ROMs

Redbooks are also available on the following CD-ROMs. Click the CD-ROMs button at <http://www.redbooks.ibm.com/> for information about all the CD-ROMs offered, updates, and formats.

CD-ROM Title	Collection Kit Number
System/390 Redbooks Collection	SK2T-2177
Networking and Systems Management Redbooks Collection	SK2T-6022
Transaction Processing and Data Management Redbooks Collection	SK2T-8038
Lotus Redbooks Collection	SK2T-8039
Tivoli Redbooks Collection	SK2T-8044
AS/400 Redbooks Collection	SK2T-2849
Netfinity Hardware and Software Redbooks Collection	SK2T-8046
RS/6000 Redbooks Collection (BkMgr Format)	SK2T-8040
RS/6000 Redbooks Collection (PDF Format)	SK2T-8043
Application Development Redbooks Collection	SK2T-8037

---

### E.3 Other publications

These publications are also relevant as further information sources:

- *Microsoft's Administrator's Guide, Zero Administration Kit*
- *Microsoft's Deployment Guide to Automating Windows NT Setup*, (White paper), available at: [www.microsoft.com](http://www.microsoft.com)
- *Guide to Microsoft Windows NT 4.0 Profiles and Policies*, (White paper), available at: [www.microsoft.com](http://www.microsoft.com)
- *Mastering Windows NT Server 4, Third Edition*, Sybex Press, ISBN:0-7821-1920-4
- *Microsoft Windows NT Server, Networking Guide*, ISBN:1-5723-1344-7
- *Windows NT 4.0 Registry: A Professional Reference*, ISBN:0-0791-3655-9



---

## How to get ITSO redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, redpieces, and CD-ROMs. A form for ordering books and CD-ROMs by fax or e-mail is also provided.

- **Redbooks Web Site** <http://www.redbooks.ibm.com/>

Search for, view, download, or order hardcopy/CD-ROM redbooks from the redbooks Web site. Also read redpieces and download additional materials (code samples or diskette/CD-ROM images) from this redbooks site.

Redpieces are redbooks in progress; not all redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

- **E-mail Orders**

Send orders by e-mail including information from the redbooks fax order form to:

	<b>e-mail address</b>
In United States	<a href="mailto:usib6fpl@ibmmail.com">usib6fpl@ibmmail.com</a>
Outside North America	Contact information is in the "How to Order" section at this site: <a href="http://www.elink.ibm.link.ibm.com/pbl/pbl/">http://www.elink.ibm.link.ibm.com/pbl/pbl/</a>

- **Telephone Orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	Country coordinator phone number is in the "How to Order" section at this site: <a href="http://www.elink.ibm.link.ibm.com/pbl/pbl/">http://www.elink.ibm.link.ibm.com/pbl/pbl/</a>

- **Fax Orders**

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	Fax phone number is in the "How to Order" section at this site: <a href="http://www.elink.ibm.link.ibm.com/pbl/pbl/">http://www.elink.ibm.link.ibm.com/pbl/pbl/</a>

This information was current at the time of publication, but is continually subject to change. The latest information may be found at the redbooks Web site.

### IBM Intranet for Employees

IBM employees may register for information on workshops, residencies, and redbooks by accessing the IBM Intranet Web site at <http://w3.itso.ibm.com/> and clicking the ITSO Mailing List button. Look in the Materials repository for workshops, presentations, papers, and Web pages developed and written by the ITSO technical professionals; click the Additional Materials button. Employees may access MyNews at <http://w3.ibm.com/> for redbook, residency, and workshop announcements.

## IBM Redbook Fax order form

**Please send me the following:**

Title	Order Number	Quantity
-------	--------------	----------

[illegible]

First name	Last name
------------	-----------

Company	Revenue	Profit	Assets	Liabilities	Equity
Company A	100	20	120	80	40
Company B	150	30	180	120	60
Company C	200	40	240	160	80
Company D	250	50	300	200	100
Company E	300	60	360	240	120
Company F	350	70	420	280	140
Company G	400	80	480	320	160
Company H	450	90	540	360	180
Company I	500	100	600	400	200
Company J	550	110	660	440	220
Company K	600	120	720	480	240
Company L	650	130	780	520	260
Company M	700	140	840	560	280
Company N	750	150	900	600	300
Company O	800	160	960	640	320
Company P	850	170	1020	680	340
Company Q	900	180	1080	720	360
Company R	950	190	1140	760	380
Company S	1000	200	1200	800	400

Address \_\_\_\_\_

City	Postal code	Country
------	-------------	---------

Telephone number	Telefax number	VAT number
------------------	----------------	------------

☐ Invoice to customer number☐ Credit card number

Credit card expiration date	Card issued to	Signature
-----------------------------	----------------	-----------

**We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.**

---

## Glossary

**Additional Server.** A server, other than the domain controller, in an OS/2 Warp Server domain. An additional server may be configured as a backup domain controller and will automatically take over the functions of the domain controller should the domain controller fail.

**Administration Client.** A fat client running OS/2 Warp 4 onto which is installed the WorkSpace On-Demand Administration Client component, thereby allowing an administrator to manage WorkSpace On-Demand servers and clients from the administration client system.

**Administrator.** A person who manages a LAN or group of LANs running OS/2 Warp Server and WorkSpace On-Demand and who is responsible for creating and maintaining client workstation, end user, and application definitions.

**Alias.** A name by which a resource can be referenced and accessed on a LAN. Aliases are normally referenced using the Universal Naming Convention (UNC).

**Application Server.** Server on which network applications are installed for access by end users on client workstations. Note that an application server may also act as a file, print, or boot server.

**Application Roaming.** Concept implemented in WorkSpace On-Demand whereby a user can move from one client workstation to another while maintaining the same application icons and desktop settings.

**Backup Domain Controller.** An additional server running OS/2 Warp Server to which user, resource, and access control information is replicated dynamically from the domain controller. In the event of a system failure at the primary domain controller, the backup domain controller can take over, without the knowledge of the end user.

**BINL Server.** See *Boot Information Negotiation Layer Server*.

**Boot Block.** A set of executable code, device drivers, and data that is downloaded from a boot server to a client workstation in order to begin the remote boot process.

**Boot Block Definition File.** ASCII text file used to define the contents of the boot block.

**Boot Image.** Directory structure on a boot server that contains the clients' operating system files and, optionally, shared application files.

**Boot Information Negotiation Layer Server.** Server that provides the IP address of a server containing a boot block to a client that boots using the DHCP PXE boot mechanism.

**Boot ROM.** A read-only memory module installed on a network adapter that allows the machine in which the adapter is installed to boot from a boot server on the network.

**Boot Server.** A server running an OS/2 Warp Server and WorkSpace On-Demand Server that is used to remotely boot client workstations.

**Boot Storm.** A scenario whereby a large number of client workstations are simultaneously booting from a boot server, creating a heavy demand on the server and network resources. A boot storm typically occurs as a result of a power outage, after which all clients boot simultaneously when power is restored.

**Client.** See *Client Workstation*.

**Client Workstation.** A physical machine, typically a personal computer, that is used by an end user to perform business tasks. In a WorkSpace On-Demand environment, a client (also known as a client workstation) loads its operating system from a WorkSpace On-Demand server.

**CNF File.** See *Boot Block Definition File*.

**Domain.** A group of servers running OS/2 Warp Server that share their resources for use by end users on client workstations. Resources in a domain can typically be accessed using only a

resource's alias, without regard to the particular server on which the resource resides.

**Domain Controller.** The primary server in an OS/2 Warp Server domain that authenticates end users' and client workstations' access requests for resources.

**Domain Name Server.** A system on a TCP/IP network that resolves hostnames to IP addresses.

**Dynamic Host Configuration Protocol.** A TCP/IP-based protocol that allows a client workstation to obtain an IP address and other related information dynamically at boot-time rather than having the information statically defined when the client is defined. See *Dynamic Domain Name Services*.

**Dynamic Domain Name Services.** A TCP/IP-based protocol that allows a client workstation running TCP/IP to automatically update its hostname when it obtains a dynamic IP address using the Dynamic Host Configuration Protocol.

**End User.** A person who uses a client workstation to perform business tasks.

**Fat Client.** A client workstation, typically a personal computer, running its own self-contained operating system environment. A fat client may run in a stand-alone mode, or may be connected to a network.

**Feature Installer.** Standard Java-based installation procedure used by IBM software products, including WorkSpace On-Demand. Feature Installer allows installation from local media, such as disk or CD-ROM, or remote installation over a network.

**File Index Table.** A table used to redirect file access requests from a client workstation's logical boot drive to the appropriate location on the boot server. Commonly known as a FIT or FIT file. There are two distinct types of file index tables: the *machine FIT* and the *user FIT*.

**File Server.** A server that shares its disk space on a LAN for access by end users on client workstations. Note that a file server may also act as an application, print, or boot server.

**FIND Frame.** A broadcast data frame sent by a client workstation, requesting a reply from any server(s) that are authorized to remotely boot that client. See *FOUND Frame*.

**FixPak.** A series of program updates and bug fixes for a piece of software, such as OS/2 Warp Server.

**FOUND Frame.** Network data frame returned by a boot server to a specific client workstation acknowledging that the server is authorized to boot that client. See *FIND Frame*.

**Hostname.** An alphanumeric name by which a node on a TCP/IP network is identified to other nodes on the network. A domain name server (DNS) is used to resolve a hostname into an IP address for use by TCP/IP-based applications.

**HPFS386.** A 32-bit file system implemented by OS/2 Warp Server Advanced and OS/2 Warp Server Advanced with SMP. HPFS386 provides enhanced file system performance and is recommended for a WorkSpace On-Demand server.

**IP Address.** The address of a node on a TCP/IP network, typically expressed in dotted decimal notation, for example, *nnn.nnn.nnn.nnn*, where *nnn* is a decimal number in the range 0 to 255.

**Java.** Platform-independent programming language and application execution environment developed by Sun Microsystems and supported by WorkSpace On-Demand.

**Locally Administered Address.** Adapter address for a network adapter defined by a network administrator and included in the boot block definition of a WorkSpace On-Demand client. See *Universally Administered Address*.

**Machine Class.** Set of device drivers and other hardware-specific files that are required to support a particular hardware configuration. Each client workstation in a WorkSpace On-Demand environment must belong to a particular machine class from which it derives its hardware-specific operating system components.

**Machine Classing.** The act of creating a new machine class.



**Machine Class Directory.** The subdirectory tree on the WorkSpace On-Demand server that contains the files that constitute a machine class. See *Machine Class File Structure*.

**Machine Class File Structure.** The combination of a machine class directory and the files that it contains. See *Machine Class Directory*.

**Machine FIT.** File index table that contains redirection entries for operating system files and for those application files that are shared by all users of the application.

**Managed PC.** A personal computer connected to a network and managed from a central location. A managed PC typically loads its operating system environment and applications from a server on the network. A WorkSpace On-Demand client workstation is a typical managed PC.

**Mutual Failover.** Situation whereby two or more servers provide automatic backup for one another. In WorkSpace On-Demand, this feature is provided automatically as part of OS/2 Warp Server's RIPL architecture whenever a client workstation is defined to two or more boot servers; it does not need to be explicitly configured.

**NDISDD.PRO File.** ASCII file used when defining a client workstation to determine the contents of the Network Adapter drop-down list on the Hardware page of the Client Definition notebook. NDISDD.PRO references other files, such as the boot block definition file, RPL.MAP, and the network adapter's NIF file.

**Network Application.** An application that is installed on a server and accessed by one or more end users on client workstations on a LAN.

**Network Name.** The name given to a server or client workstation on the network, by which it is identified to other nodes on the network.

**Network Resource.** See *Resource*.

**Node.** A physical device, such as a server, client workstation, or network-attached printer that is directly attached to a network.

**OS/2 Warp Server.** A 32-bit, Intel-based network server operating system.

**PMLOGON Shell.** A simplified version of the OS/2 Workplace Shell implemented by default on a WorkSpace On-Demand client workstation.

**Preboot Execution Environment (PXE).** Also known as *Network Service Boot*. A specification developed by Intel Corporation that defines an interface and API to allow an operating system and/or application software to be loaded onto a system from a remote server. PXE is implemented as a Boot PROM on a network adapter card.

**Print Server.** A server that shares its attached printers on a LAN for access by end users on client workstations. Note that a print server may also act as an application, file, or boot server.

**Public Application.** See *Network Application*.

**Reference Client.** A network client workstation running OS/2 Warp 4 that is used to determine the correct file locations and environment settings when installing network applications.

**Remote Boot Service.** See *RIPL Service*.

**Remote IPL.** The process by which a client workstation obtains its operating system code across a network from a boot server.

**Resource.** An item, such as a directory, printer, or serial device, that can be shared by a server and accessed by client workstations on a LAN.

**RIPL Server.** See *Boot Server*.

**RIPL Service.** Component of OS/2 Warp Server that supports remote IPL. The RIPL service is not installed by default and must be explicitly selected as an installation option when you install OS/2 Warp Server.

**RPL.MAP File.** ASCII file used by a boot server to determine which client workstations are authorized to obtain their operating system image from that server and the image that is to be supplied to each client.

**RPL Module.** See *Boot ROM*.

**RPL Service.** See *RIPL Service*.

**Segment.** A single local area network in which network traffic passes freely between attached

systems without the need to pass through a bridge or router.

**Server.** A machine running OS/2 Warp Server that shares its resources on a LAN for use by end users on client workstations. When running OS/2 Warp Server, a server may be defined as a domain controller or an additional server.

**Share.** The name given to a network resource that identifies it on the network.

**Shared Resource.** See *Resource*.

**Software Choice.** Online subscription service offered by IBM, whereby customers can download new and updated software from the Internet.

**Thin Client.** A client workstation that loads its operating system environment and applications across a network from a server. The degree of local processing power in a thin client may vary considerably depending upon the implementation of the thin client concept.

**Traditional PC.** See *Fat Client*.

**Universal Naming Convention.** System of nomenclature whereby a network alias is referenced by its server name, followed by the alias name within the server. For example, an alias might be \\RPLSRVR\WRKFILES, where RPLSRVR is the server name and WRKFILES is the alias name.

**Universally Administered Address.** The adapter address of a network adapter that is burned in when the adapter is manufactured. A universally administered address consists of a twelve-digit hexadecimal number. See *Locally Administered Address*.

**User.** See *End User*.

**User FIT.** File index table that contains redirection entries for operating system and/or application files that are unique to a particular end user.

**Wake On LAN.** Term used to describe an adapter that inserts itself into a network at a predetermined time each day. A Wake On LAN adapter typically works in conjunction with a machine's BIOS to turn the machine on and

initiate the remote boot process. The insertion time is programmable so that different machines can stagger their boot process and thereby avoid a boot storm situation. See *Boot Storm*.

**Workplace Shell.** Object-oriented user interface implemented in OS/2 Warp and OS/2 Warp 4. By default, WorkSpace On-Demand implements a simplified version of the Workplace Shell, known as the PMLOGON shell.

**WorkSpace On-Demand. (1)** A set of management utilities that enables an OS/2 Warp Server server to remotely load a thin client operating system, known as WorkSpace On-Demand Client, into a client workstation across a LAN. **(2)** The client workstation component of WorkSpace On-Demand that is loaded into a client workstation from a server machine running OS/2 Warp Server and WorkSpace On-Demand Server.

**WorkSpace On-Demand Manager.** The server component of WorkSpace On-Demand that is installed on top of OS/2 Warp Server. See *WorkSpace On-Demand Client*.

**WorkSpace On-Demand Server.** A server running OS/2 Warp Server and WorkSpace On-Demand that is used to boot client workstations.

---

## List of abbreviations

<b>API</b>	Application Programming Interface	<b>MAC</b>	Media Access Control
<b>BINL</b>	Boot Information Negotiation Layer Server	<b>NDIS</b>	Network Driver Interface Specifications
<b>BIOS</b>	Basic Input/Output System	<b>NIC</b>	Network Interface Card
<b>CID</b>	Configuration, Installation, and Distribution	<b>NTFS</b>	NT File System
<b>CLI</b>	Command Line Interface	<b>OEM</b>	Original Equipment Manufacturer
<b>DCDB</b>	Domain Control Database	<b>PXE</b>	Preboot Execution Environment
<b>DHCP</b>	Dynamic Host Configuration Protocol	<b>RDP</b>	Remote Desktop Protocol
<b>DNS</b>	Domain Name Server	<b>REM</b>	Remarkd Lines
<b>DOS</b>	Disk Operating System	<b>REXX</b>	Restructured Extended Executor Language
<b>FIT</b>	File Index Table	<b>RIPL</b>	Remote Initial Program Load
<b>GUI</b>	Graphical User Interface	<b>RPL</b>	Remote Program Load
<b>HPFS</b>	High Performance File System	<b>SDM</b>	Software Distribution Manager
<b>IBM</b>	International Business Machines Corporation	<b>SMP</b>	Symmetric Multiprocessors/ Multiprocessing
<b>ICA</b>	Independent Computing Architecture	<b>TCO</b>	Total Cost of Ownership
<b>IPL</b>	Initial Program Load	<b>TMA</b>	Tivoli management agent
<b>ITSO</b>	International Technical Support Organization	<b>TSR</b>	Terminate-and-Stay-Resident
<b>ITU</b>	International Telecommunications Union	<b>UNC</b>	Universal Naming Convention
<b>JVM</b>	Java Virtual Machine	<b>WCF</b>	Windows Foundation Classes
<b>LAN</b>	Local Area Network	<b>ZAW</b>	Zero-Administration Windows
<b>LCF</b>	Lightweight Client Framework		



---

## Index

### Symbols

\$OEM\$ 142

### Numerics

32-bit Windows applications 3

3COM adapters 165

### A

abbreviations 247

acronyms 247

adapter address 28

adm files 195

administering System Policies 198

administrator policy file 47

Alias 94

Alternatives 215

APPDIR 87

Appearance tab 183

appidINI.INF files 101

appidSYS.INF files 101

appidSYS.REG files 101

Application capture 84

Application commands 119

Application Control Files 128

Application files

client 101

copying 109

server 103

Application ID 178

Application management 17

Application Package Directory 129

Application packages

Overview 85

Application Processing 83

Application selection 89

application server 3

Application support 83

Applications Types 88

Application architecture 224

APPSTORE.INI 103

Assigning applications 115

Assigning desktops 49

Attended installation 62

autoexec.bat 22

### B

Bandwidth 219

BIOS 8, 136

Boot architecture 18

boot block definition file 27, 34, 35

Boot Process 15, 18

Boot Process Options 43

Boot Sequence

Install 19

Operational 24

BootDrive 68

Broad application support 1

### C

Capturing applications 90

CDKEY 76, 132

Centralize 222

centrally manage 205

Changing system policy files 200

Citrix Meta Frame 217

Client agents 211

Client Application Information 129

Client Commands 174

client operating system 72

Client Side Architecture 14

Client-based applications 89

Client-Server 1

cmdlines.txt 78

CNF file 27, 35

CNF file field descriptions 37

co-exist 2

Command Line 17

Command Line Interface 17, 177

command prompt 187

Comments 134

Common desktops 181

COMMON.ADM 196

Complexity 224

Computing architectures 221

config.sys 36

CONNECT.EXE 21

Consistent interfaces 44

Corporate standard 182

Creating applications 90

CRTPKG 95, 96, 124

Crystal Audio 173

Custom bitmap 183

- Customized desktops 17
- Customizing applications 97
- Customizing desktops 179

## D

- Data files 136, 139
- data management 1
- Data stores 128
- Database structures 17
- Datafile 132
- DCDB 54
- Define Application Packages 103
- Defining desktops 181
- Defining multiple workstations 139
- defining workstations 131
- Delete switch 178
- Deleting applications 115, 117
- Desktop appearance 44
- Desktop management 17
- Desktop parameter 178
- Desktop planning 180
- Desktop structure 50
- Desktops 6
- DHCP 132
- DIP switches 8
- Directory Resource 95
- Disk 58
- Disk partitions 133
- Disk space 58
- diskette image 28
- Distributing files 104
- Distributing application files 104
- Distributing Applications 86
- DLSNET 97
- Domain Name 135
- DOS 17
- DOS installation 23
- DOS RIPL 60
- Duplicate IDs 55
- Dynamic Host Configuration Protocol 5

## E

- Easier end user support 1
- Effective software 1
- Emulators 215
- End user mobility 1
- Endpoint 206
- Endpoint Gateway 206

- Endpoint Manager 206
- end-to-end management 205
- Enhanced security 1
- Enumerate applications 118
- Error Log files 69
- Explorer 190

## F

- FAT 9
- FAT32 24
- FDISK 24
- Feature for Windows
  - Architecture 14
- Feature for Windows Clients
  - Application Management 5
  - Application management 85
  - Architecture 5
  - Boot Process 15
  - Components 4
  - Desktop Management 5
  - Installation Roadmap 61
  - Integration 3
  - JVM 16
  - Licensing 10
  - Network Logon Client 15
  - Operating System 15
  - Overview 3, 13
  - Restricted desktop 15
  - Tivoli Management Agent 16
  - User access 85
  - Web Browser 16
  - Windows applications 16
  - Workstation Management 5
- File locations 128
- File systems 57, 82
- FILE.DATA.RESPONSE frames 35
- FIND frame 20
- FIT file 27
- Fixpack 60
- flow chart 191
- FOUND frame 21
- framework 206
- function 41h 23

## G

- graphical interface 4
- Group management 17
- Groups 181

## H

- Hard disk requirements 5
- Hardware 7
  - Client 7
- Hardware device support 10
- Hardware prerequisites 9, 58
- HDBOOT.COM 36
- HDBOOT.COM file 26
- High volume 222
- History Log files 69
- Hives 46
- HKEY\_CURRENT\_USER 103
- HKEY\_LOCAL\_MACHINE 101
- Home directory 53
- Host name 135

## I

- IBM DOS 2000 3
- IBM LAN Support Program 29
- IBM PC 155
- IBM/MS API 23
- ICA 218
- ICW97.INF file 190
- Image file 38
- Image updates 106
- initial desktop 188
- Install environment 55
- install scripts 140
- Install types 62
- Installation roadmap 61
- Intel 1
- Internet Icons 189
- Interpreter 232
- Investment protection 1

## J

- Java Virtual Machine 7, 16, 78
- JFS 82
- JVM Installation 79

## L

- LASTDRIVE 29
- lcf daemon 213
- Licensing 10
- Lightweight Client Framework 206
- links 186
- loader 21

- Local profiles 45
- Log file 42
- Logon client 50
- Logon process 51
- LOGONCLT.BAT 151
- LOGONCLT.REG 212

## M

- MAC address 6, 133
- Machine Classes 155
- Makeimg 38
- MAN files 49
- Mandatory profiles 45
- Memory requirements 59
- MetaFrame 219
- Microsoft Exchange 57
- Microsoft Windows Terminal Server 215
- MODFILE.LST 102
- MSBATCH.INF 79, 136
- Multimedia 173
- multiple endpoint gateways 207
- Multiple server environments 56

## N

- NameServers 133
- Naming Conventions 4
- NDIS 29
- NET ACCESS 58
- NET APIs 49
- NET SHARE 94
- net start 213
- NET.ACC 54
- NETGUI 94
- NETLOGON share 48
- NETWIN 20, 126
- NETWIN APP 104
- NETWIN parameters 132
- NETWIN RIPLMACH 76, 131
- NETWIN USER command 177
- network adapters 159, 227
  - Windows 95 228
  - Windows 98 229
  - Windows NT 227
- Network Interface Card 8
- Network Logon Client 6
- Network Station Manager 3
- New clients 14
- new commands 17

New desktops 181  
NEWFILE.LST 102  
NTCONFIG.POL 48  
NTUSER.DAT 48

## O

Object-oriented programming 231  
OEM devices 43  
OOP 231  
Operating System files 153  
Operating system install 62  
Operating System release 134  
OS/2 Warp Server 73  
OS/2 Warp Server for e-business 55  
other policy files 47

## P

Packaging 10  
Partition 133  
Performance 80  
PKG\_DIR 128  
Planning desktop 180  
Platforms supported 4  
poledit.exe 193  
Policy 191  
policy file format 194  
port 213  
Post-Install 71  
Primary Domain Controller 56  
Printer ports 133  
Printer response files 169  
Printers 133  
Printing 165  
Processor requirements 58  
Profiles  
    Local 45  
    Mandatory 45  
    Roaming 45  
Protocol support 169

## Q

Query client 176

## R

Record identifier 134  
Registered user 134  
registry changes 100

registry keys 96  
Reinstall 69  
Remote boot 8  
Remote commands 177  
Remote Desktop Protocol 216  
Remote Program Load 5  
remove endpoint 214  
Rename.txt file 147  
RENFIL.BAT 102  
Response files 42, 69, 135, 155  
Restricted desktop 16  
REXX 231  
REXX installation 233  
RIPL Support 8  
RISC-based Network Stations 3  
Roaming profiles 45  
RPL control files 26  
RPLBOOT.SYS 21, 26  
RPLDIR 37  
RPLGROUP 72  
RPLTERM.BAT 22

## S

sandbox machine 91  
sandbox policy file 47  
Screen Saver 184  
ScriptIt Utility 170  
SEND.FILE.REQUEST frame 21  
Server 16  
Server back-up 62  
Server integration 55  
Server Operating Systems 7  
Server record 30  
server record format 31  
Server Side Changes 16  
Server-based applications 89  
Servers supported 7  
SETIVOLI.CMD 212  
setup.exe 148  
SETUP.ISS 213  
Single Server environments 55  
Software distribution 104  
Software Distribution Manager 62  
Software prerequisites 59  
sourcepath 142  
standard management 205  
StarOffice 102  
Start menu 203



- State files 39
- State Switch Daemon 41
- STATE values 40
- STATE.FIL 20
- Super-Thin Client 217
- System Policy Editor 193
- System policy editor 48
- System Policy Files 47, 179
  - location 48
- System policy templates 195

## T

- TCP/IP 132
- TCP/IP Subnet 135
- Technology choices 225
- Terminal Server 216
- Testing 76
- Tivoli 78
- Tivoli gateways 213
- Tivoli Management Agent 7, 16
- Tivoli Management Framework 206
- Tivoli software 205
- TIVOLI.BAT 212
- TMA
  - enabling 209
  - install prerequisites 208
  - installation 211
  - introduction 206
  - role 208
  - starting 213
  - steps 210
  - stopping 213
- TMA Installation 79
- TMR Server 208
- Total Cost of Ownership 222
- Traditional Client Licensing 11
- TYPE parameter 178

## U

- UNATTEND.TXT 136
- Unattend.txt 143
- unattend.txt 20
- Unattended installation 69
- unattended installation 68
- Uninstalling 72
- User Access 85
- User Datastore 129
- User environment 43

- User management 16, 53
- User Profile 44, 51
- User Profiles 179
- Users desktop 44
- USF file 142

## V

- Version File 20
- Video Adapters 158

## W

- W32DOSSB File 38
- W32DOSUB file 38
- WCF\_95R2 E C 72
- WCF\_98 E C 72
- WCF\_NT E C 72
- Web Browsers 16
- Well Behaved Applications 88
- wildcard characters 28
- Win32 Application Updates 87
- WIN32APP 57
- WIN32APP.INI 65
- Windows 16-bit 17
- Windows 95 adapters 227
- Windows 98 adapters 229
- Windows 9x
  - Diskspace 9
  - Hardware 9
  - Licensing 10
  - Memory 9
  - Processor 9
- Windows Application Datastore 128
- Windows Application Share or Local Drive 129
- Windows applications 16
- Windows desktop 44
- Windows device support 10
- Windows NT integration 57
- Windows NT Workstation
  - Adapters 227
  - Disk space 10
  - Hardware 9
  - Licensing 10
  - Memory 9
  - Processor 9
- Windows registry 45
- Windows Registry Updates File 129
- Windows Resource Kits 10
- winnt 141

## WorkSpace On-Demand

6

Application Management 5

Benefits 1

Components 2

Desktop Management 5

Desktops 6

Environments 2

Feature for Windows Clients 5

Java Virtual Machine 7

Manager 4

Network Logon Client 6

Operating systems supported 7

Overview 1

Server platforms 7

TMA 7

Workstation Management 5

Workstation machine ID 28

Workstation record 28

Workstation type 136

## Y

Year 2000 73

Year 2000 readiness 73

---

## ITSO redbook evaluation

WorkSpace On-Demand 2.0 Feature for Windows Clients  
SG24-5396-00

Your feedback is very important to help us maintain the quality of ITSO redbooks. **Please complete this questionnaire and return it using one of the following methods:**

- Use the online evaluation form found at <http://www.redbooks.ibm.com>
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to [redbook@us.ibm.com](mailto:redbook@us.ibm.com)

Which of the following best describes you?

☐ **Customer**   ☐ **Business Partner**   ☐ **Solution Developer**   ☐ **IBM employee**  
☐ **None of the above**

**Please rate your overall satisfaction** with this book using the scale:  
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction \_\_\_\_\_

**Please answer the following questions:**

Was this redbook published in time for your needs?      Yes\_\_\_\_ No\_\_\_\_

If no, please explain:

---

---

---

---

What other redbooks would you like to see published?

---

---

---

**Comments/Suggestions:      (THANK YOU FOR YOUR FEEDBACK!)**

---

---

---

---

