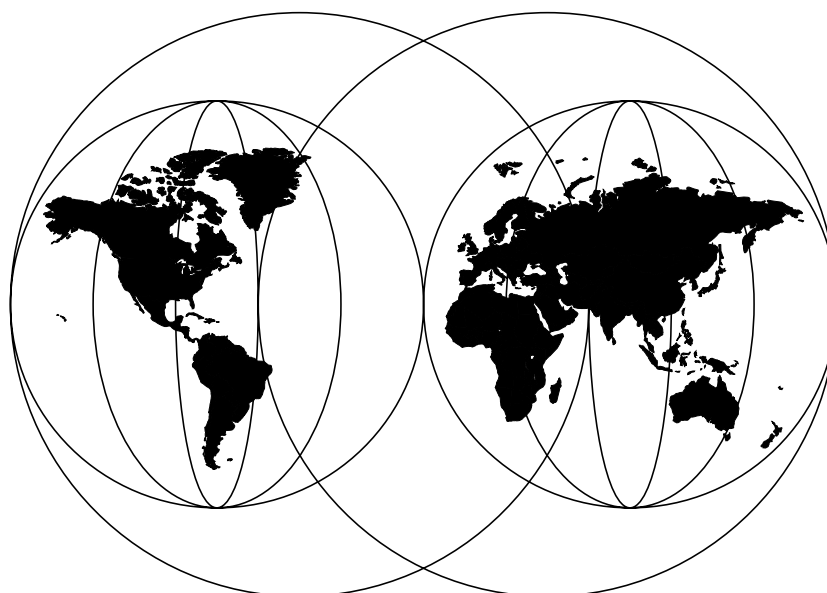


# **WorkSpace On-Demand 2.0 Handbook**

*Paul Craton, Angel Hernandez Bravo, Norm Mattson, Reinaldo de Medeiros  
Charlie Rouh, Neil Stokes*



**International Technical Support Organization**

<http://www.redbooks.ibm.com>





International Technical Support Organization

**WorkSpace On-Demand 2.0  
Handbook**

October 1998

**Take Note!**

Before using this information and the product it supports, be sure to read the general information in Appendix F, "Special Notices" on page 363.

**First Edition (October 1998)**

This edition applies to Release 2.0 of WorkSpace On-Demand and WorkSpace On-Demand Manager for use with OS/2 Warp Server Version 4.0.

Comments may be addressed to:

IBM Corporation, International Technical Support Organization  
Dept. JN9B Building 045 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 1998. 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> .....	xi
<b>Tables</b> .....	xv
<b>Preface</b> .....	xvii
The Team That Wrote This Redbook .....	xviii
Comments Welcome .....	xxi
<hr/>	
<b>Part 1. Understanding WorkSpace On-Demand</b> .....	23
<b>Chapter 1. Introduction to WorkSpace On-Demand</b> .....	25
1.1 Server-Managed Clients .....	25
1.1.1 Advantages of a Server-Managed Client Environment .....	25
1.1.2 Points to Consider .....	28
1.2 The Benefits of WorkSpace On-Demand .....	29
1.2.1 Effective Software and Data Management .....	30
1.2.2 Easier End-User Support .....	30
1.2.3 Enhanced Security .....	31
1.2.4 Broad Application Support .....	31
1.2.5 End User Mobility .....	32
1.2.6 Migration Path to Full Network Computing .....	32
1.2.7 Investment Protection in Current Hardware .....	33
1.3 WorkSpace On-Demand Components .....	33
1.3.1 WorkSpace On-Demand Client Operating System .....	34
1.3.2 WorkSpace On-Demand Manager .....	34
1.4 New Features in WorkSpace On-Demand 2.0 .....	35
1.4.1 DHCP PXE Boot Mechanism .....	36
1.4.2 Client Enhancements .....	36
1.4.3 Desktop Enhancements .....	36
1.4.4 Application Support .....	37
1.4.5 Hardware Support .....	38
<b>Chapter 2. The Remote Boot Process</b> .....	39
2.1 The WorkSpace On-Demand Client Operating System .....	39
2.1.1 What is an Operating System? .....	39
2.1.2 Real Mode vs. Protect Mode .....	41
2.1.3 Loading the WorkSpace On-Demand Client Operating System .....	42
2.1.4 Loading Over a Network - Redirection .....	43
2.1.5 File Mappings - The File Index Table .....	43
2.1.6 Phases of the Boot Process .....	44
2.2 The Boot PROM .....	45

2.3	The IEEE 802.2 RPL Boot Mechanism . . . . .	45
2.3.1	Supporting an IEEE 802.2 Remote Boot Environment . . . . .	45
2.3.2	The Boot PROM and IEEE 802.2 . . . . .	47
2.3.3	Loading a Client to Use NetBIOS/NetBEUI Only . . . . .	48
2.3.4	Loading a Client to Use NetBIOS Over TCP/IP . . . . .	53
2.3.5	Server Services . . . . .	55
2.3.6	Key Server Files Used by the IEEE 802.2 RPL Boot Mechanism . . . . .	56
2.4	The DHCP PXE Boot Mechanism . . . . .	66
2.4.1	Supporting a DHCP PXE Boot Environment . . . . .	66
2.4.2	The Boot PROM and DHCP PXE . . . . .	68
2.4.3	Loading a Client . . . . .	70
2.4.4	Key Client Files Used by DHCP PXE . . . . .	72
2.4.5	Server Services . . . . .	74
2.4.6	Key Server Files Used by DHCP PXE . . . . .	75
2.5	Server Directory Structures . . . . .	78
2.5.1	Generic Directories and Files . . . . .	80
2.5.2	Client-Specific Directories and Files . . . . .	81
2.5.3	User-Specific Directories and Files . . . . .	84
	<b>Chapter 3. Understanding File Index Tables . . . . .</b>	<b>85</b>
3.1	FIT Entry Syntax . . . . .	85
3.1.1	Using Wildcard Characters in FIT Files . . . . .	86
3.1.2	Using Variables in FIT Files . . . . .	87
3.2	The Machine FIT File . . . . .	87
3.2.1	Access Control Profiles . . . . .	90
3.2.2	Default Machine FIT Files . . . . .	90
3.2.3	Adding Your Own Entries to the Machine FIT File . . . . .	91
3.3	The User FIT File . . . . .	91
3.3.1	Access Control Profiles . . . . .	93
3.3.2	Default User FIT File . . . . .	93
3.4	The Application FIT File . . . . .	94
3.4.1	Access Control Profiles . . . . .	95
3.4.2	Pre vs. Post Application FITs . . . . .	95
3.5	Nesting and Combining FITs . . . . .	96
3.6	FITs in Memory . . . . .	98
	<hr/>	
	<b>Part 2. Installing Workspace On-Demand . . . . .</b>	<b>101</b>
	<b>Chapter 4. WorkSpace On-Demand Planning . . . . .</b>	<b>103</b>
4.1	Hardware Prerequisites . . . . .	103
4.1.1	WorkSpace On-Demand Server . . . . .	103
4.1.2	Client Workstation . . . . .	107
4.2	Software Prerequisites . . . . .	109

4.2.1	WorkSpace On-Demand Server . . . . .	109
4.2.2	Administration Client . . . . .	110
4.2.3	WorkSpace On-Demand Client Workstation . . . . .	111
4.2.4	Year 2000 Readiness . . . . .	111
4.2.5	Euro-Currency Support . . . . .	111
4.3	Supported Network Adapters . . . . .	113
4.4	Network Resource Requirements . . . . .	113
<b>Chapter 5. Installing WorkSpace On-Demand . . . . .</b>		<b>115</b>
5.1	Before You Install WorkSpace On-Demand . . . . .	115
5.2	OS/2 Warp Server Considerations . . . . .	116
5.2.1	Choosing a Version of OS/2 Warp Server . . . . .	116
5.2.2	Installing OS/2 Warp Server . . . . .	116
5.2.3	Applying FixPaks to OS/2 Warp Server . . . . .	119
5.3	Installing WorkSpace On-Demand on Your Server . . . . .	119
5.3.1	Attended Installation . . . . .	119
5.3.2	Unattended Installation . . . . .	128
5.4	Installing the WorkSpace On-Demand Administration Client . . . . .	131
5.4.1	Attended Installation . . . . .	131
5.4.2	Unattended Installation . . . . .	133
5.5	Uninstalling WorkSpace On-Demand . . . . .	133
<hr/>		
<b>Part 3. Managing WorkSpace On-Demand . . . . .</b>		<b>135</b>
<b>Chapter 6. Working with Client Workstations . . . . .</b>		<b>137</b>
6.1	Defining a Client Workstation . . . . .	137
6.1.1	Using the GUI . . . . .	137
6.1.2	Using the Command Line Interface . . . . .	147
6.1.3	The Client Definition Process . . . . .	150
6.2	Defining Client Workstations Automatically . . . . .	151
6.2.1	The Process . . . . .	152
6.2.2	Possible Extensions . . . . .	159
6.3	Deleting a Client . . . . .	161
6.3.1	Using the GUI . . . . .	161
6.3.2	Using the Command Line Interface . . . . .	162
6.4	Querying Client Information . . . . .	163
6.5	Printing . . . . .	165
6.5.1	Network Printing . . . . .	165
6.5.2	Local Printing . . . . .	165
<b>Chapter 7. Working with Network Applications . . . . .</b>		<b>167</b>
7.1	Application Structure Under Workspace on Demand . . . . .	167
7.1.1	Application Types . . . . .	168

7.1.2	Generating Directory Structures . . . . .	169
7.1.3	Copying Files to the Directory Structures . . . . .	172
7.1.4	Redirecting File I/O Requests . . . . .	172
7.1.5	Access Control Profiles . . . . .	173
7.2	Installing an Application on the Server . . . . .	174
7.2.1	Hardware Requirements . . . . .	174
7.2.2	Create a Temporary Environment . . . . .	175
7.2.3	Install the Application . . . . .	176
7.2.4	Identify Client- and User-Specific Files . . . . .	178
7.2.5	Copy Application Code to the Server . . . . .	179
7.2.6	Copy Client- and User-Specific Files to the Server . . . . .	179
7.2.7	Modify Operating System Configuration Files . . . . .	180
7.2.8	Redirect Required Files . . . . .	181
7.2.9	Test the Application Definition . . . . .	182
7.2.10	Clean Up the Installation Environment . . . . .	182
7.3	Defining the Application Using the GUI Interface . . . . .	182
7.3.1	Identify the Application . . . . .	184
7.3.2	Specify the Program Name . . . . .	184
7.3.3	Specify the Program Location . . . . .	185
7.3.4	Assign a Work Directory . . . . .	186
7.3.5	Specify the Program Mode . . . . .	187
7.3.6	Allocate Network Resources . . . . .	187
7.3.7	Define Required Environment Variables . . . . .	188
7.3.8	Create the Application Definition . . . . .	193
7.4	Defining the Application Using the Command Line Interface . . . . .	193
7.4.1	OS/2 Presentation Manager Application . . . . .	194
7.4.2	OS/2 Command Line Application . . . . .	194
7.4.3	WINOS2 Application . . . . .	194
7.4.4	DOS Application . . . . .	195
7.4.5	Using Response Files . . . . .	195
7.5	DOS and WIN-OS/2 Applications . . . . .	196
7.5.1	DOS and WIN-OS/2 Settings . . . . .	196
7.6	Java Applications . . . . .	197
7.6.1	Java Applets . . . . .	197
7.6.2	Java Applications . . . . .	199
7.7	Granting Access to Users . . . . .	199
7.7.1	Using the GUI . . . . .	200
7.7.2	Using the Command Line Interface . . . . .	201
7.7.3	User-Specific Application Parameters . . . . .	202
7.8	Customizing INI Files . . . . .	203
7.8.1	INI File Structure . . . . .	203
7.8.2	Comparing INI Files . . . . .	204
7.8.3	Modifying INI File Entries . . . . .	204



7.9 Application Roaming .....	206
<b>Part 4. Customizing WorkSpace On-Demand .....</b>	<b>209</b>
<b>Chapter 8. Modifying the User Interface .....</b>	<b>211</b>
8.1 PMLOGON .....	211
8.2 Modifying the Default Shell .....	212
8.2.1 Logon Options .....	213
8.2.2 User Exits During Boot and Logon .....	215
8.2.3 Replacing the Logon Bitmap .....	222
8.2.4 Replacing the Background Bitmap .....	223
8.3 Customizing Program Objects .....	223
8.3.1 Positioning Icons on the Desktop .....	226
8.3.2 Creating Folders .....	228
8.3.3 Modifying the Default Icons .....	235
8.4 Locking Up the Desktop .....	236
8.5 Logoff and Shutdown Options .....	238
8.6 Replacing the PMLOGON Shell .....	238
<b>Chapter 9. Supporting Network Adapters .....</b>	<b>241</b>
9.1 Enabling Adapters Supported by WorkSpace On-Demand 2.0 .....	241
9.2 Enabling Adapters Supported by OS/2 Warp Server .....	241
9.3 Enabling Unsupported Adapters .....	242
9.3.1 Obtain the DOS and OS/2 NDIS Drivers .....	243
9.3.2 Install the DOS Driver .....	243
9.3.3 Create a PROTOCOL.INI File for DOS .....	244
9.3.4 Install the OS/2 Driver .....	245
9.3.5 Create a PROTOCOL.INI File for OS/2 .....	246
9.3.6 Create a Boot Block Definition File .....	247
9.3.7 Edit the NDISDD.PRO File .....	248
9.3.8 Edit the RPL.MAP File .....	249
9.3.9 Test Your Definition .....	249
<b>Chapter 10. Supporting Additional Hardware .....</b>	<b>251</b>
10.1 Machine Classes .....	251
10.1.1 When Do You Need a Machine Class? .....	251
10.1.2 What is in a Machine Class? .....	252
10.1.3 What is Outside a Machine Class? .....	253
10.1.4 Machine Classes Shipped with WorkSpace On-Demand .....	254
10.1.5 Additional Machine Classes .....	255
10.1.6 Planning Your Machine Classes .....	256
10.2 The Machine Class Create Utility .....	257
10.3 Creating Your Own Machine Class .....	257

10.3.1	Hardware Requirements . . . . .	258
10.3.2	Feasibility Tests . . . . .	258
10.3.3	Useful Tools for Machine Classing . . . . .	259
10.4	Machine Classing Methodology . . . . .	261
10.4.1	Install OS/2 Warp 4 on the Reference Client. . . . .	263
10.4.2	Define the Reference Client to WorkSpace On-Demand . . . . .	263
10.4.3	Create a User ID. . . . .	263
10.4.4	Create Temporary Directories on the Server. . . . .	264
10.4.5	Capture the Basic System State . . . . .	264
10.4.6	Configure Hardware Support for OS/2 Warp 4 . . . . .	265
10.4.7	Capture the Enhanced System State . . . . .	266
10.4.8	Create New Subdirectories . . . . .	271
10.4.9	Create Configuration Template Files. . . . .	272
10.4.10	Copy Files to New Subdirectories . . . . .	282
10.4.11	Modify the MCLASS.RSP File. . . . .	283
10.5	Creating the New Machine Class . . . . .	285
10.5.1	Identify the New Machine Class . . . . .	286
10.5.2	Configure Hardware Support . . . . .	287
10.5.3	Configure the Peripheral Components . . . . .	288
10.5.4	Configure Video Support. . . . .	289
10.5.5	Configure Individual CONFIG.SYS Entries . . . . .	290
10.5.6	Create the New Machine Class. . . . .	291
10.5.7	Test the New Machine Class . . . . .	291
10.5.8	Points to Consider . . . . .	292
10.6	Common Problems. . . . .	292
10.6.1	Corrupted NETGUI.INI . . . . .	292
10.6.2	Unable to Create a New Remote IPL Requester . . . . .	293
10.6.3	No Entries in the Machine Class Dropdown List . . . . .	293
10.6.4	No Entries in the Video Monitor Drop-Down List . . . . .	293
10.6.5	Exception in Device Driver SINGLEQ\$ . . . . .	294
10.6.6	Black Screen Instead of a Client Desktop . . . . .	294
10.7	Querying Machine Class Information . . . . .	295
	<b>Chapter 11. Deploying WorkSpace On-Demand . . . . .</b>	<b>297</b>
11.1	Deployment Planning . . . . .	297
11.1.1	Points to Consider . . . . .	297
11.1.2	Analyze Deployment Techniques . . . . .	301
11.1.3	Testing . . . . .	305
11.2	Infrastructure Considerations . . . . .	306
11.2.1	Current Hardware Inventory . . . . .	306
11.2.2	Using Multiple Boot Mechanisms on a Single Server . . . . .	306
11.2.3	Number of Clients Per Server . . . . .	306
11.2.4	Network Topology . . . . .	307

11.2.5 Network Traffic Profiles . . . . .	307
11.2.6 Network Protocols . . . . .	308
11.2.7 Routers and Bridges . . . . .	308
11.3 Performance Planning for WorkSpace On-Demand . . . . .	311
11.3.1 Server Performance Considerations . . . . .	311
11.3.2 Client Performance Considerations . . . . .	314
11.4 Automation . . . . .	315
11.4.1 Unattended Installation of WorkSpace On-Demand . . . . .	316
11.4.2 Creating Client Workstations Automatically . . . . .	316
11.4.3 Creating Public Applications . . . . .	317
<hr/>	
<b>Part 5. Appendices . . . . .</b>	<b>319</b>
<b>Appendix A. Automated Client Definition Programs . . . . .</b>	<b>321</b>
A.1 Client Hardware Query Program – QUERYCFG.CMD . . . . .	321
A.2 Client Definition Program – ADDNEWCL.CMD . . . . .	322
<b>Appendix B. Application Definition Scripts and Response Files . . . . .</b>	<b>329</b>
B.1 Application Definition Script – ADDAPPL.CMD . . . . .	329
B.2 Application Definition Response File – ADDAPPL.RSP . . . . .	333
<b>Appendix C. DOS and WIN-OS2 Settings . . . . .</b>	<b>335</b>
<b>Appendix D. Video Handling . . . . .</b>	<b>339</b>
D.1 Video Modes . . . . .	339
D.2 Video Configuration Files . . . . .	341
D.3 Editing SVGADATA.PMI . . . . .	345
D.3.1 Deleting Entries . . . . .	345
D.3.2 Adding Entries . . . . .	347
<b>Appendix E. Tools and Utilities . . . . .</b>	<b>353</b>
E.1 FWATCH . . . . .	353
E.1.1 File Watch Utility (FWU.EXE) . . . . .	353
E.1.2 Installation requirements . . . . .	354
E.1.3 Syntax . . . . .	355
E.1.4 Deciphering FWATCH output . . . . .	355
E.2 Network Analyzers . . . . .	359
E.3 SMBTOOL . . . . .	359
E.4 SYSLOGPM . . . . .	360
E.5 RDRDebug . . . . .	361
E.6 WSOD-CV . . . . .	361

<b>Appendix F. Special Notices</b> .....	363
<b>Appendix G. Related Publications</b> .....	367
G.1 International Technical Support Organization Publications .....	367
G.2 Redbooks on CD-ROMs .....	367
G.3 Other Publications .....	368
<b>How to Get ITSO Redbooks</b> .....	369
How IBM Employees Can Get ITSO Redbooks .....	369
How Customers Can Get ITSO Redbooks .....	370
IBM Redbook Order Form .....	371
<b>Glossary</b> .....	373
<b>List of Abbreviations</b> .....	377
<b>Index</b> .....	379
<b>ITSO Redbook Evaluation</b> .....	385

## Figures

1. Server-Managed Clients . . . . .	26
2. WorkSpace On-Demand Components . . . . .	34
3. Sample FIT Entry - File Redirection . . . . .	44
4. IEEE 802.2 Boot - Phase One . . . . .	49
5. IEEE 802.2 Boot - Phase Two . . . . .	50
6. IEEE 802.2 Boot - Phase Three . . . . .	51
7. IEEE 802.2 Boot - Phase Four . . . . .	52
8. Enhanced FIT File with Mini-FSD Cache List . . . . .	54
9. IEEE 802.2 Boot - Mini-FSD Caching Mechanism . . . . .	54
10. Sample RPL.MAP File . . . . .	56
11. Boot Block Definition File . . . . .	64
12. Sample FIT Entry . . . . .	66
13. DHCP PXE Boot Mechanism . . . . .	69
14. DHCP PXE Boot - Caching Mechanism . . . . .	71
15. Client.INF File . . . . .	73
16. Common Descriptor File BPCOMMON.xxx . . . . .	74
17. PXEDHCP Server Configuration File DHCP.D.CFG . . . . .	76
18. PXEPROXY Server Configuration File DHCP.D.CFG . . . . .	77
19. BINL Server Configuration File BINL.D.CFG . . . . .	78
20. Remote IPL Directory Structure . . . . .	80
21. Sample FIT Entry - File Redirection . . . . .	85
22. Sample FIT Entry - Directory Redirection . . . . .	86
23. User FIT File - User-Based File Redirection . . . . .	92
24. User FIT File Entry - User FIT File Redirection . . . . .	93
25. Default User FIT File - BB20USDU.FIT . . . . .	93
26. Nesting FIT Files . . . . .	96
27. FITs in Memory . . . . .	98
28. Selecting Remote Boot Service for OS/2 . . . . .	118
29. Ready to Install Upgrades . . . . .	121
30. Select WorkSpace On-Demand Components (1) . . . . .	122
31. Select WorkSpace On-Demand Components (2) . . . . .	124
32. Select WorkSpace On-Demand Components (3) . . . . .	125
33. Installing the Administration Client . . . . .	132
34. Defining a Client in the Remote IPL Requesters Folder . . . . .	138
35. Defining the Client - Identity Page . . . . .	139
36. Defining the Client - System Page . . . . .	140
37. Defining the Client - Hardware Page . . . . .	141
38. Defining the Client - Printer Page . . . . .	143
39. Defining the Client - Adding a Network Printer . . . . .	144
40. Defining the Client - Protocols Page . . . . .	145

41. Defining the Client - IP Address Page . . . . .	146
42. NET RIPLMACH Command Syntax . . . . .	148
43. Response File to Create WorkSpace On-Demand Clients . . . . .	149
44. Automated Client Detection and Definition . . . . .	152
45. RPL.MAP File with Model Client Definitions . . . . .	155
46. Model Client CONFIG.SYS Changes . . . . .	155
47. Model Client FIT File Changes . . . . .	156
48. RPL.MAP File with Model Client Definitions . . . . .	157
49. Dynamic Detection of Network Adapter Name . . . . .	157
50. RPL.MAP File after Client Definition . . . . .	158
51. TME 10 NetfinityNetfinity System Information Tool Output . . . . .	160
52. TME 10 Netfinity System Information Tool Video Adapter Detection . . .	161
53. Deleting Multiple Clients - Sample REXX Script . . . . .	162
54. Deleting Multiple Clients - Input File . . . . .	162
55. Querying Client Information Using NET RIPLMACH . . . . .	163
56. Gathering Client Information - Sample REXX Script . . . . .	164
57. Application Directory Structures . . . . .	169
58. Redirecting Application File I/O Requests . . . . .	172
59. Creating a Machine Class - Test Hardware Environment . . . . .	174
60. Public Application Definitions Folder . . . . .	183
61. Application Definition - Create Settings Notebook . . . . .	184
62. Application Definition - Program Location Page . . . . .	185
63. Application Definition - Work Directory Page . . . . .	186
64. Application Definition - Network Resources . . . . .	187
65. Application Definition - Parameters . . . . .	188
66. Application Definition - Adding Configuration Parameters . . . . .	189
67. Application Definition - Adding Application-Specific Parameters . . . . .	191
68. Defining the Java Configuration Applet . . . . .	198
69. Granting Access to an Application . . . . .	200
70. Add Public Applications Pop-Up . . . . .	201
71. Adding a User-Specific Application Parameter Value . . . . .	202
72. Set User-Specific Value Pop-Up . . . . .	203
73. PMLOGON Shell - Default RUNWORKPLACE Statement . . . . .	211
74. PMLOGON Shell - Logon Panel . . . . .	212
75. PMLOGON Shell - Modified RUNWORKPLACE Statement . . . . .	216
76. PMLOGON Shell - Flow of Control for User Exits . . . . .	217
77. PMLOGON Shell - Custom Logon Panel . . . . .	219
78. PMLOGON Shell - Logon String Format . . . . .	219
79. Example of NCC_SETUP_POST variable . . . . .	224
80. Icon Positioning - Using the GUI . . . . .	227
81. Icon Positioning Example . . . . .	228
82. NCC_FOLDER Environment Variable . . . . .	232
83. Multiple Applications in the Same Folder . . . . .	232

84. NCC_FOLDER Environment Variable - Multilevel Folders (1).....	233
85. NCC_FOLDER Environment Variable - Multilevel Folders (2).....	233
86. Multilevel Folders.....	234
87. Environment Variables.....	234
88. Customizing Folders.....	235
89. Default Desktop Icons - INI.RC Entries.....	236
90. Modified SET RUNWORKPLACE Statement.....	238
91. Netscape Navigator as the User Shell.....	239
92. DOS PROTOCOL.INI for Madge Adapter.....	244
93. OS/2 PROTOCOL.INI for Madge Adapter.....	246
94. BB2USMDG.CNF.....	247
95. NDISDD.PRO file.....	248
96. RPL.MAP file - Server Record Entry.....	249
97. Creating a Machine Class - Test Hardware Environment.....	258
98. DSPINSTL.LOG.....	268
99. Batch File Copy Command.....	271
100. Comparing CONFIG.SYS Files.....	273
101. CONFIG.HW.....	273
102. Machine Class INF File.....	274
103. DELTA.INI File.....	277
104. INI.RCH File.....	278
105. FIT Extension CL5465.FIT.....	280
106. SYSTEM.INH.....	281
107. WIN.INH.....	281
108. Video Subdirectories and Files for the CL-GD546X Chip Set.....	283
109. Adding a New Video_Type to MCLASS.RSP.....	284
110. Editing the New Video_Type in MCLASS.RSP.....	284
111. Creating a New Machine Class.....	285
112. Machine Class Create Notebook - Identity Page.....	286
113. Machine Class Create Notebook - Hardware Page.....	287
114. Machine Class Create Notebook - Peripherals Page.....	288
115. Machine Class Create Notebook - Video Page.....	289
116. Machine Class Create Notebook - User CONFIG.SYS Page.....	290
117. Machine Class File Structure.....	291
118. NET RIPLMCLAS /OS:BB20.US.....	295
119. NET RPLMCLAS NEW300GL /OS:BB20.US.....	296
120. LAN Segments Connected by Bridges.....	309
121. LAN Segments Connected by a Backbone.....	310
122. Video Configuration Files.....	342
123. Video Mode Determination Flow.....	344
124. README.1ST File for CL-GD546X Video Chip Set.....	346
125. Editing SVGADATA.PMI - Removing Entries.....	347
126. Example of Missing Color Depth.....	348

127.Editing SVGADATA.PMI to Include Color Depth . . . . . 349  
128.Supported Video Modes . . . . . 350



## Tables

1. Predefined Machine Classes in WorkSpace On-Demand 2.0 . . . . .	38
2. Client Operating System - File Types . . . . .	39
3. RPL.MAP File - Workstation Record Field Descriptions . . . . .	57
4. RPL.MAP File - Server Record Field Descriptions . . . . .	59
5. Boot Block Definition File - Field Descriptions . . . . .	65
6. Client Operating System - File Types . . . . .	78
7. Client-Specific Configuration Files - Read-Only Access . . . . .	82
8. Client-Specific Files - Read-Write Access . . . . .	83
9. Wildcard Characters in FIT Entries . . . . .	86
10. Default Machine FIT Files . . . . .	90
11. Server Memory Requirements - RIPL Service . . . . .	104
12. Disk Space Requirements - RIPL Service . . . . .	105
13. Disk Space Requirements - WSOD Features . . . . .	105
14. Disk Space Requirements - WSOD Client Components . . . . .	106
15. Predefined Machine Classes in WorkSpace On-Demand 2.0 . . . . .	108
16. National Language Versions and FixPak Levels . . . . .	112
17. Network Resource Requirements . . . . .	113
18. File Types - Application Software . . . . .	167
19. Application/File Types - Access Control Profiles . . . . .	173
20. MKTMPENV Utility Directory Copies . . . . .	175
21. WSOD_LAUNCH_SESSION Values . . . . .	192
22. NET APPPARAM Command - INI File Parameters . . . . .	205
23. WPProgram Setup Strings . . . . .	225
24. WPFolder Setup Strings . . . . .	230
25. WPObject Setup Strings . . . . .	231
26. Desktop Lockup - PMLOGON Parameters . . . . .	237
27. Machine Class Permutations for a Given Hardware Combination . . . . .	252
28. Predefined Machine Classes in WorkSpace On-Demand 2.0 . . . . .	254
29. DOS and WIN-OS2 Settings . . . . .	335
30. Video Resolutions and Video RAM . . . . .	340
31. Possible DosOpen() API Return Codes . . . . .	356
32. Possible DosOpen() API Open Flags . . . . .	357
33. Possible DosOpen() API Open Mode Values . . . . .	357



## **Preface**

This redbook will help you install, configure and administer a WorkSpace On-Demand environment. It is the result of a residency conducted at the International Technical Support Organization, Austin Center, during the final development of WorkSpace On-Demand 2.0. The redbook will help you to:

- Understand the WorkSpace On-Demand product, and the situations in which it is most appropriate to deploy WorkSpace On-Demand.
- Understand the remote IPL concepts behind WorkSpace On-Demand, and their implementation under WorkSpace On-Demand.
- Plan and install WorkSpace On-Demand in your enterprise.
- Define client workstations on your WorkSpace On-Demand server, and boot these client workstations remotely over the network using the WorkSpace On-Demand client operating system.
- Install network applications in your WorkSpace On-Demand environment, and make them available to client workstations and end users over the network.
- Add support for additional network adapters and other types of hardware, to expand the hardware support provided by the WorkSpace On-Demand product.
- Understand the planning steps necessary to deploy WorkSpace On-Demand in a large-scale enterprise environment.
- Understand the implications of installing WorkSpace On-Demand in your enterprise, in terms of its impact on existing network traffic and infrastructure.

The residents working with WorkSpace On-Demand 2.0 have taken their experiences and knowledge gained during the residency and captured it in this redbook. By using this redbook, you can capitalize on their experiences as you work with WorkSpace On-Demand in your environment.

---

## 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.

**Paul Craton** is an Advisory IT Specialist with the IBM Systems Center at Hursley Park in the United Kingdom, and provides pre-sales and post-sales support for large customers throughout Europe. He has extensive practical experience with WorkSpace On-Demand, having performed a large number of "proof-of-concept" projects for customers in Europe. He has an honors degree in Electrical Engineering from Southampton University.

**Angel Hernandez Bravo** is a Systems Engineer in the IBM Network Computing Software Division in Spain. He provides technical support and consulting services in many areas including OS/2, AIX, Windows NT and other networking software, and has been involved in several major OS/2-related projects in the banking and insurance industries. He has written about OS/2 and OS/2 Warp Server for a number of newspapers and has translated several OS/2-related books into Spanish.

**Norm Mattson** is a consultant with Golden Code Development Corporation, based in Atlanta, Georgia. He was a member of the team that implemented WorkSpace On-Demand at NationsBank, and has eight years of experience with OS/2, LAN Server and networking technologies. His areas of expertise include systems integration and automation.

**Reinaldo de Medeiros** is an independent consultant in Brazil, with six years of experience in OS/2 and related software. He is currently a member of the team implementing WorkSpace On-Demand at Banco do Brasil. He has written two books about OS/2, along with articles for several newspapers and magazines in Rio de Janeiro.

**Charlie Rouh** is a Senior Marketing Support Representative in the Personal Solutions Systems Center located in Westlake, Texas. He is responsible for providing IBM Business Partners and Software Marketing Specialists with advanced technical support for IBM Warp Server and LAN Server products. He has worked with OS/2 since 1988 and has over 28 years of field and headquarters experience in both marketing and technical organizations.

**Neil Stokes** is a Systems Engineer at the International Technical Support Organization, Austin Center. He has written extensively for IBM and for several major newspapers, and teaches IBM classes worldwide on all areas of Network Computing and Networking Software. Before joining the ITSO in 1996, he worked as a technical consultant for IBM Australia.

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

Ron Aguirre  
International Technical Support Organization, Austin Center

Antonio Arias  
IBM U.S.

Marcus Brewer, Editor  
International Technical Support Organization, Austin Center

Mala Anand  
IBM Network Computing Systems Division, Austin

Martin Bense  
GAD Gesellschaft fuer automatische Datenverarbeitung eG

Axel Buecker  
IBM Germany

Marcus Brewer  
International Technical Support Organization, Austin Center

John Case  
IBM U.S.

Oscar Cepeda  
International Technical Support Organization, Austin Center

Martin Dippold  
IBM Global Services, Germany

Andy Erhenzeller  
IBM Network Computing Systems Division, Austin

Ivan Faisal  
IBM Indonesia

Brad Fraley  
IBM Network Computing Systems Division, Austin

Steve French  
IBM Network Computing Systems Division, Austin

Mike Foster  
International Technical Support Organization, Austin Center

Ann Gleason  
IBM Network Computing Systems Division, Austin

Charlies Gotwald  
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.

Indran Naick  
IBM South Africa

Tutsoma Ohya  
IBM Japan

Marcello Savio  
IBM Brazil

Ira Schneider  
IBM Network Computing Systems Division, Austin

Tim Sennitt  
IBM U.K.

James Schoech  
IBM Network Computing Systems Division, Austin

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

Kevin Tapperson  
IBM Network Computing Systems Division, Austin

Keiichi Togashi  
IBM Japan

Robert Rose  
IBM Network Computing Systems Division, Austin

Dan Wiggins  
IBM Network Computing Systems Division, Austin

Uwe Zimmermann  
International Technical Support Organization, Austin Center

---

## 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 385 to the fax number shown on the form.
- Use the electronic evaluation form found on the Redbooks Web sites:

For Internet users <http://www.redbooks.ibm.com>

For IBM Intranet users <http://w3.itso.ibm.com>

- Send us a note at the following address:

[redbook@us.ibm.com](mailto:redbook@us.ibm.com)





## **Part 1. Understanding WorkSpace On-Demand**

This part of the redbook explains the basic concepts of WorkSpace On-Demand, including the purpose and concept behind the product.

Chapter 1, "Introduction to WorkSpace On-Demand" on page 25, provides an overview of WorkSpace On-Demand and discusses the kinds of environments for which WorkSpace On-Demand is appropriate.

Chapter 2, "The Remote Boot Process" on page 39, describes the IEEE 802.2 RPL and DHCP PXE boot mechanisms and how WorkSpace On-Demand uses each of these boot mechanisms to remotely boot client workstations on a network.

Chapter 3, "Understanding File Index Tables" on page 85, describes the file index table used by WorkSpace On-Demand to route file I/O requests to the correct locations on the server.



---

## Chapter 1. Introduction to WorkSpace On-Demand

WorkSpace On-Demand is IBM's Intel-based (or compatible) 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.

This chapter discusses the way in which WorkSpace On-Demand implements a server-managed client environment and describes the WorkSpace On-Demand product itself.

---

### 1.1 Server-Managed Clients

A server-managed client is an intelligent device, such as a network computer or personal computer, that loads its operating system, applications and data over a network from one or more servers, rather than storing them locally. More than simply loading software and data from a server, however, the server-managed client concept allows centralized management of:

- Client workstations
- Applications
- Users and groups of users

thereby providing timely, secure, controlled access to applications and information for the end user.

#### 1.1.1 Advantages of a Server-Managed Client Environment

A server-managed client environment has a number of advantages over more traditional client/server environments. These are discussed in the following sections.

##### 1.1.1.1 More Effective Software Management

In a traditional client/server environment, the client's operating system, along with much of the application software, is stored locally on the client. This may cause a number of problems:

Since the end user has effectively unrestricted access to the client's local storage devices, it is quite easy for a user to accidentally or maliciously corrupt their operating system or application environment.

Since there may be large numbers of clients within an enterprise, and therefore many copies of the client operating system and applications, it becomes very difficult to apply updates and fixes and to ensure that all clients use the same level of software.

In a server-managed client environment, the client's operating system and applications are stored at the server(s). They can be therefore be protected from accidental or malicious corruption through judicious use of access control profiles.

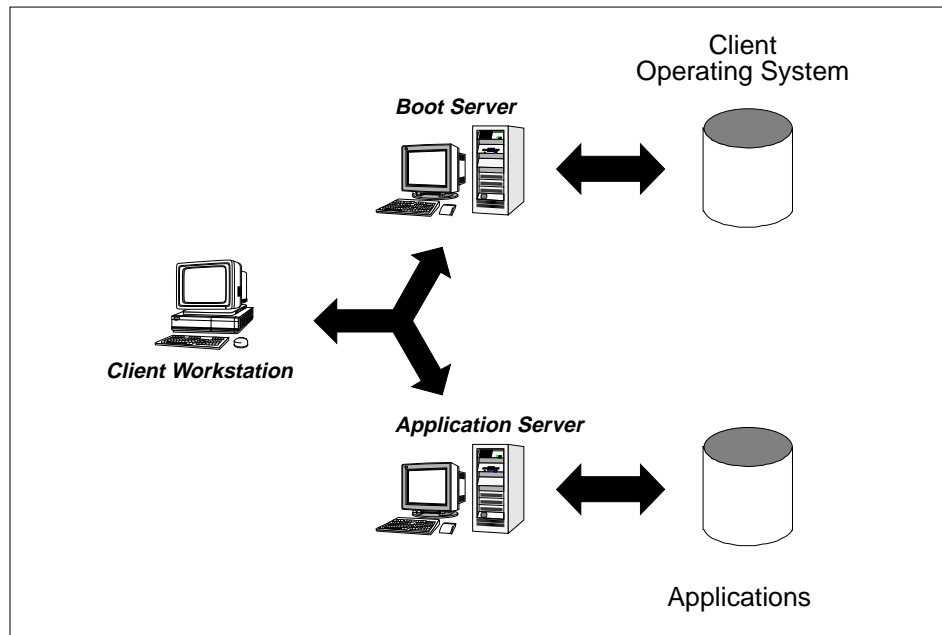


Figure 1. Server-Managed Clients

Storing the client's operating system and application software at the server also makes it easier to apply updates and fixes, since the number of copies of the software is greatly reduced. It is therefore much more feasible to apply updates across the entire network within a short period of time, thereby ensuring that all clients use the same level of operating system and application software.

#### 1.1.1.2 More Effective Application Management

In a traditional client/server environment, applications are installed on a client's local storage devices. This means that when end users require a new application, they must install it, or have someone install it, on their client

workstation. In large enterprises, this quickly becomes a costly and time-consuming exercise.

In a server-managed client environment, all applications are installed on a server, where they can be accessed by users on multiple clients. This not only simplifies the issue of application installation and maintenance but also makes it very simple for an end user to gain access to a new application, since an administrator must simply authorize the user for access to the application on the server. When the user next logs on, or refreshes the client environment, he/she will immediately have access to the application.

#### **1.1.1.3 More Effective Data Management**

In a traditional client/server environment, it is quite difficult to prevent users from storing critical data on the clients' local storage devices. If a local storage device fails or files are deleted by the user, the data can be lost. In a server-managed client environment, however, all data is stored at the server, where it can be properly managed, backed up and protected. A server-managed client environment can therefore provide a more robust data storage capability.

#### **1.1.1.4 Easier End User Support**

We have already mentioned that, in a traditional client/server environment, it is relatively easy for an end user to corrupt the client's operating system or application environment. When this happens, it is often quite difficult for end user support personnel to determine precisely what has been done and to rectify the problem. The problem determination and rectification process can be complicated and time-consuming both for the support person and for the end user.

A server-managed client environment can help to simplify the issue:

- Since the client operating system and applications reside at the server, the stored code can be easily protected from accidental or malicious corruption. It is therefore less likely that an end user will encounter a problem with the operating system or applications.
- If a problem does occur, it is unlikely that the stored copies of the operating system or applications on the server are corrupted. In most instances, the user can simply reboot the client and restore it to a working state.
- In the unlikely event that an operating system or application file is corrupt, the support personnel can work on the server to rectify the problem. Since the server is typically a standardized, controlled environment, it is often much easier to determine the cause of the problem and to rectify it.

A server-managed client environment can therefore reduce the complexity and consequent cost of problem determination and rectification.

#### **1.1.1.5 Enhanced Security**

In a traditional client/server environment, an end user can introduce new software or data files through the client's local storage devices such as diskette or CD-ROM drives. This means that threats such as viruses can easily be introduced into an enterprise's network. Measures can be taken to guard against such threats, but the process becomes quite costly.

Some server-managed client implementations allow an administrator to disable a client's local storage devices, such as diskette or CD-ROM drives, thereby preventing an end user from loading software or data onto the client or server. This helps to greatly reduce the danger of a virus being inadvertently or deliberately introduced into the network.

### **1.1.2 Points to Consider**

While there are many advantages of a server-managed client environment, there are several characteristics of the environment that may make it less suitable for certain organizations or uses. A number of these are discussed in the following sections.

#### **1.1.2.1 Network Availability is Critical**

In a server-managed client environment, a client's operating system, applications and data are all loaded over the network from a server. This naturally means that users' ability to carry out their work depends on the network and the server(s) being available. If the network goes down, or a server fails, this can seriously impact end-user productivity.

There are a number of measures that you can take to mitigate the risk of network or server failure. Many of these measures also apply to traditional client/server environments, and many organizations with existing client/server networks will already have taken steps to ensure the availability of their networks and servers.

#### **1.1.2.2 Savings Depend on Economies of Scale**

Many of the cost savings associated with a server-managed client environment accrue from economies of scale. For example, if an organization has 3500 clients and 3500 servers, it is far more economical to apply software updates to the servers, rather than to each client individually.

This means that a server-managed client environment is most effective where large numbers of clients use the same operating system and application

environment, which can be stored on a server and shared by multiple clients. In an organization where end users require different operating system and application environments, however, the economies of scale may not be so great, and a server-managed client environment may not be so attractive. You must weigh the relative merits of server-managed clients with regard to your own organizational environment.

Another aspect of this issue is the environment where a business location has only a single client workstation, and no server. In such circumstances, the cost of adding a server to the location, simply to support one client, may be prohibitive.

### **1.1.2.3 Mobile Users**

As mentioned above, a server-managed client environment relies on the availability of a server and a network connection. In situations where users are highly mobile, a network connection may not always be available. The user will need to work in a stand-alone mode, and the client must therefore store its own operating system, applications and data.

Even if a network connection is available, bandwidth and reliability considerations may make it impractical to load the operating system, applications and data over the network connection. Again, the client may need to store its own operating system, applications and data, and a traditional "fat" client model may be more appropriate.

---

## **1.2 The Benefits of WorkSpace On-Demand**

WorkSpace On-Demand implements the server-managed client environment using Intel-based client workstations and servers. As such, it provides the benefits of the 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

The following sections describe each of these benefits in more detail.

### 1.2.1 Effective Software and Data Management

WorkSpace On-Demand stores the client operating system on a WorkSpace On-Demand server and downloads it to the client at boot-time. Application software is also stored at the server (either the WorkSpace On-Demand server or another server on the network).

The server software allows operating system and application files to be selectively placed in read-only and read/write areas on the server(s). This ensures that the operating system and application software is protected from accidental or malicious corruption by an end user and that users can only access and modify those aspects of the operating system and applications that are permitted by the system administrator.

Storing operating system and application files on the server allows updates and fixes to be applied more easily since one set of updates on a server will automatically update many clients. The server also provides a more controlled environment, thereby making it easier and faster to apply updates and fixes to the clients' operating system and application environments.

By default, WorkSpace On-Demand requires an end user to store all data on a server rather than on the client's local storage devices. Storing data on a server means that the data can be more effectively managed and protected by system administrators, thereby reducing the likelihood that critical data might be lost.

### 1.2.2 Easier End-User Support

By protecting the operating system and applications from accidental or malicious corruption by an end user, WorkSpace On-Demand helps to reduce the incidence of software-related errors at the client. This reduces the cost associated with diagnosing and rectifying such errors.

An administrator can design and implement simple, standardized client desktop environments for WorkSpace On-Demand clients, which end users cannot change. This helps to reduce the cost of end user support since it reduces the number of options available to the end user and therefore reduces both the likelihood of errors and the subsequent complexity of problem determination.

The server environment allows users to gain access only to those applications for which they are authorized and trained. This helps to reduce the cost associated with users inadvertently "getting lost" and requiring help from support personnel.



### 1.2.3 Enhanced Security

When a WorkSpace On-Demand client starts up, it immediately displays a logon panel. A user is only able to log on to the network or to shut down the client. This prevents anyone without a valid user ID and password from gaining access to applications and data.

When a user logs on to a client, the client presents the user with a customized desktop environment containing only the applications for which the user is authorized. Users cannot accidentally gain access to applications and other resources they are not authorized to use.

By default, WorkSpace On-Demand does not allow an end user to access the client's local storage devices. This makes it impossible for an end user to introduce applications or data files onto the client, and therefore to the network, using diskettes or CD-ROMs.

As mentioned in Section 1.2.1, "Effective Software and Data Management" on page 30, WorkSpace On-Demand requires users to store their data on a server. This facilitates effective backup and other data security measures, making it less likely that critical data is lost.

The WorkSpace On-Demand client operating system can be tailored to allow additional security measures, such as customized shells and anti-virus measures. Such software can be implemented on your WorkSpace On-Demand clients in a similar manner to OS/2 Warp clients.

### 1.2.4 Broad Application Support

WorkSpace On-Demand provides a broad range of application support:

- *Java applications*

Java support is built directly into the WorkSpace On-Demand client operating system, thus providing a performance advantage.

- *OS/2 applications*

The WorkSpace On-Demand client operating system supports OS/2 applications natively.

- *DOS and Windows 3.x applications*

The WorkSpace On-Demand client operating system provides the same DOS and Windows 3.x support available in OS/2 Warp Version 4.

- *3270 and 5250 access*

The WorkSpace On-Demand client operating system allows host access to S/390 or AS/400 mainframes through native OS/2 applications such as the IBM Personal Communications family of products.

- *Intranet and Internet access*

WorkSpace On-Demand allows you to run browsers such as Netscape Navigator and other OS/2-based utilities to access intranet and Internet resources in the enterprise environment.

- X-Windows applications

You can implement a third-party X-Server application to use X-Windows applications.

- *32-bit Windows applications*

You can install a third-party multi-user Windows NT application server such as Citrix Winframe to serve your WorkSpace On-Demand clients.

WorkSpace On-Demand therefore provides a range of application support that allow you not only to run your existing platform-specific application software, but also to run emerging Java applets and applications natively on your WorkSpace On-Demand clients.

### 1.2.5 End User Mobility

WorkSpace On-Demand allows a system administrator to define a client desktop environment for each end user containing the applications for which that user is authorized, and only those applications. If a user moves from one client to another and logs on, he or she is presented with the same desktop environment. This is known as *application roaming*.

The application roaming function becomes particularly useful in environments where end users do not always work at a specific client workstation but may move between a number of clients at different times. For example, customer service representatives in a retail banking environment may use different client workstations at different times.

### 1.2.6 Migration Path to Full Network Computing

WorkSpace On-Demand provides a migration path from your current OS/2-based line-of-business applications to a "full" network computing implementation using technologies such as Java. WorkSpace On-Demand allows you to run your current applications while providing the necessary infrastructure to support emerging technologies.

You can use WorkSpace On-Demand as a migration platform to maintain support for your current application base while beginning to deploy Java-based network computing applications. At some point in the future, when your Java-based applications are in place and you are ready to phase out your existing, platform-specific applications, you can easily do so on the WorkSpace On-Demand platform.

### 1.2.7 Investment Protection in Current Hardware

WorkSpace On-Demand allows you to implement a server-managed client environment using the existing Intel-based client and server hardware that you use for a client/server environment. WorkSpace On-Demand *does not* require you to discard your current clients or servers and purchase expensive, new hardware.

You can implement WorkSpace On-Demand in your existing network simply by installing the software on a server and configuring your client's BIOS to boot the client from the network. In this way, WorkSpace On-Demand helps you to protect the investments you have already made in client/server technology while realizing the benefits of a server-managed client environment.

---

## 1.3 WorkSpace On-Demand Components

WorkSpace On-Demand consists of two basic components:

- The client component, known simply as *WorkSpace On-Demand*. In this redbook, however, we will often refer to the client component as the *WorkSpace On-Demand client operating system*, for reasons of clarity.
- The server component, known as *WorkSpace On-Demand Manager*, which runs on an OS/2 Warp Server system.

Figure 2 shows each of these components.

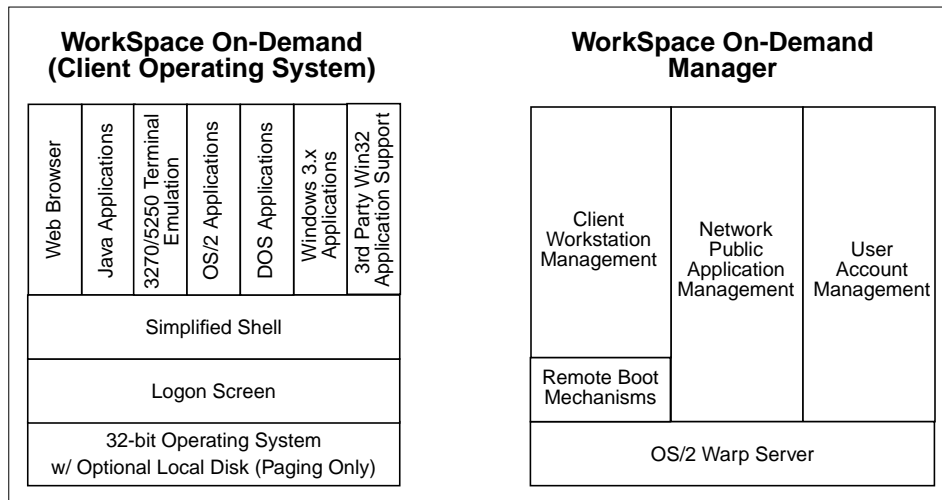


Figure 2. WorkSpace On-Demand Components

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

### 1.3.1 WorkSpace On-Demand Client Operating System

The WorkSpace On-Demand client operating system runs on the client workstation. This component is downloaded over the network from the WorkSpace On-Demand server.

You can use a wide variety of different types of machines as WorkSpace On-Demand clients. Some of these types are supported by the WorkSpace On-Demand product and are listed in Section 4.1.2.4, "Predefined Clients" on page 108. You can also configure WorkSpace On-Demand to support additional machines and hardware devices, as discussed in Chapter 10, "Supporting Additional Hardware" on page 251.

### 1.3.2 WorkSpace On-Demand Manager

The WorkSpace On-Demand Manager component is a set of server utilities that runs on OS/2 Warp Server and is used to install, configure and maintain the network client hardware and software. An OS/2 Warp Server system with WorkSpace On-Demand Manager installed is known as a *WorkSpace On-Demand server*. The WorkSpace On-Demand server therefore consists of the following:

- OS/2 Warp Server (prerequisite)

- File and Print Sharing (prerequisite)
- RIPL Support (prerequisite)
- WorkSpace On-Demand Manager software including:
  - Client configuration/administration utilities
  - RIPL service enhancements
  - Client application management

Note that all software for both the client and the server is physically installed on the server and resides on the server. Those components that run on the client are downloaded to the client at boot-time or, in the case of dynamic link libraries, when needed during the client's operation.

### **1.3.2.1 Heterogeneous Client Support**

WorkSpace On-Demand is designed to coexist with other end-user computing solutions that may exist in your environment. You can use a single OS/2 Warp Server system to provide support for a combination of traditional "fat" clients and DOS- or OS/2-based RIPL clients in addition to WorkSpace On-Demand clients. However, you should consider the performance and capacity requirements of your particular situation before determining whether or not to configure your WorkSpace On-Demand server(s) for heterogeneous client support.

### **1.3.2.2 Multi-Function Server Support**

You can combine the WorkSpace On-Demand Manager function on an OS/2 Warp Server system with other server functions such as the domain controller, backup domain controller or additional server, to create a multi-function server. For example, this kind of installation may be appropriate in a small branch office where cost considerations may make it impractical to install a dedicated WorkSpace On-Demand server. Note, however, that it is important to consider capacity and performance implications before implementing multi-function servers.

---

## **1.4 New Features in WorkSpace On-Demand 2.0**

WorkSpace On-Demand 2.0 introduces a number of new features and capabilities over those included in WorkSpace On-Demand 1.0. These features are briefly described in the following sections and are discussed in detail in the remainder of the redbook.

### 1.4.1 DHCP PXE Boot Mechanism

WorkSpace On-Demand includes support for the DHCP PXE boot mechanism as well as the earlier IEEE 802.2 RPL boot mechanism supported by WorkSpace On-Demand 1.0 and earlier OS/2-based RIPL implementations. With network adapter hardware that supports the DHCP PXE boot mechanism, you can use WorkSpace On-Demand 2.0 to boot your client workstations on a TCP/IP network. The DHCP PXE boot mechanism and the way it is implemented by WorkSpace On-Demand 2.0 is described in Section 2.4, "The DHCP PXE Boot Mechanism" on page 66.

### 1.4.2 Client Enhancements

WorkSpace On-Demand 2.0 enhances the client operating system with a number of updated components, including:

- Java Version 1.1.6
- Netscape Communicator Version 4.04
- MPTS Version 5.3
- TCP/IP Version 4.2.1

The WorkSpace On-Demand client operating system has also been upgraded to include a number of specific fixes for problems that existed in WorkSpace On-Demand 1.0. The client operating system in most national language versions of WorkSpace On-Demand 2.0 incorporates all fixes up to and including FixPak 6 for WorkSpace On-Demand 1.0, while certain national language versions (those requiring Euro-currency support) incorporate fixes up to and including FixPak 7.

### 1.4.3 Desktop Enhancements

WorkSpace On-Demand 2.0 allows you to customize the appearance and behavior of your client desktop by defining user exits that may be called during the boot process, logon and logoff. For example, you can display your own customized logon panel to handle user logon and perform any specialized processing in addition to the standard OS/2 Warp Server logon process.

The default client desktop now includes a Lockup icon, enabling a user to lock their desktop if they leave the client workstation unattended for any length of time. The lockup feature includes an automatic lockup function that will lock the desktop after a specified period of keyboard/mouse inactivity. When the user returns to the client workstation, entering the logon password unlocks the desktop.

In addition, you can specify your own unique logon bitmap to customize the WorkSpace On-Demand client's logon process for your own organization.

WorkSpace On-Demand 2.0 allows you to customize the program objects that appear on the desktop to a much greater extent than WorkSpace On-Demand 1.0. New capabilities include the ability to:

- Position icons on the desktop
- Create folders, including nesting folders within folders, and place program objects inside folders.
- Control the appearance and behavior of folders and program objects using setup strings

These enhancements are described in Chapter 8, "Modifying the User Interface" on page 211.

#### **1.4.4 Application Support**

WorkSpace On-Demand 2.0 includes a number of tools that facilitate the task of installing applications on the server for use by WorkSpace On-Demand clients. These tools (MKTMPENV and DIFFTREE) are used by a system administrator when installing the application and setting up its execution environment for use by clients. These tools and their uses are described in Chapter 7, "Working with Network Applications" on page 167.

WorkSpace On-Demand also introduces application-specific File Index Tables (FITs), which allow an application to redirect file access requests for application-specific files to the appropriate areas on a server. Application-specific FITs allow a user to move from one client workstation to another and run the same applications, without the need for the client to be pre-configured for each application.

While FITs were available in WorkSpace On-Demand 1.0, the application-specific FITs introduced in WorkSpace On-Demand 2.0 are only loaded into memory when an application is loaded, thereby optimizing memory usage by the FIT. FITs are described in detail in Chapter 3, "Understanding File Index Tables" on page 85, and application-specific FITs are discussed in Section 3.4, "The Application FIT File" on page 94.

### 1.4.5 Hardware Support

WorkSpace On-Demand 2.0 provides support for additional types of client hardware by providing additional predefined machine classes. Table 1 shows the list of predefined machine classes in WorkSpace On-Demand 2.0 and the hardware they support.

*Table 1. Predefined Machine Classes in WorkSpace On-Demand 2.0*

Machine Class	Comments
ISAVGA	Any PC with an ISA bus and a VGA display adapter that is supported by OS/2 Warp 4.
Generic GRADD	Generic ISA/PCI systems using VESA 1.2-compliant display adapters supported by GRADD drivers.
IBM300GL	IBM Personal Computer 300GL model 6282-38U
IBM300G2	IBM Personal Computer 300GL model 6272-880
IBM350	BM Personal Computer 350 model 6586-57H
IBM730	IBM Personal Computer 730 model 6877-KAZ
IBM750	IBM Personal Computer 750 model 6887-86B
MCAVGA	IBM PCs (486 +) with Micro Channel Bus and standard VGA with SCSI/ESDI support.
MCAXGA	BM PCs (486 +) with Micro Channel Bus and XGA video with SCSI/ESDI support.

The most notable additions to the list of supported hardware types are Micro Channel bus machines and XGA video support.

Note that the model numbers listed in Table 1 are significant. For example, the IBM PC 300GL has several different models, which may contain different display adapters. The IBM300GL machine class supports an IBM PC 300 GL with a specific display adapter.

WorkSpace On-Demand 2.0 allows you to create your own additional machine classes. WorkSpace On-Demand 2.0 includes a new Machine Class Create tool to facilitate this process. This tool is described, along with the concept of machine classes in general, in Chapter 10, "Supporting Additional Hardware" on page 251.



## Chapter 2. The Remote Boot Process

This chapter discusses the mechanisms by which a client workstation can load its operating system from a remote storage location on a network. We discuss the two different mechanisms (IEEE 802.2 RPL and DHCP PXE) supported under Workspace On-Demand 2.0.

### 2.1 The Workspace On-Demand Client Operating System

Before we discuss the way that Workspace On-Demand uses the two boot mechanisms, we need to understand the Workspace On-Demand client operating system itself and the way in which it loads itself into a client. This section provides a high-level explanation of the loading process.

#### 2.1.1 What is an Operating System?

An operating system can be considered as simply a group of programs and data files that must be loaded from storage media into memory. It is this loading process with which we are primarily concerned when discussing the remote boot process, since remote boot is nothing more than the process of loading these files over a network, from storage media located on a boot server, into memory on a client workstation.

##### 2.1.1.1 Types of Files

The files that comprise an operating system are typically of several types, as shown in Table 2.

Table 2. Client Operating System - File Types

File Type	Read-Only	Read/Write
Generic (All Clients)	95% of code	Not Applicable
Client-Specific	CONFIG.SYS	LANTRAN.LOG

Table 2 shows that there are two basic distinctions between the types of files that comprise an operating system:

- Generic operating system code and data that is used by all client workstations that run this operating system.
- Client-specific code and data that is unique to a particular client workstation or type of client workstation. For example, individual clients may have unique desktop layouts, and different types of client hardware may use different video configuration files.

Within the category of client-specific files, we must make a further distinction:

- Files that are accessed in read-only mode
- Files that are accessed in read/write mode

While this distinction may not appear relevant to a normal "fat" client workstation (for example, most users can access and modify their operating system configuration files such as CONFIG.SYS), it becomes significant later when we discuss a remote boot environment, where many clients may be using the same copy of the operating system code on a server. In such environments, you may wish to restrict users from modifying certain aspects of their operating system configuration by placing some configuration files in a read-only area on the server, while placing other configuration files in a read/write area where they can be modified by users.

In a normal "fat" client workstation, the operating system and its configuration files are only accessed by one user at a time. In a remote boot environment, however, many client workstations, and therefore many users, may be accessing the same copy of the operating system. You must therefore ensure that you keep separate copies of the client-specific, read/write files to ensure that the different clients and users do not interfere with one another.

#### **2.1.1.2 Loading an Operating System**

When you boot a PC, the machine's BIOS completes its Power-On Self Test (POST) sequence, then checks its bootable media. A *bootstrap* routine is loaded by the BIOS and searches the bootable media for a file known as a *loader*.

The term "bootstrap" originated in the days when computers used punched tape as a storage medium. The piece of code required to initialize these machines was often no longer than a belt or strap, and was hence called a "bootstrap."

When the bootstrap routine finds the loader, it loads it into memory and passes control to it. The loader is then responsible for loading the core of the operating system, known as the *kernel*, and any other files required for the kernel to access storage media directly.

After these files are loaded into memory, the loader passes control to the kernel. From this point on, the kernel is responsible for loading any remaining files and for preparing the operating system to run applications.

### 2.1.2 Real Mode vs. Protect Mode

The Intel 80386 processors, and their descendants, such as the 80486 and Pentium family), support several modes of operation. The two key modes are real mode and protect mode.

Real mode was originally designed for the older x86 family of processors (such as the 80286) and is based on a 16-bit architecture. This mode enables direct access to BIOS interrupts (that is, the machine's hardware). IBM PC DOS is an example of an operating system that uses real mode.

Protect mode was introduced in the 80386 family of processors to support the more advanced features of the processor (mainly related to memory management and "virtual machine" support). Protect mode uses the 32-bit architecture of the processor and unlike real mode, does not provide direct access to the BIOS interrupts. OS/2, Windows NT and Linux are all examples of protect mode operating systems that run on the 80386 processor family.

When a PC containing an 80386-family processor is turned on, the processor initially operates in real mode. The bootstrap routine searches for, locates and begins to load an operating system, then passes control to the operating system's loader which begins to boot the operating system. If the operating system supports protect mode, it will issue a command to the processor to switch to protect mode at some point during the boot sequence.

To support both modes of operation without the need for manual operations, a machine must start in real mode and allow the operating system to switch the processor to protect mode if the operating system supports protect mode operation. In this way, the latest PC hardware is able to support older, real mode operating systems such as DOS, while also supporting more modern, protect mode operating systems such as OS/2 and Windows NT.

A protect mode operating system must therefore start in real mode and while in real mode, must load and initialize the key operating system files for the protect mode environment. The operating system must then switch to protect mode and pass control to its protect mode component.

Once the protect mode environment has been initialized, it can only interface with the system hardware using device drivers that are also running in protect mode. A protect mode device driver and file system driver are also needed by the protect mode environment to access any form of storage media while in protect mode. The storage media holds all operating system files, including the protect mode drivers.

### 2.1.3 Loading the WorkSpace On-Demand Client Operating System

In most operating systems, the initial real mode functions are provided by a component known as a *loader*. The WorkSpace On-Demand client operating system uses a loader named OS2LDR, while Windows NT uses a loader named NTLDR. In a protect mode operating system, the function of the loader is to load the protect mode environment into the client's memory and then pass control to the protect mode environment. In order to access storage media and load the protect mode environment, OS2LDR uses a real mode file system driver known as the *micro file system driver (micro-FSD)*.

The protect mode environment is managed by an operating system *kernel*. In the WorkSpace On-Demand client operating system, this kernel is named OS2KRNL. The kernel runs in protect mode and can only operate and communicate when the processor is running in protect mode.

When the protect mode environment is first initialized and control is passed to the protect mode kernel, not all of the necessary device drivers and file system drivers may be present in memory to allow the protect mode kernel to access the machine's storage media. The kernel must therefore have a real mode interface to access files from the storage media until it can load its own protect mode file system and device drivers. In OS/2 and the WorkSpace On-Demand client operating system, this function is provided by a special file system driver called a *mini file system driver (mini-FSD)*, which is loaded into memory by OS2LDR before it passes control to the protect mode kernel. The mini-FSD handles requests for files in protect mode and can switch to real mode to access files from the bootable storage media until such time as the protect mode device drivers and file system driver are loaded and the operating system is able to access storage media directly from protect mode.

When the operating system is loading from a locally attached drive, the operation of the file system drivers is quite simple:

1. The micro-FSD supports OS2LDR as it loads the protect mode kernel and the mini-FSD from storage media.
2. The mini-FSD supports OS2KRNL as it loads its protect mode device drivers and file system driver from storage media.
3. The protect mode file system driver/redirector supports OS2KRNL as it loads any remaining operating system files from storage media.

The protect mode file system driver remains active throughout the operating system's execution, handling I/O requests from the operating system and applications.

### 2.1.4 Loading Over a Network - Redirection

In a network environment, the operating system must be able to access files on a server rather than a local storage device, in real and protect modes. For an OS/2-based operating system such as Workspace On-Demand to handle this requirement, the micro-FSD and mini-FSD must act as network *redirectors*, working through a network interface to access files on the server.

As with loading any protect mode OS locally, the micro-FSD and mini-FSD are only needed until the protect mode environment is able to access files directly. When loading Workspace On-Demand over a network, this requires a protect mode network transport layer, a protect mode file system driver, and a protect mode redirector that can handle file requests from OS2KRNL and redirect them to the file server on the network. Under WorkSpace On-Demand, this function is provided by the LAN Requester redirector (NETWKSTA.200).

When the LAN Requester redirector is loaded and active, OS2KRNL loads the remainder of the operating system environment using the protect mode redirector to access files from the server's storage media over the network. The operating system is then active and ready to begin running applications.

### 2.1.5 File Mappings - The File Index Table

In a normal "fat" client environment, where all operating system files are loaded from local storage media, it is quite simple for the operating system to locate the files it requires. The location of critical files is typically hard-coded into the loader or kernel, and the operating system has its own mechanisms (such as `PATH` and `LIBPATH` environment variables) that enable it to locate other files.

In a remote boot environment, however, the task of locating required files is somewhat more involved. All files, even those for which the location is hard-coded into the loader or kernel, are located on the server's storage media and must be accessed through a redirector.

To further complicate the task, the storage media on which the operating system resides may be shared by multiple clients. Client-specific files may therefore be located in different areas on the server, and the location mechanism must take this into account.

The WorkSpace On-Demand client operating system uses a *file index table (FIT)* to locate the files it requires to boot the operating system and to redirect the file I/O requests to the correct locations on the server. A FIT consists of a list of matched name pairs, an example of which is shown in Figure 3.

```
Z:\NETSCAPE\NETSCAPE.INI \\MYSERVER\NETSCAPE\NETSCAPE.INI
```

Figure 3. Sample FIT Entry - File Redirection

The item on the left-hand side is known as the *prototype filename* and corresponds to the logical drive, path and file name by which the client workstation refers to a file. The item on the right-hand side is known as the *substitution text* and is typically a Universal Naming Convention (UNC) name that identifies a server, network share name, path, and file name on which the file actually resides.

When the client operating system requests access to a file, the active file system driver (micro-FSD, mini-FSD or protect mode redirector) searches the FIT for a prototype file name that matches the requested file. It then substitutes the substitution text before passing the request on to the server.

This file mapping mechanism allows a client to access both shared files and client-specific files, as well as read-only and read/write files. If each client has its own FIT, requests for client-specific files can be mapped to client-specific storage areas on the server with the appropriate level of access, while shared operating system files can be mapped to common storage areas on the server.

See Chapter 3, "Understanding File Index Tables" on page 85, for a detailed description of FITs, their structure and uses.

### 2.1.6 Phases of the Boot Process

We can therefore see that there are four phases in loading the WorkSpace On-Demand client operating system over the network:

1. The bootstrap routine loads OS2LDR and the micro-FSD into the client workstation's memory and passes control to OS2LDR.
2. OS2LDR loads OS2KRNL and the mini-FSD into memory from the server's storage media using the micro-FSD to handle the I/O requests and passes control to OS2KRNL.
3. OS2KRNL loads its protect mode device drivers and file system drivers into memory from the server's storage media using the mini-FSD to handle the I/O requests. OS2KRNL now initializes its protect mode file system drivers and redirector.
4. OS2KRNL loads the remainder of the operating system environment using its own protect mode redirector. The mini-FSD is no longer required and is released from memory.

These phases become significant when we begin to discuss the boot mechanisms supported by WorkSpace On-Demand since each boot mechanism carries out these phases, particularly phases 1 and 2 above, in a different way.

---

## 2.2 The Boot PROM

For a client to request its operating system from a server on the network, it requires a special piece of code that can initialize the network adapter when the machine is turned on and can request and receive the bootable code from a server. This code is normally made available on a special Boot PROM that is installed on most network adapters shipped today. For some older adapters, however, the Boot PROM may be an optional component that must be purchased separately and installed on the adapter. In addition, certain adapters allow you to create a diskette that simulates the Boot PROM code.

Several standards for the Boot PROM code have evolved in recent years. The primary difference between the standards is the way they communicate with other devices on the network to receive their boot blocks.

WorkSpace On-Demand 1.0 supports a standard based on the IEEE 802.2 RPL model developed by the Institute of Electrical and Electronics Engineers (IEEE). As the name suggests, this standard used the IEEE 802.2 protocol. More recently, another standard has evolved based on TCP/IP protocols. WorkSpace On-Demand 2.0 supports this IP-based standard in addition to the IEEE 802.2 standard. Section 2.3, "The IEEE 802.2 RPL Boot Mechanism" on page 45, and Section 2.4, "The DHCP PXE Boot Mechanism" on page 66, describe the way in which each boot mechanism loads the WorkSpace On-Demand operating system environment.

---

## 2.3 The IEEE 802.2 RPL Boot Mechanism

This section discusses the way in which the IEEE 802.2 RPL mechanism loads the WorkSpace On-Demand client operating system. You may contrast this with the PXE/DHCP boot mechanism, which is described in Section 2.4, "The DHCP PXE Boot Mechanism" on page 66.

### 2.3.1 Supporting an IEEE 802.2 Remote Boot Environment

In order for a client workstation to load its operating system from a server using IEEE 802.2 RPL, both the client and the server must satisfy certain requirements, as described in the following sections.

### 2.3.1.1 Client Workstation

The client must have the ability to determine, during Power On Self Test (POST), that it must load its operating system code from a network rather than a local device. On a PC, this ability is typically provided by the system BIOS and is configured by selecting the network as the startup device.

Having been configured to start from the network, the client must then send a request over the network to a server, asking for the code it requires. This support is typically provided by a special Boot PROM located on the client's network adapter. This Boot PROM is discussed in more detail in Section 2.2, "The Boot PROM" on page 45.

The client must also be able to receive and interpret the server's response in order to start to load its operating system. Again, code in the Boot PROM interprets responses from the server and passes them on to the client machine itself. These responses are built into a piece of code known as a *boot block*, which is sent to the client by the server, and which contains the necessary program code and data to boot the client.

### 2.3.1.2 Boot Server

The server that supplies the code to the client workstation must be able to receive the request from the client over the network and recognize that it should respond. This ability is provided by services on the server that listen for requests from clients and are able to respond. WorkSpace On-Demand 2.0 uses the REMOTEBOOT and DHCP services to provide these functions for the two different network boot mechanisms it supports.

When the server has determined that it needs to respond to a client's request, it must then build the boot block and send it to the client. Configuration files on the server enable it to determine how to respond to the client's request and how to assemble the boot block information. These configuration files depend on the boot mechanism being used. Section 2.3, "The IEEE 802.2 RPL Boot Mechanism" on page 45, and Section 2.4, "The DHCP PXE Boot Mechanism" on page 66, discuss the two different boot mechanisms in more detail, including the configuration files used by each mechanism.

After a client has received and loaded the boot block from the server, it may need to reconnect back to the server in order to continue loading its operating system. In such cases, the server must make the relevant read-only and read/write areas available to the client. Depending on the boot mechanism being used, this ability is provided by the LAN Server file-sharing capability of OS/2 Warp Server or through the TFTP server component of OS/2 Warp Server's TCP/IP services.



### 2.3.2 The Boot PROM and IEEE 802.2

When a client workstation is configured to boot from the network, the BIOS passes control to the Boot PROM when the machine is turned on, and the Power On Self Test (POST) sequence completes. The code within the Boot PROM initializes the network adapter and begins sending a series of broadcast frames, known as FIND frames, over the network, in search of a server that will respond. Each FIND frame contains the client's network address.

When a server receives a FIND frame and determines that it should respond to the client, it sends a FOUND frame which provides the client's Boot PROM with the server's network address. Since the FIND frame originally sent by the client included the client's network address, the FOUND frame can be sent directly to the client.

A network session is now established between the client and the server at the IEEE 802.2 LLC level. The client's Boot PROM now sends a SEND.FILE.REQUEST frame to the server and goes into receive mode waiting for data frames.

The server builds a *boot block* that is sent to the client's Boot PROM in a series of FILE.DATA.RESPONSE data frames. The exact number of frames depends on the size of the block, which in turn depends upon a number of factors such as the type of network adapter in the client workstation.

The Boot PROM loads the boot block into the client's system memory. The boot block contains a bootstrap routine, the memory address of which is the last piece of information contained in the boot block. When the Boot PROM has completed receiving and loading the boot block, it passes control to the bootstrap routine.

After the Boot PROM has received the boot block, it takes over the diskette drive interrupt (INT 13h) and, from the client hardware's viewpoint, appears to be a diskette drive containing a write-protected, bootable diskette. The contents of this diskette are, in fact, the data contained in the boot block, including the bootstrap routine. When the Boot PROM passes control to the bootstrap routine, it effectively boots the client from the "virtual" diskette in memory. The bootstrap routine loads the initial loader, which is part of the boot block, and passes control to it.

Since the client regards the Boot PROM as a diskette drive, it requires all of the low-level data normally contained on a diskette, including system sectors, file access table and so on.

The "virtual" diskette image contained within the boot block may contain a simple operating system such as DOS, or may simply provide a real mode environment that loads a more complex, protect mode operating system such as WorkSpace On-Demand. When booting DOS in a RIPL environment, the basic operating system can be contained entirely within the boot block, so the loading process is quite simple. With a protect mode operating system such as WorkSpace On-Demand, however, the boot process has a number of steps and is therefore more complicated.

### 2.3.3 Loading a Client to Use NetBIOS/NetBEUI Only

There are differences in the boot process when booting a client that will use native NetBIOS protocols only compared to a client that will use NetBIOS over TCP/IP. This section describes how a client loads the WorkSpace On-Demand client operating system using the IEEE 802.2 RPL boot mechanism in a NetBIOS-only environment.

We have already discussed the way in which the Boot PROM receives the boot block from the server, loads it into memory and passes control to the bootstrap routine. The "virtual" diskette image created by the Boot PROM from the contents of the boot block contains the following items:

- An initial loader which, for the IEEE 802.2 RPL boot mechanism, is named RPLBOOT.SYS. This is a real mode program that is used to initialize the system and begin loading the other files.
- A real mode NetBIOS interface that provides access to the network.
- A file index table (FIT), which contains redirection statements enabling the client to locate files on the server. See Chapter 3, "Understanding File Index Tables" on page 85, for more information on FITs.
- A real mode file system driver/SMB redirector. This is the micro-FSD described in Section 2.1.3, "Loading the WorkSpace On-Demand Client Operating System" on page 42. It provides a mechanism by which real mode programs can use the file index table to access files on the server through the real mode NetBIOS interface. For the IEEE 802.2 RPL boot mechanism, this driver is named UFSD.SYS.
- A protect mode file system driver/SMB redirector. This is the mini-FSD described in section mentioned in Section 2.1.3, "Loading the WorkSpace On-Demand Client Operating System" on page 42. It provides a mechanism by which protect mode programs can access files on the server through the real mode NetBIOS interface (which makes it rather special). In RPL boot environments, this file is named MFSD30.SYS. This mechanism is very slow because only one file at a time can be passed from the real mode interface to programs running in protect mode.

- A real mode loader named OS2LDR, which loads the protect mode operating system environment.

The bootstrap routine loads RPLBOOT.SYS into an address specified in the last data frame of the boot block and passes control to RPLBOOT.SYS, which then loads a number of other files from the "virtual" diskette in the order specified by the boot block:

1. For the system to boot correctly, access to the network is required before control is passed to OS2LDR. For this reason, the real mode NetBIOS interface and the file index table are loaded next.
2. RPLBOOT.SYS then loads the real mode redirector to provide an interface for OS2LDR to start to download the files necessary to load OS2KRNL.
3. Finally, RPLBOOT.SYS loads and calls OS2LDR. When OS2LDR is successfully initialized, RPLBOOT.SYS is terminated, and control is passed to OS2LDR itself.

Figure 4 shows the flow of control at this point in the boot process.

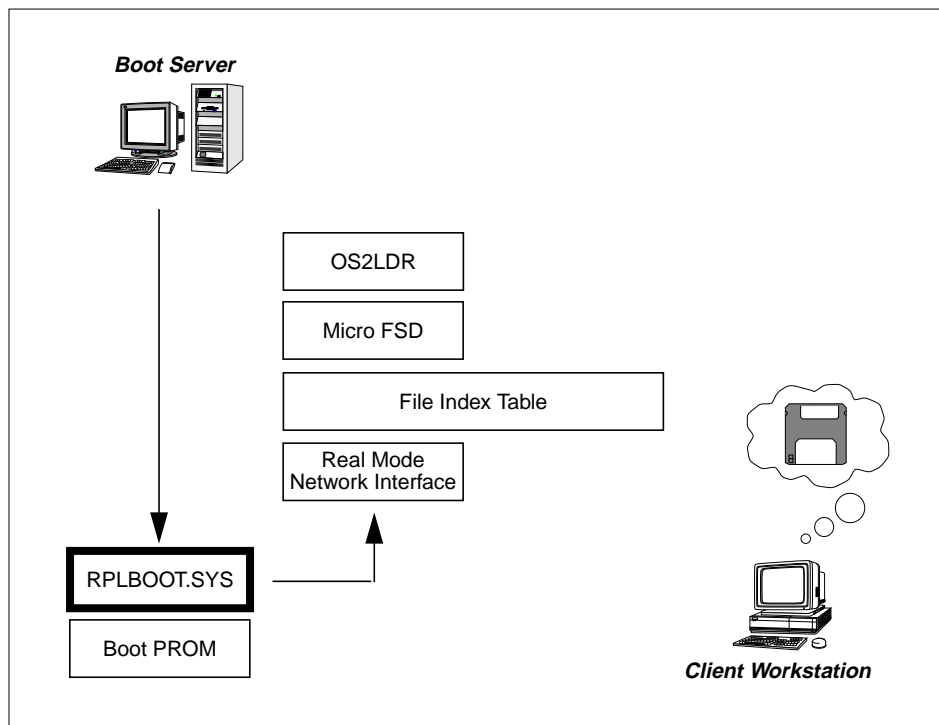


Figure 4. IEEE 802.2 Boot - Phase One

When OS2LDR receives control, it prepares the protect mode environment by performing the following tasks:

1. OS2LDR initializes the real mode micro-FSD (UFSD.SYS) and interfaces with it in real mode to retrieve the OS2LDR.MSG and OS2KRNL files from the server.
2. OS2LDR terminates the micro-FSD
3. OS2LDR initializes OS2KRNL.
4. OS2LDR switches to protect mode and passes control to OS2KRNL.
5. OS2KRNL initializes the mini-FSD.

OS2LDR has now completed its function and takes no further part in the boot process. Figure 5 shows the flow of control at this point in the boot process.

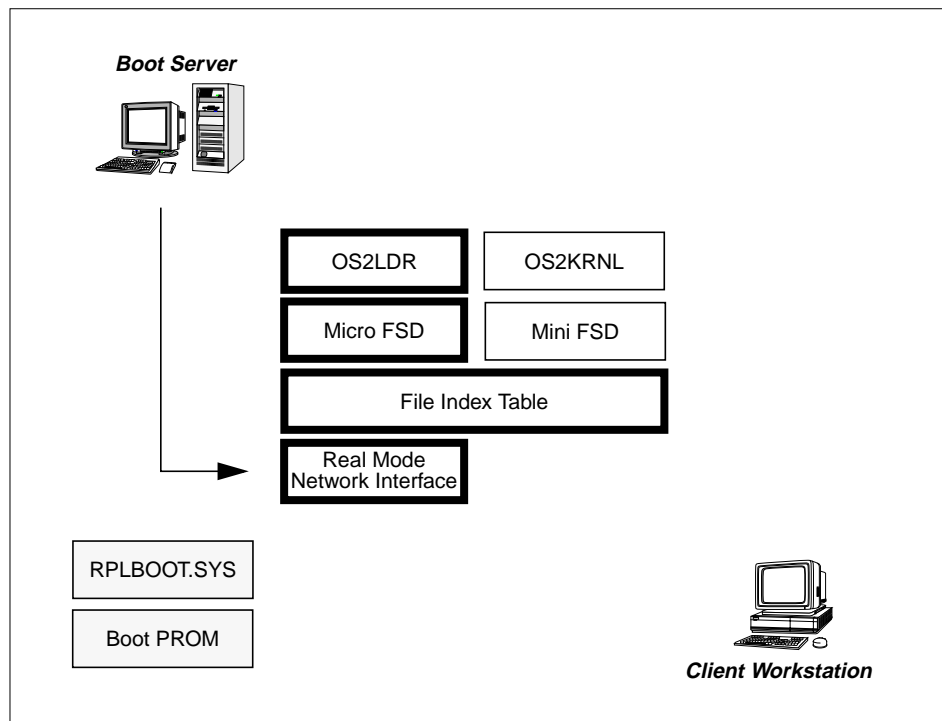


Figure 5. IEEE 802.2 Boot - Phase Two

Note that both the micro-FSD and the mini-FSD use the file index table and the real mode NetBIOS interface to locate and access files located on the remote boot server.

When OS2KRNL receives control, it completes its own initialization (which requires an additional two segments of free real memory) and starts to request its system files (CLOCK01.SYS, RESOURCE.SYS and so on) from the server through the mini-FSD and the real mode NetBIOS interface. Figure 6 shows the flow of control when loading a file at this point.

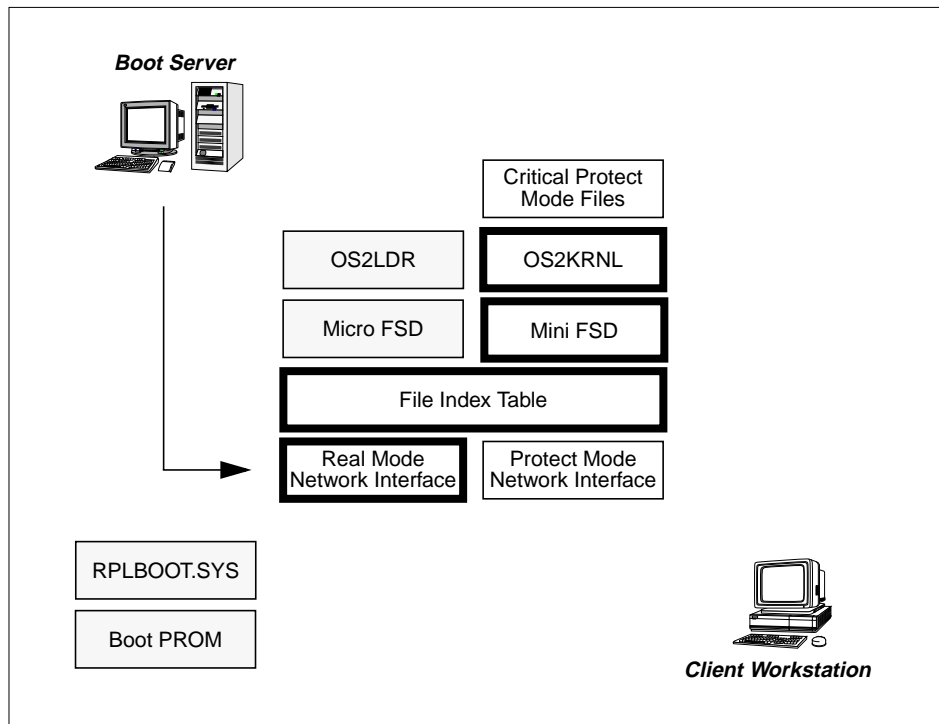


Figure 6. IEEE 802.2 Boot - Phase Three

After all the vital non-configurable system files are loaded, the client's CONFIG.SYS file is read, and the relevant BASEDEV drivers are loaded in the order in which they are listed in the CONFIG.SYS file. When the BASEDEV drivers are loaded, the regular device drivers and installable file system drivers are loaded in the order in which they are listed in CONFIG.SYS.

At this point, the number of steps required to load each file is quite large due to the need for the mini-FSD to constantly switch from protect mode to real mode and back again for each file. This means that the loading process is quite slow, and it is therefore important for the client to load a protect mode installable file system at the earliest possible opportunity. If you examine the

CONFIG.SYS file for a WorkSpace On-Demand client, you will notice that the transport files to load the protect mode FSD are loaded as early as possible when processing the CONFIG.SYS entries.

OS2KRNL requires a protect mode installable file system that can access files from the server, and WorkSpace On-Demand therefore uses the LAN Requester SMB redirector. The redirector uses the file index table to map the operating system's file I/O requests to the appropriate locations on the server. Figure 7 shows the flow of control at this point in the boot process.

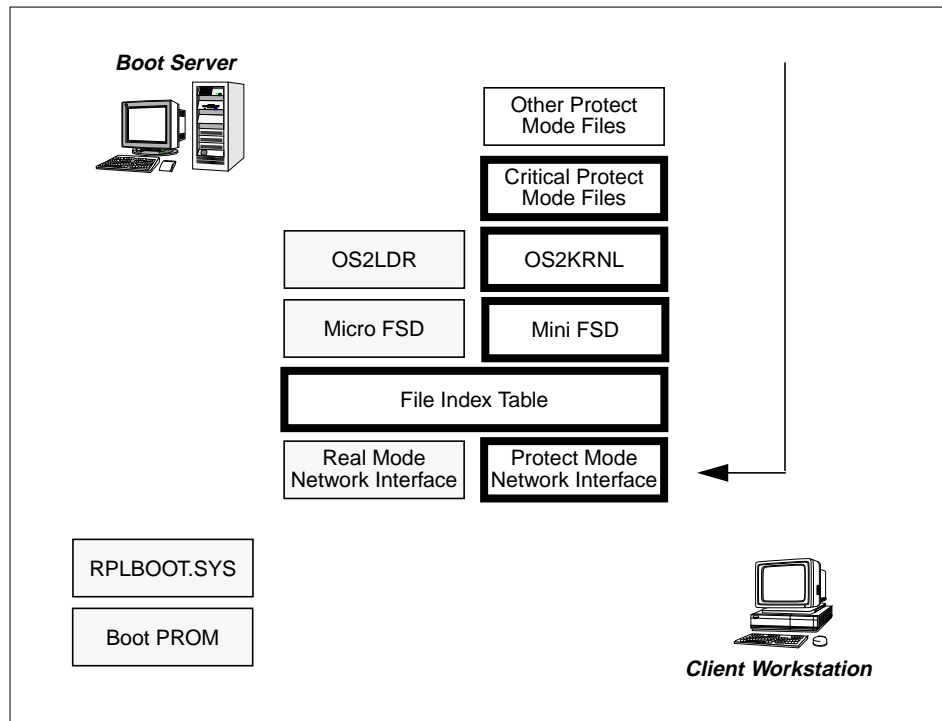


Figure 7. IEEE 802.2 Boot - Phase Four

The NetBIOS interface that supports the redirector's communication with the server must be loaded and active before the redirector itself is loaded. Initializing the redirector terminates the real mode mini-FSD, performs a netbind and reconnects to the server. Although there is a break in communication with the server between the time at which the real mode interface terminates and the protect mode interface is established, this occurs after all the code needed to initialize the redirector has been downloaded; so no server communication is required during this time.

From this stage onward, the SMB redirector, in conjunction with the file index table, uses the protect mode NetBIOS interface to access files on the server. The boot process is now common to both RPL and DHCP boot mechanisms.

#### **2.3.4 Loading a Client to Use NetBIOS Over TCP/IP**

The boot mechanism described in Section 2.3.3 works well when a client workstation is configured to support only a native NetBIOS/NetBEUI interface, but creates a problem if the client must support NetBIOS over TCP/IP (TCPBEUI) instead of, or in addition to, the native NetBIOS/NetBEUI interface. This section describes the problem and how it is overcome in WorkSpace On-Demand 2.0.

In Section 2.3.3, we mentioned the fact that all configured NetBIOS interfaces must be loaded before the protect mode redirector is initialized. We also mentioned that NetBEUI can load before a netbind occurs since the netbind actually initializes the interface.

This is not the case with TCPBEUI. The OS/2 TCPBEUI interface loads as an executable and uses the IP interface that it is configured to use. The IP interface must be available in order for TCPBEUI to load (rather like the redirector needing all its network interfaces loaded), but in order for an IP interface to be available, the underlying network transport drivers must be loaded and active; that is, we must perform a netbind.

We therefore need to perform a netbind before we can begin to initialize TCPBEUI. However, a netbind will terminate the existing, real mode interface to the network. The client is therefore left in a state where it has not yet downloaded all the files it needs to establish a protect mode interface, but has terminated its real mode interface.

This problem required a new method of downloading files, whereby files needed by the system at a later time could be downloaded from the server while the read mode interface was still active and cached in the client's memory for later use. WorkSpace On-Demand 2.0 provides a modified mini-FSD to provide this function.

To cache files with the mini-FSD, there are several prerequisites;

1. The mini-FSD must have a list of files to cache.
2. The client operating system must provide a place to put them.

The new mini-FSD uses an enhanced FIT file to provide the list of files that must be cached. Figure 8 shows a section of the new FIT file.

```

Z:\OS2\DLL\BVHVGA.DLL    BB20.US\OS2\DLL\BVHVGA.DLL @
Z:\OS2\DLL\DOSCALL1.DLL BB20.US\OS2\DLL\DOSCALL1.DLL @
Z:\OS2\DLL\OS2CHAR.DLL  BB20.US\OS2\DLL\OS2CHAR.DLL @
Z:\OS2\DLL\QUECALLS.DLL BB20.US\OS2\DLL\QUECALLS.DLL @
Z:\OS2\DLL\BVSCALLS.DLL BB20.US\OS2\DLL\BVSCALLS.DLL @

```

Figure 8. Enhanced FIT File with Mini-FSD Cache List

This looks very similar to a normal FIT file, but some of the entries have an @ sign at the end of their lines. This indicates to the mini-FSD that it must cache this file entry. Note that the @ signs are only included for clients that will use TCPBEUI as the boot protocol.

When OS2LDR initializes the mini-FSD, the mini-FSD reads the FIT file, immediately downloads all files with an @ sign to the client, and places them in an area between the 10 and 16 MB regions of memory. It can then access this area in a similar manner to a VDISK as shown in Figure 9.

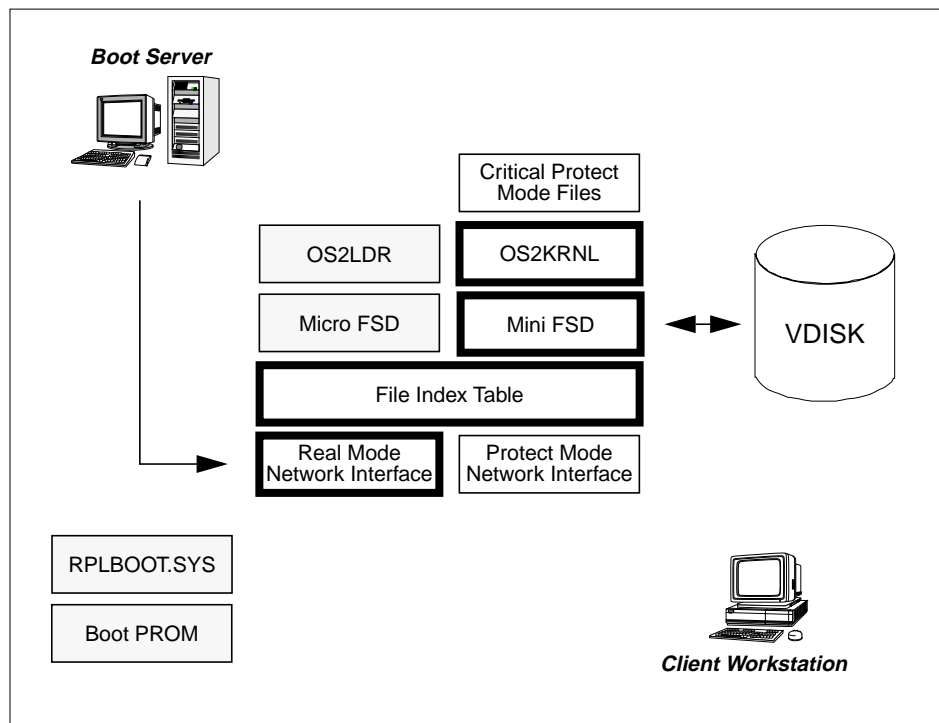


Figure 9. IEEE 802.2 Boot - Mini-FSD Caching Mechanism



Because of the area in memory reserved for use by the new mini-FSD, the OS2KRNL and loader needed to be modified to prevent them using the same area. For this reason, WorkSpace On-Demand 2.0 also includes new OS2LDR and OS2KRNL modules.

**Note**

Both the new mini-FSD and new OS2LDR/OS2KRNL are required to support TCPBEUI in WorkSpace On-Demand 2.0, and WorkSpace On-Demand 1.0 *will not* be updated to provide this function.

When the necessary files have been cached at the client by the mini-FSD, communication can be terminated by the netbind before the protect mode redirector has loaded. TCPBEUI and the redirector now load from the local cache, and communication with the server is then reestablished when the redirector is initialized. The mini-FSD has also been modified in WorkSpace On-Demand 2.0 to free its own memory and cache area as part of its termination.

From this point onward, the redirector is now used to access the server. The boot process now proceeds in a similar manner to that already described for a NetBIOS-only RPL boot mechanism.

### 2.3.5 Server Services

WorkSpace On-Demand uses two key services at the boot server to support WorkSpace On-Demand clients. These services are:

- The RIPL (REMOTEBOOT) service, which receives and responds to a client's boot request and which prepares and sends the boot block to the client.
- The File and Print Sharing (SERVER) service, which services file I/O requests from the micro-FSD, mini-FSD and protect mode redirector during the boot process.

Note that after the REMOTEBOOT service sends the boot block to the client, it is no longer involved in the boot process. From that point onward, the SERVER service assumes sole responsibility for serving operating system files to the client.

## 2.3.6 Key Server Files Used by the IEEE 802.2 RPL Boot Mechanism

There are a number of files that are crucial to WorkSpace On-Demand's implementation of the IEEE 802.2 RPL boot mechanism. The following sections describe each file.

### 2.3.6.1 RPL.MAP

Section 2.3, "The IEEE 802.2 RPL Boot Mechanism" on page 45, describes how a server receives a FIND frame from a client and determines whether it should respond to the client and if so, how it should respond. Under OS/2 Warp Server and WorkSpace On-Demand, these two decisions are made by consulting a file named RPL.MAP. This file resides in the \IBMLAN\RPL directory on the server.

The RPL.MAP file contains two basic types of records: workstation records and server records, as shown in Figure 10.

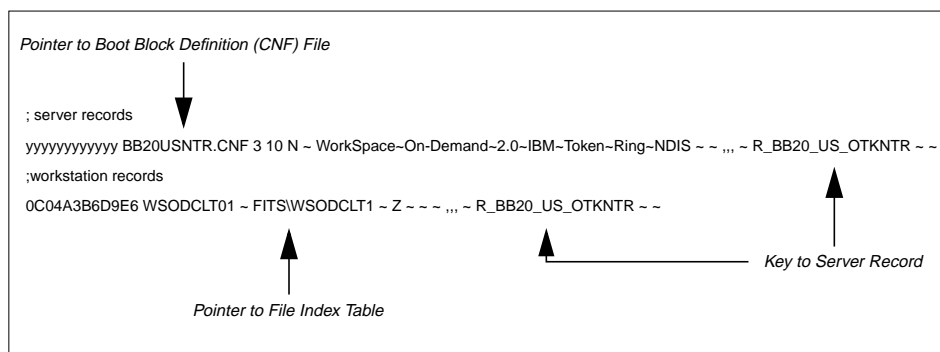


Figure 10. Sample RPL.MAP File

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 information required 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 10) 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 will be 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 2.3.6.2, "Boot Block Definition (CNF) File" on page 63.

**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, which serves as a place holder. Table 3 on page 57 describes each of the fields in the workstation record.

Table 3. 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, which are used as wildcards. Do not use these wildcard characters in OS/2 or WorkSpace On-Demand Remote IPL records.

Field	Name/Value	Description
2	WSODCLT01	Workstation machine ID. Can be up to 15 characters in length.
3	-	Not used. Must contain a tilde character.
4	\FITS\WSODCLT01	<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 8 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	BBSRV01	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 BBDOM	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.
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> <p>The DXM*.SYS drivers are obsolete and do not support many currently available network adapters. For this reason, these parameters are normally set to the tilde character.</p>

Field	Name/Value	Description
10	,,,	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.  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.
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 4 on page 59 describes each field in the server record.

*Table 4. RPL.MAP File - Server Record Field Descriptions*

Field	Name/Value	Description
1	yyyyyyyyyyyy	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 2.3.6.2, "Boot Block Definition (CNF) File" on page 63.

Field	Name/Value	Description
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.
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.

Field	Name/Value	Description
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 yyyyyyyyyyy 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\_BB2vv\_nn\_aaapppp...

- 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\_BB20\_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\_BB2vv\_nn\_aaapppp...

- 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.

**Note**

The server record identifiers are used only for standard OS/2 and DOS RIPL. If you do not follow the naming convention described here, the server record identifiers will not be selectable from the graphical user interface for DOS and OS/2 requesters. However, they may be used from the command line.



### 2.3.6.2 Boot Block Definition (CNF) File

Section 2.3, "The IEEE 802.2 RPL Boot Mechanism" on page 45, 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 are determined by the server using two sources:

- The FIT 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.

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 11 on page 64 shows the default WorkSpace On-Demand CNF file for the IBM Token-Ring Network Adapter.

```

; OS/2 Boot Block Configuration
; (IBM Token-Ring Compatible Adapter)
RPL DOS\RPLBOOT.SYS
DAT DOSMFSD20.SYS
ORG 1000H
LDR BB20.US\OS2LDR ~ OS2LDR UFSD.SYS MFSD30.SYS
DATA DOS\UFSD.SYS
DAT DOS\TOKENRNG\OS2\PROTOCOL.INI
DAT C:\IBMLAN\DOSLAN\DSP\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\DXMT0MOD.SYS PBA=0~S=12~ST=12~C=14~O=N ~ ~
; DRV C:\IBMLAN\DOSLAN\LSP\DXME0MOD.SYS ~ 10 ~
; ** NETBIOS and IEEE 802.2*****
;
; **NETBIOS*****
DRV C:\IBMLAN\DOSLAN\LSP\DXMJ0MOD.SYS ~ 6 ~
; **NETBIOS*****
DRV C:\IBMLAN\DOSLAN\LSP\DXMA0MOD.SYS 001 ~ ~
DRV C:\IBMLAN\DOSLAN\LSP\DOS\IBMTOK.DOS ~ ~ ~
DRV C:\IBMLAN\DOSLAN\LSP\PROTMAN.DOS /I: ~ ~

```

Figure 11. Boot Block Definition File

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 5 on page 65 describes the entries in the CNF file along with their expected file names and parameters.

Table 5. *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 MFSD30.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.
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	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'.

### 2.3.6.3 Machine FIT File

One of the items that is included in the boot block sent to the client by the server is a file index table (FIT). A FIT allows the client operating system's FSDs (micro-FSD, mini-FSD and the protect mode redirector) to accept file I/O requests for files located on the client's "boot drive" and redirect them to the correct locations on the server. Figure 12 on page 66 shows an example of a FIT entry.

```
Z:\OS2\*.INI \\TESTSVR\WRKFILES\WSODCLT01\OS2\OS2.INI
```

Figure 12. Sample FIT Entry

The entry shown in Figure 12 redirects any requests for Z:\OS2\\*.INI to the OS2.INI file that resides in the \WSODCLT01\OS2 directory under the network alias WRKFILES on the server named TESTSVR. See Chapter 3, "Understanding File Index Tables" on page 85, for a detailed discussion of FITs and their use.

The FIT created at boot-time and used to boot the client operating system is loaded from a file known as the machine FIT file. Each client workstation has its own unique FIT file, which is referenced by the workstation record in RPL.MAP.

---

## 2.4 The DHCP PXE Boot Mechanism

Since the IEEE 802.2 RPL boot mechanism was developed in the late 1980s, there have been significant technological advances that make it possible to provide more function in a Boot PROM. DHCP PXE is a boot mechanism developed to take advantage of the Dynamic Host Configuration Protocol (DHCP) functionality on a TCP/IP network, and it utilizes Intel's Preboot Execution Environment (PXE) technology.

### 2.4.1 Supporting a DHCP PXE Boot Environment

In order for a client workstation to load its operating system from a server using DHCP PXE, both the client and the server must satisfy certain requirements, as described in the following sections.

#### 2.4.1.1 Client Workstation

The client must have the ability to determine, during Power-On Self Test (POST), that it must load its operating system code from a network rather than a local device. On a PC, this ability is typically provided by the system BIOS and is configured by selecting the network as the startup device.

The client must then broadcast a request over the network and ask for the information it requires. This function is performed by a Boot PROM that supports the DHCP PXE boot mechanism. A DHCP PXE client requires the following information:

1. An IP configuration (including IP address, router address, and so on). This can be provided using a normal DHCP DISCOVER request broadcast by a Boot PROM that supports DHCP PXE.
2. The address of the *Boot Information Negotiation Layer (BINL) server* containing the boot block that the client is to use. The client must request this information in addition to the normal DHCP request. The Boot PROM does this by using some of the previously reserved bits in PXE options in the DHCP DISCOVER request frame.
3. The name of the file on the server that contains the client's bootstrap routine. The client obtains this information from the BINL server.

#### **2.4.1.2 Boot Server**

When using the IEEE 802.2 RPL boot mechanism, OS/2 Warp Server's REMOTEBOOT service provides all of the boot server functions necessary to support remote clients. When using the DHCP PXE boot mechanism, however, the requirements for the server are somewhat more complex. The server must be able to receive a request from a DHCP PXE client and respond with the information that the client requires. The server must provide the following functions:

1. The server must provide an IP configuration to the client.

In order to boot over a TCP/IP network, a client must first have its own IP address. This enables the boot server to route its responses to the client. In a DHCP environment, IP addresses are dynamically allocated by a DHCP server when requested. A client may use a different IP address each time it boots.

The allocation of IP addresses is provided by the normal DHCP service of a DHCP server. OS/2 Warp Server includes a DHCP service that supports this function, or you can use an existing DHCP server on your network.

2. The server must provide the IP address of the BINL server containing the client's bootstrap routine.

A normal DHCP server provides only an IP configuration to the client. However, the PXE standard introduces an extension to the DHCP protocol that provides the client with the address of the *Boot Information Negotiation Layer (BINL) server* it must use to obtain its bootstrap routine. This extension is known as the *PXE support extension*.

If the main DHCP server on your network does not support PXE, you can use a proxy DHCP server that includes the PXE support extension. WorkSpace On-Demand 2.0 includes a PXE support extension for OS/2 Warp Server's DHCP service (PXEPROXY), and this extension is installed and the DHCP service updated when you install WorkSpace On-Demand 2.0. The updated DHCP service allows OS/2 Warp Serve to act as its own DHCP server or to compliment an existing DHCP server that does not support the PXE environment (as a *DHCP Proxy server*).

3. The server must provide the IP address of the server and the name of the file containing the client's bootstrap routine. The BINL server provides this information to the client. OS/2 Warp Server includes a BINL service.
4. The server must provide a mechanism whereby the client can download the bootstrap routine. This mechanism is provided by a TFTP server. OS/2 Warp Server includes a TFTP server service (TFTPD).

Note that these functions can be performed by different services running on the same server or by different servers on the network. In a WorkSpace On-Demand 2.0 environment, the PXEPROXY, BINL and TFTPD services are typically combined on a single server known as the *installation server*. Note that the TTFD service must run on the WorkSpace On-Demand server.

## 2.4.2 The Boot PROM and DHCP PXE

When a client workstation is configured to boot from the network, the BIOS passes control to the Boot PROM when the machine is turned on and the Power-On Self Test (POST) sequence completes.

### Note

WorkSpace On-Demand 2.0 supports only those network adapters that include a PXE-compliant Boot PROM or offer such a Boot PROM as an option. Other network adapters that may provide BOOTP support are not supported by WorkSpace On-Demand

The Boot PROM then contacts the required server services to obtain a bootstrap routine. The code within the Boot PROM initializes the network adapter and sends out a DHCPDISCOVER packet on the standard DHCP port (67). An option field in this packet contains the following:

- A tag for client identifier (if the client identifier is known)
- A tag for the client Network Interface Identifier
- A tag for the client system architecture

Figure 13 illustrates the interaction between the client's Boot PROM and the various server services required to obtain a bootstrap routine.

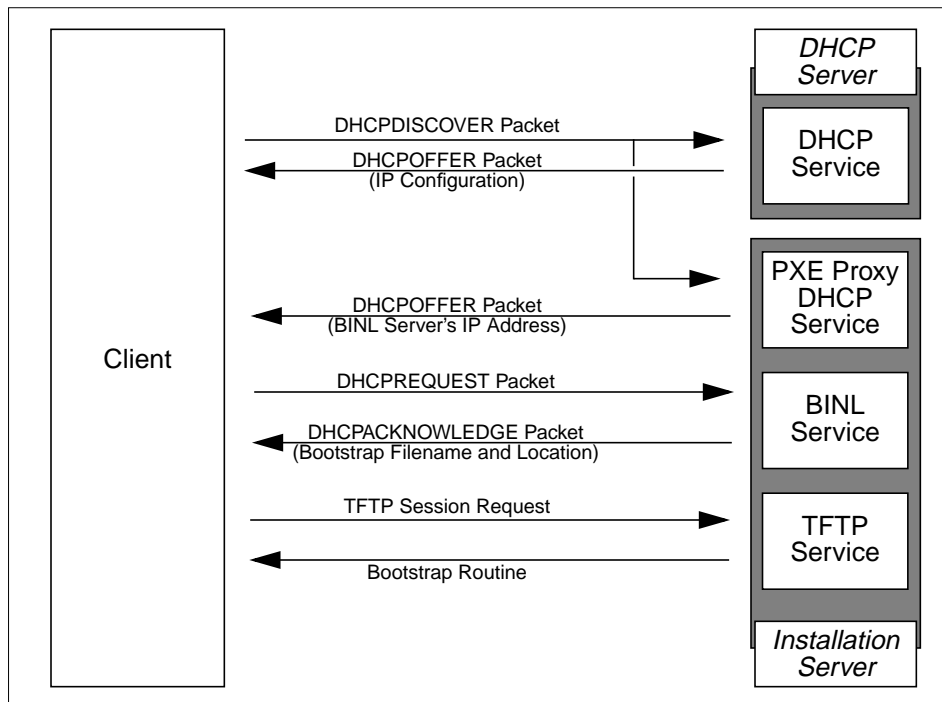


Figure 13. DHCP PXE Boot Mechanism

A DHCP server will respond by offering an IP configuration in a DHCPOFFER packet on port 68. In a PXE Proxy server environment, the PXE Proxy DHCP server will also respond with a DHCPOFFER packet on the same port. The DHCPOFFER packet sent by the PXE Proxy DHCP server contains the address of the BINL server in the *siaddr* field.

If the DHCP server is running the PXEDHCP service, it may respond with a single DHCPOFFER packet on port 68 containing both the IP configuration and the address of the BINL server.

Note that any DHCP server on the network can reply to the client's request with a DHCP OFFER packet. The client can choose which DHCPOFFER packet it will accept.

The timeout for a reply from a DHCP server is standard. The timeout for rebroadcasting to receive a DHCPOFFER packet with PXE extensions or a Proxy DHCP OFFER packet is based on the standard DHCP timeout, but is

substantially shorter to allow reasonable operation of the client in standard BOOTP or DHCP environments that do not provide a DHCP OFFER packet with PXE extensions. The PXE timeout for rebroadcast is 4, 8 and 16 seconds, yielding three broadcasts and a timeout after 28 seconds. The PXE timeout for rebroadcast is 4 seconds after receiving a DHCP OFFER without PXE extensions but with a valid *bootfile name* option.

The client uses the IP address contained in the *siaddr* field of the DHCP OFFER packet to contact the BINL server by sending a DHCPREQUEST packet to the BINL server on port 4011. This packet contains the following information:

- The IP address assigned to the client by the IP configuration received from the main or PXE Proxy DHCP server
- All the PXE options fields received from the selected DHCP OFFER packet which contained the PXE options

The BINL server sends a DHCPACKNOWLEDGE packet to the client, also on port 4011. This reply packet contains:

- The bootstrap file name and location
- The Client UUID/GUID option in the PXE proxy DHCP offer
- MTFTP configuration parameters

The client downloads the bootstrap routine using standard TFTP or MTFTP protocols through TFTP client code built into the DHCP PXE Boot PROM. The file downloaded and the placement of the bootstrap routine in memory is dependent on the client's CPU architecture.

The Boot PROM now passes control to the bootstrap routine, which now determines the behavior of the client for the next phase of the boot process. The bootstrap routine uses the TFTP client interface built into the Boot PROM to reconnect to the server and access the files it requires.

### 2.4.3 Loading a Client

Loading the WorkSpace On-Demand client operating system using the DHCP PXE boot mechanism is quite similar to the process by which the IEEE 802.2 RPL boot mechanism loads a client to use NetBIOS over TCP/IP. The bootstrap routine obtained by the client uses the TFTP interface built into the PXE Boot PROM to contact the boot server and download the files it needs to initialize the operating system, up to and including initializing the protect mode redirector. After the protect mode redirector is initialized, the client reconnects to the server and downloads the remaining files.



When the bootstrap routine, which for DHCP PXE clients is named RPLBOOTP.SYS, first connects to the server, it requests a configuration file. The name of this file is composed of the last eight digits from the Universally Administered Address (UAA) of the client's network adapter, with an extension of .INF. For example, a client with a UAA of 0004ACEC2E6E would request a file named ACEC2E6E.INF. For the rest of this discussion, we will refer to this file as *client*.INF. See Section 2.4.4.1, "Client.INF" on page 72 for more information on this file.

The *client*.INF file contains a list of client-specific files required to boot this particular client. It also contains a reference to one of two *common descriptor files* named BPCOMMON.xxx, where xxx refers to the bus type in the client. See Section 2.4.4.2, "Common Descriptor File" on page 73, for more information on the common descriptor files.

The bootstrap routine now uses the TFTP interface built into the PXE Boot PROM to download the files listed in the *client*.INF and BBCOMMON.xxx files. It caches these files in an area of high memory from where the bootstrap routine can access them as a "virtual disk" as shown in Figure 14. After all the required files are downloaded and cached, the bootstrap routine passes control to OS2LDR (which is one of the files that it has downloaded).

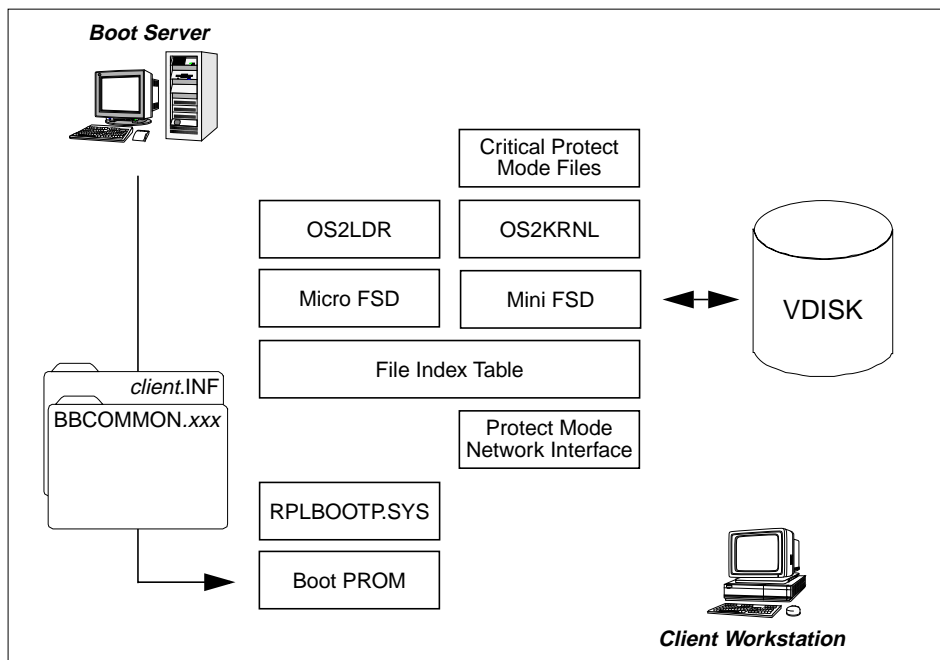


Figure 14. DHCP PXE Boot - Caching Mechanism

**Note**

This process represents the most fundamental difference between the IEEE 802.2 RPL and DHCP PXE boot mechanisms. When using DHCP PXE, all the code necessary to load and boot the operating system, up to the point of initializing the protect mode redirector, is downloaded using the TFTP interface built into the boot PROM. There is hence no need for a real mode redirector when using DHCP PXE.

From this point on, the boot process proceeds in a very similar manner to that discussed in Section 2.1.3, "Loading the WorkSpace On-Demand Client Operating System" on page 42. The micro-FSD used by OS2LDR and the mini-FSD used by OS2KRNL both access their files from the "virtual disk" in memory.

The client initializes its own protect mode redirector (this is the same LAN Requester redirector NETWKSTA.200 as is loaded by clients using the IEEE 802.2 RPL boot mechanism). The client then reconnects to the boot server using this redirector, and the remaining files are then downloaded using NetBEUI or TCPBEUI, depending on how the client is configured.

## 2.4.4 Key Client Files Used by DHCP PXE

The DHCP PXE boot mechanism uses a number of files to configure the client operating system environment. These files are described in the following sections. Note that these files, like all other files in a WorkSpace On-Demand environment, reside on the server's hard disk.

### 2.4.4.1 *Client*.INF

When you define a client to use the DHCP PXE boot mechanism, WorkSpace On-Demand creates a *client*.INF file, where *client* is the last eight digits from the UAA of the client's network adapter. This *client*.INF file resides in the \BMLAN\RPL\MACHINES\MACADDR\*macaddr* subdirectory, where *macaddr* is the first four digits from the UAA of the client's network adapter. For example, a client with a UAA of 0004ACEC2E6E would have a *client*.INF file named \BMLAN\RPL\MACHINES\0004\ACEC2E6E.INF.

The *client*.INF file contains information about all the client-specific files required by the WorkSpace On-Demand client operating system to load and boot itself, up to the point where it initializes the protect mode redirector. Figure 15 on page 73 shows an example of a *client*.INF file.

```
0004acec2e6e BJ100 ~ Fits\BJ100 BB2SVR Z ~ ~ ~ , , , ~ R_BB2_cc_BOOTP
\bb20.us\rplbootp.msg
\bbb20.us\bpcommon.isa
\fits\BJ100.FIT
\machines\bj100\ibmcom\macs\ibmtrp.os2
\machines\bj100\ibmcom\al.msg
\machines\bj100\ibmcom\alh.msg
\machines\bj100\config.sys
\machines\bj100\ibmlan\ibmlan.ini
\machines\bj100\ibmcom\protocol.ini
\machines\bj100\mptn\bin\setup.cmd
\machines\bj100\mptn\etc\dhcpcd.chf
\machines\bj100\mptn\etc\services
```

Figure 15. Client.INF File

The first line in the *client*.INF file is client-specific configuration information, including the UAA of the network adapter. This line is the same as the workstation record for this client in the RPL.MAP file (The RPL.MAP entry is actually a dummy entry that simply provides the OS/2 Warp Server GUI and command line interface with a way to show the client's configuration).

The *client*.INF file includes the name of the FIT file that the client's protect mode redirector will use to access files on the boot server. This is the fourth line in the file shown in Figure 15.

The *client*.INF file includes a pointer to the common descriptor file BPCOMMON.xxx. See Section 2.4.4.2 for more information on this file.

#### 2.4.4.2 Common Descriptor File

The *client*.INF file contains a pointer to a common descriptor file that contains all of the common (that is, not client-specific) files required to load and boot the WorkSpace On-Demand client operating system. There are two different common descriptor files:

- BPCOMMON.ISA is used for ISA and PCI bus clients.
- BPCOMMON.MCA is used for Micro Channel clients.

These files all reside in the \IBMLAN\RPL\BB20.US\IBMLAN\NETPROG directory. Figure 16 on page 74 shows an excerpt from a common descriptor file.

```
      :  
      \bb20.US\os2\keyboard.dcp  
      \bb20.US\os2\system\country.sys  
      \bb20.US\os2\boot\kdbase.sys  
      \bb20.US\ibmcomprotocol\netbeui.os2  
      \bb20.US\ibmlan\netprog\netwksta.200  
      :
```

Figure 16. Common Descriptor File BPCOMMON.xxx

When you add a base device (that is, a device driver referenced using a BASEDEV statement in the client's CONFIG.SYS file), you must add a line to the common descriptor file identifying the path to that device driver since the driver must be downloaded. You must also update the client's CONFIG.SYS file to include the appropriate BASEDEV statement.

#### 2.4.4.3 RFCNAMES.LST

The RFCNAMES.LST file resides in the \BMLAN\RPL\BB20.US\IBMCOM directory, and is used by a DHCP PXE client that will use TCPBEUI. WorkSpace On-Demand creates RFCNAMES.LST if it does not already exist. The file contains the NetBIOS name of the boot server and its corresponding IP address. RFCNAMES.LST is only used by B-node clients, but all TCPBEUI clients operate in B-node mode during the boot process and then switch modes if they are configured for another node type.

When you define a client to use the DHCP PXE boot mechanism and to use TCPBEUI as its boot protocol, WorkSpace On-Demand places the name and IP address of the boot server into the RFCNAMES.LST file. If you want the client to be able to access other servers in addition to the boot server, you must add their NetBIOS names and IP addresses to the RFCNAMES.LST file.

#### 2.4.4.4 RFCBCST.LST

RFCBCST.LST contains a list of subnets on which the client should broadcast a request if it cannot resolve a NetBIOS name using the list contained in RFCNAMES.LST.

### 2.4.5 Server Services

WorkSpace On-Demand uses a number of services on the server to support a WorkSpace On-Demand client booting using the DHCP PXE boot mechanism. These services are:

- The DHCP service (DHCP) provides an IP configuration to a client workstation.
- The PXE Proxy service (PXEPROXY or PXEDHCP) provides the address of a BINL server or, in the case of the PXEDHCP service, provides an IP configuration *and* the address of a BINL server, thereby combining the functions of the PXE Proxy and the DHCP services.
- The BINL service (BINL) provides the name and server location of the file containing the client workstation's bootstrap routine.
- The TFTP server service (TFTPD) accepts connection requests from the TFTP client interface built into the Boot PROM and downloads files to the client workstation using the TFTP protocol.
- The File and Print Sharing service (SERVER) accepts connection requests from the client workstation's protect mode redirector and downloads any remaining files using NetBEUI or TCPBEUI protocols.

If you contrast the behavior of the DHCP PXE boot mechanism with that of the IEEE 802.2 RPL boot mechanism, you can see that the functions performed by the DHCP, PXEPROXY, BINL and TFTP services for DHCP PXE are all performed by the REMOTEBOOT service for IEEE 802.2 RPL. The server-side functions of DHCP PXE are more complex, but this is offset by reduced complexity at the client, since there is no need for real mode redirectors and multiple reconnection to the server. The DHCP PXE boot mechanism also uses less memory and causes fewer interrupt conflicts, thereby allowing WorkSpace On-Demand to support a greater number of network adapters.

## 2.4.6 Key Server Files Used by DHCP PXE

There are a number of configuration files used by the DHCP PXE server services to define their behavior and the configuration data that they provide to clients. These files are described in the following sections.

### 2.4.6.1 PXEDHCP Server Configuration File

The PXEDHCP service and the PXEPROXY service store their configuration parameters in a file named DHCP.DHCP.CFG, which resides in the \MPTN\ETC directory. DHCP.DHCP.CFG is an ASCII file that can be edited using a normal text editor.

The DHCP.DHCP.CFG file for the PXEDHCP service includes IP configuration details for client workstations, as well as the BINL server address with which the PXEDHCP service should reply to each client. A sample DHCP.DHCP.CFG file for the PXEDHCP service is shown in Figure 17 on page 76.

```

leaseExpireInterval 20 Seconds
leaseTimeDefault 120 Minutes
allRoutesBroadcast no
UserMatchesVendorClass no
servertype pxedhcp #note this is the line that defines the role of the dhcp server
imageserver 10.0.0.1 #this defines the default image (BINL) server name

appendDomainName yes
#   Global options.  Passed to every client unless overridden at a lower scope.
#   domain name
option 15 np0150.itsc.austin.ibm.com

# Setup to send options to the pxeclient.  Sent in the encapsulated option 43.
vendor PXEClient
{
  option 1 1.1.4.5
  option 2 2
  option 3 3
  option 4 4
  option 5 5
}

subnet 10.0.0.0 255.255.255.0 10.0.0.50-10.0.0.80 (alias=np0150
{
  supportUnlistedClients both
  option 1 255.255.255.0 # subnet mask
  option 28 10.0.0.1 # broadcast address
  option 31 0 # no router discovery
}

# Sample config for a specific client machine
client 1 0x00104b335759 10.0.0.49 (alias=PC750
{
  option 1 255.255.255.0 # subnet mask
  option 12 ra0151c # TCP host name
  option 31 0 # no router discovery
}

# end of dhcpsd.cfg

```

Figure 17. PXEDHCP Server Configuration File DHCP.DCFG

The parameters contained within the DHCP.DCFG file are described in detail in the online **TCP/IP DHCP Administration** object, which resides in the **DHCP Server Services** folder within the **TCP/IP Shadows** folder on your OS/2 Warp Server desktop.

The DHCP.DCFG file used by the PXEPROXY service differs somewhat since it only needs to contain the PXE-related information, rather than all the configuration parameters necessary to supply IP configurations to client workstations. A sample DHCP.DCFG file for the PXEPROXY service is shown in Figure 18 on page 77.

```
logFileName dhcpd.log
logFileSize 100
numLogFiles 10
logItem SYSERR
logItem OBJERR
logItem WARNING
logItem INFO
leaseExpireInterval 20 Seconds
leaseTimeDefault 120 Minutes
supportBOOTP no
supportUnlistedClients no
allRoutesBroadcast no
UserMatchesVendorClass no
servertype pxeproxy
imageserver 10.0.0.1

appendDomainName yes
canonical no
proxyARec no
vendor PXEClient
{
  option 1 1.1.4.5
  option 2 2
  option 3 3
  option 4 4
  option 5 5
}
client 6 4000044347C1 ANY (alias=PXERIPL1
{
  bootstrapServer 9.180.120.115
}
```

Figure 18. PXEPROXY Server Configuration File DHCPD.CFG

You can make changes to the BINLSD.CFG file using the BINL Server Configuration utility, which resides in the **DHCP Server Services** folder within the **TCP/IP Shadows** folder on your OS/2 Warp Server desktop.

#### 2.4.6.2 BINL Server Configuration File

The BINL service stores its configuration parameters in a file named BINLSD.CG, which resides in the \MPTN\ETC directory. BINLSD.CFG is an ASCII file that can be edited using a normal text editor. A sample BINLSD.CFG file is shown in Figure 19 on page 78.

```

logFileName binlstd.log
logFileSize 100
numLogFiles 10
logItem SYSERR
logItem OBJERR
logItem WARNING
logItem INFO

tftp 10.0.0.1
bpname DOS\DHCPBOOT\RPLBOOTP.SYS
sa 0 (alias="Ethernet adapter 0" # This is a demo setup for an adapter type
{
  tftp 10.0.0.1
  bpname \dos\dhcpboot\rplbootp.sys
  client 1 4000044347c1 (alias=300gl # The 300gl in the software lab
  {
    tftp 10.0.0.2
    bpname \dos\dhcpboot\apctldr.img
  }
  client 1 4000044347c5 (alias=300gl-lab2 # The 300gl in the hardware lab
  {
    bpname \dos\dhcpboot\apctldr.img
  }
  client 1 1234567abcd3 (alias=Siemens # The siemens Nixdorf pc
  client 1 4000044347b1 exclude (alias=Gateway # The gateway box in the garage
}

```

Figure 19. BINL Server Configuration File BINLSD.CFG

You can make changes to the BINLSD.CFG file using the BINL Server Configuration utility, which resides in the **DHCP Server Services** folder within the **TCP/IP Shadows** folder on your OS/2 Warp Server desktop.

## 2.5 Server Directory Structures

The complexity associated with remote IPL is largely created by the many files that are used for configuration. While Workspace On-Demand insulates you from much of this complexity, it is nonetheless important to understand what is happening "under the surface" and the tasks that are being performed by the Workspace On-Demand Manager component.

The Workspace On-Demand client operating system is composed a series of files of different types. Table 6 shows these types along with an example of each.

Table 6. Client Operating System - File Types

File Type	Read-Only	Read/Write
<b>Generic (all clients)</b>	95% of code	Not Applicable
<b>Client-Specific</b>	CONFIG.SYS	LANTRAN.LOG



OS/2 Warp Server and WorkSpace On-Demand place these files in different locations on the server. Different types of files reside in different directory structures according to whether the files will be shared by all client workstations or specific to one client and whether they will be accessed in read-only or read/write mode. These directory structures, and the types of files that reside in them, are significant when you begin to define network applications and create machine classes to support additional types of hardware.

When you install the REMOTEBOOT service under OS/2 Warp Server, it creates two subdirectories:

- \IBMLAN\RPL\
- \IBMLAN\RPLUSER\

When you install WorkSpace On-Demand on an OS/2 Warp Server system, it creates a number of additional subdirectories within the \IBMLAN\RPL and \IBMLAN\RPLUSER directories, as shown in Figure 20 on page 80. These subdirectories within the \IBMLAN\RPL and \IBMLAN\RPLUSER directories contain the different types of files listed in Table 6 on page 78.

**Note**

During the course of this redbook, we assume that the U.S. English version is installed. We therefore use BB20.US to refer to the U.S. English version of WorkSpace On-Demand 2.0.

If you are using a different language version of WorkSpace On-Demand 2.0, you should substitute the appropriate language code in place of the "US" in the directory name. For example:

- EN refers to the U.K. English version.
- DE refers to the German version.
- JP refers to the Japanese version.
- FR refers to the French version.

and so on.

Note that while the directory structure implies that you can install multiple language versions of the WorkSpace On-Demand client operating system on a single server, IBM does not recommend or support such installations.

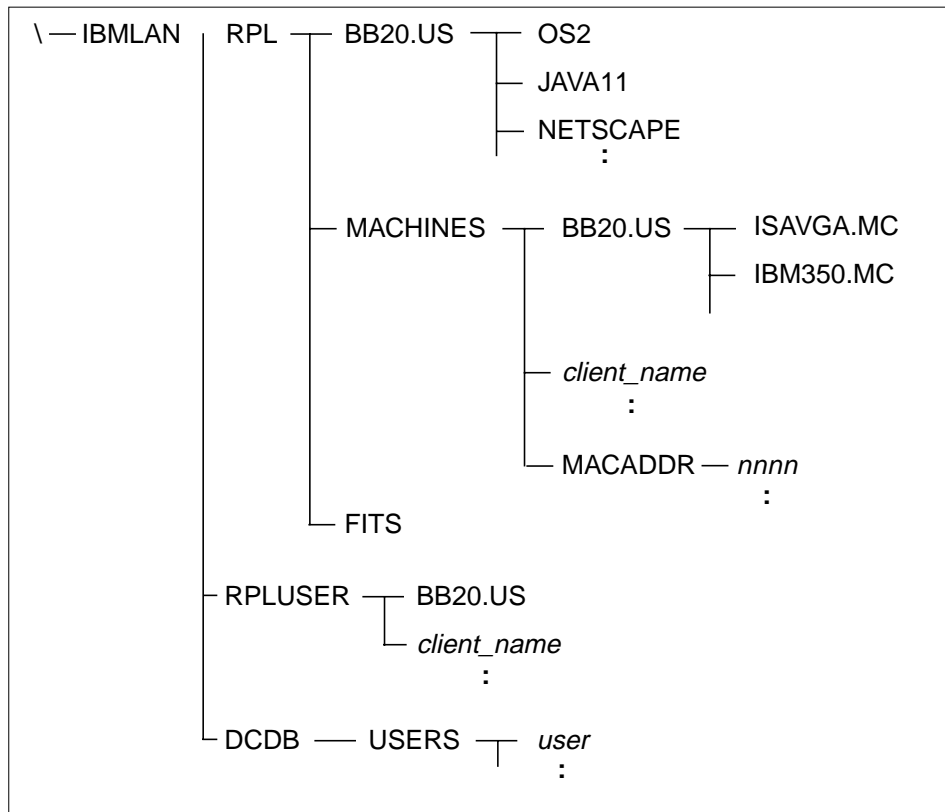


Figure 20. Remote IPL Directory Structure

### 2.5.1 Generic Directories and Files

The \IBMLAN\RPL\BB20.US directory and its subdirectories contain generic, read-only operating system files. Device drivers, executables and dynamic link libraries are among the file types located here.

Application files may also reside here. For example, if you select Netscape Navigator during installation, the WorkSpace On-Demand installation program installs it into the \IBMLAN\RPL\BB20.US\NETSCAPE directory.

#### Note

When the remote boot service is started, this directory is shared under the RPLFILES network alias.

## 2.5.2 Client-Specific Directories and Files

When you define a client workstation, WorkSpace On-Demand automatically creates two directories that are specific to that client:

- \BMLAN\RPL\MACHINES\*client\_name*
- \BMLAN\RPLUSER\*client\_name*

where *client\_name* is the name that you specified in the client definition.

If you define a client to use the DHCP PXE boot mechanism, WorkSpace On-Demand creates a \BMLAN\RPL\MACHINES\MACADDR\*nnnn* directory, where *nnnn* is the first four digits of the Universally Administered Address for the client's network adapter. Note that this directory is created only once and is used to store files relating to any clients for which the Universally Administered address begins with *nnnn*.

### 2.5.2.1 Read-Only Directories and Files

The \BMLAN\RPL\MACHINES directory contains a unique subdirectory for each client workstation defined on the server. This subdirectory is named \BMLAN\RPL\MACHINES\*client\_name* and contains client-specific files to which the client requires only read access and not write access. Files such as the client's CONFIG.SYS, PROTOCOL.INI, and so forth reside here.

For clients that will use the DHCP PXE boot mechanism, WorkSpace On-Demand stores the *client*.INF file (described in Section 2.4.4.1, "Client.INF" on page 72) in the \BMLAN\RPL\MACHINES\MACADDR\*nnnn* directory. Since multiple clients can have Universally Administered Addresses beginning with the same four digits, there may be multiple client.INF files in the same directory.

Note that this directory is only created for clients that you define to use the DHCP PXE boot mechanism. It is not created for clients that use the IEEE 802.2 RPL boot mechanism.

The \BMLAN\RPL\MACHINES\BB20.US subdirectory contains the various machine class templates. The files in this directory structure contain entries specific to a particular machine class and are used to create the client-specific, read-only files in the \BMLAN\RPL\MACHINES\*client\_name* directory when you define a client. See Section 10.1, "Machine Classes" on page 251, for a detailed discussion of machine classes.

Table 7 shows the client-specific, read-only files that WorkSpace On-Demand Manager places in the \BMLAN\RPL\MACHINES\*client\_name* directory when you define a client.

Table 7. Client-Specific Configuration Files - Read-Only Access

File	Description
CONFIG.SYS	This file is constructed at client definition time by building the network adapter entries required to support the appropriate network adapter and merging them with the standard entries already in the template CONFIG.SYS file for the machine class located in the \BMLAN\RPL\MACHINES\BB20.US\ <i>machine-class</i> directory.
<i>client_name</i> .INI	This file is copied from the machine class directory and is then modified based upon selections made by the administrator using the GUI or the command line interface.
\BMCOM\PROTOCOL.INI	This file is constructed using the THINLAPSR utility, from the PROTOCOL.INI file in the machine class directory and the adapter information contained in the NDISDD.PRO record for the network adapter selected by the administrator.
\BMLAN\BMLAN.INI	This file is constructed by copying the template file located in the machine class directory and adding the client name defined by the administrator.

Note that if you wish to add entries to the CONFIG.SYS file for your clients, or to modify the information contained in any of the other files listed in Table 7, you can do so by modifying the template files in the machine class directory, which are used to create the client-specific configuration files. If you do this before defining a client workstation, the modified files will automatically be copied into that client's directory.

FIT files are an exception to the general rules for placement of files. All machine FIT files reside in the \BMLAN\RPL\FITS directory. This includes the default machine FIT file for each machine class as well as the client-specific machine FIT file created for each client workstation when it is defined.

### 2.5.2.2 Read/Write Directories and Files

The \BMLAN\RPLUSER directory contains a unique subdirectory for each client workstation defined on the server. This subdirectory is named \BMLAN\RPLUSER\*client\_name* and contains all of the client-specific files to which the client requires read/write access.

The \BMLAN\RPLUSER\BB20.US directory contains templates for client-specific, read/write files. These templates are copied into each client-specific read/write directory (\BMLAN\RPLUSER\*client\_name*) when you define a client.

Table 8 shows the client-specific, read/write files that WorkSpace On-Demand Manager places into the \BMLAN\RPLUSER\*client\_name* directory when you define a client.

Table 8. Client-Specific Files - Read-Write Access

File	Description
AUTOEXEC.BAT	This file is constructed at client definition time by building the network adapter entries required to support the appropriate network adapter and merging them with the standard entries already in the template CONFIG.SYS file for the machine class to which this client workstation belongs located in the machine class directory.
\BMLAN\ACCOUNTS\NET.ACC	This file is copied from the machine class directory and is then modified based upon selections made by the administrator using the GUI or the command line interface.
\OS2\OS2.INI	This file is constructed by copying the template file located in the machine class directory and adding the client name defined by the administrator.
\OS2\OS2SYS.INI	This file is constructed by copying the template file located in the machine class directory and adding the client name defined by the administrator.
\OS2\MDOS\WINOS2 files: - ATM.INI - CONTROL.INI - MOUSE.INI - MSD.INI - P9000RES.INI - P9X00RES.INI - PROGRAM.INI - STARTUP.GRP - SYSTEM.INI - WIN.INI - WOS2ACCE.GRP - WOS2MAIN.GRP	These files are copied from templates in the \OS2\MDOS\WINOS2 subdirectory within the \BMLAN\RPLUSER\BB20.US directory structure. The only exceptions are the WIN.INI and SYSTEM.INI files, which are copied from the OS2\MDOS\WINOS2 subdirectory within the machine class directory.

**Note**

When the remote boot service is started, the \IBMLAN\RPLUSER directory structure is shared under the WRKFILES network alias.

If you wish to modify the information contained in any of the files listed in Table 8, you can do so by modifying the template files in the machine class directory, which are used to create the client-specific configuration files. If you do this before defining a client workstation, the modified files will automatically be copied into that client's directory.

### 2.5.3 User-Specific Directories and Files

Most user-specific directories and files are used by applications rather than by the WorkSpace On-Demand client operating system and are discussed further in Chapter 7, "Working with Network Applications" on page 167. The exceptions, however, are user FIT files, which contain FIT entries unique to a particular end user.

Each user defined in an OS/2 Warp Server domain has a unique subdirectory within the \IBMLAN\DCDB\USERS directory on the domain controller. This directory is named \IBMLAN\DCDB\USERS\user\_id and is created by OS/2 Warp Server when you create the user definition. WorkSpace On-Demand places user FIT files in this directory.

## Chapter 3. Understanding File Index Tables

This chapter describes the concept of the File Index Table (FIT), how it is used by WorkSpace On-Demand, and how you can modify FIT files to handle file redirection for your network applications.

The file index table (FIT) provides a mechanism for a client workstation to redirect file I/O requests to the Workspace On-Demand server. The FIT is loaded into the client's memory, and whenever a file access is requested by the WorkSpace On-Demand client operating system or an application running on the client, the FIT is checked to resolve the request.

FITs are defined in FIT files, which are ASCII text files that you can view or edit with a normal text editor.

There are three different types of FIT file:

- Machine FIT file
- User FIT file
- Application FIT file

Each type of FIT file has its own particular uses, which are explained later in this chapter.

---

### 3.1 FIT Entry Syntax

Figure 21 illustrates the standard format for a FIT file entry. The left-hand side of the entry lists the *prototype filename* by which the client operating system or an application refers to a file or directory. The right-hand side of the entry lists the location to which an access request for this file is redirected. The right-hand side of the entry is known as the *substitution text*, and typically comprises a UNC name that identifies the server, network alias and any remaining path and file name information.

```
Z:\NETSCAPE\NETSCAPE.INI \\MYSERVER\NETSCAPE\NETSCAPE.INI
```

Figure 21. Sample FIT Entry - File Redirection

The example shown in Figure 21 redirects a file access request for the file NETSCAPE.INI, which the client system assumes is located in the \NETSCAPE directory on its drive Z:, to the NETSCAPE alias on the server MYSERVER.

You can use a semicolon character (;) anywhere on a line of the FIT file to begin a comment. Blank lines are ignored.

Note that files do not necessarily need to be redirected down to the file level. You can specify redirection at the directory level if you wish, and all files within that directory will be redirected as indicated in the FIT entry. An example of such a redirection is shown in Figure 22 on page 86.

```
Z:\OS2\*.INI \\MYSERVER\WRKFILES\DEFAULT\OS2
```

Figure 22. Sample FIT Entry - Directory Redirection

If you wish, you can omit the UNC name from the substitution text and specify only a directory name and, if redirecting to the individual file level, a file name. If you do this, the redirector will assume the UNC name specified in the first entry in the machine FIT file (see Section 3.2, "The Machine FIT File" on page 87), and append the substitution text in the current entry to that UNC name.

Note that the ability to include or omit a UNC name in the substitution text allows a FIT to redirect file access requests to more than one server. For example, your WorkSpace On-Demand client operating system files will reside on your WorkSpace On-Demand server, but you may wish to access application or middleware files at boot-time from a different file or application server on your network.

### 3.1.1 Using Wildcard Characters in FIT Files

Under certain circumstances, you can use wildcard characters in your FIT entries. Table 9 shows the allowable wildcard characters.

Table 9. Wildcard Characters in FIT Entries

Wildcard	Example	Meaning
*	*.ini	This specifies any group of letters. (for example, *.ini would specify all files with a .ini extension).
?	os2.??i	This specifies a single character. (for example, test.??i would specify all files that start with test. prefix and end with a three letter extension the last letter of which is the letter i).

Wildcard characters (?) and (\*) are supported subject to the following rules:



- Wildcard characters can appear only in the prototype filename on the left-hand side of the entry. They cannot appear in the substitution text on the right-hand side of the entry.
- Wild card references can be redirected only to the directory level. For example, the following is valid:

```
Z:\OS2\*.INI \\BBSRV01\WRKFILES\DEFAULT\OS2
```

The following is not valid:

```
Z:\OS2\*.INI \\BBSRV01\WRKFILES\DEFAULT\OS2\ANY.INI
```

- Individual file names may only be mapped to an individual file name. For example, the following is not valid:

```
Z:\OS2\ANY.INI \\BBSRV01\WRKFILES\DEFAULT\OS2
```

since it maps a specific filename to a directory rather than an individual file

- No more than one wildcard entry can appear in the same prototype filename field. For example, the following is not valid:

```
Z:\OS2\ANY???.* \\BBSRV01\WRKFILES\DEFAULT\OS2
```

### 3.1.2 Using Variables in FIT Files

There are a number of variables that may be used in FIT files under certain circumstances. This capability allows you to create "standard" user FIT files that can be used by groups of users who have common application requirements.

- ? is resolved as the client's boot drive (typically Z:).
- <DCSERVER> is resolved as the OS/2 Warp Server domain controller.
- <USER> is resolved as the user name.
- <RPLBOOTSERVER> is resolved as boot server.
- <OS2VERSION> is resolved as the current OS/2 version being booted on the client workstation. This enables for different OS/2 versions to be managed on different clients by the same boot server.
- <MACHINE> is resolved as the client workstation name.

However, you may *only* use variables in a user FIT file or an application FIT file. You may *not* use variables in a machine FIT file.

---

## 3.2 The Machine FIT File

The machine FIT file contains redirection commands that are unique to a particular client workstation. The machine FIT file will always contain

redirection commands for the WorkSpace On-Demand client operating system, including device drivers, and can also contain commands for applications and middleware that you add to the WorkSpace On-Demand client image on your server.

For IEEE 802.2 RPL clients, the entries in the machine FIT file form part of the boot block that is downloaded to the client at boot-time, while for PXE DHCP clients, the machine FIT file entries are downloaded using TFTP as part of the first phase in the boot process. These entries are used by the micro-FSD, the mini-FSD and then by the FSD or redirector. The entries in the machine FIT file remain in memory until the user shuts down the client.

Each OS/2 or WorkSpace On-Demand client workstation that you define on your WorkSpace On-Demand server has its own unique machine FIT file. DOS clients do not require a FIT file. The machine FIT files for all clients reside in the \IBMLAN\RPL\FITS directory.

The machine FIT file is created automatically when you define a client, and is assembled from two components:

- A default WorkSpace On-Demand FIT file named DFBB20`cc`.FIT (where `cc` is a two-character language code), located in the \IBMLAN\RPL\FITS directory
- Hardware-specific redirection information, unique to the machine class for this client. These redirection commands are stored in a machine-class FIT file, located in the \IBMLAN\RPL\MACHINES\BB20.US\`machine_class` directory.

These two files are merged together when you define a client, to produce a client-specific machine FIT file. This is done in the following manner:

1. The default machine FIT file \IBMLAN\RPL\FITS\DFBB20`nn`.FIT is copied into the \IBMLAN\RPL\FITS directory to create a client-specific machine FIT named `client_id`.FIT.
2. All references to the client name in the client-specific FIT file are changed to reflect the name of the client. That is, all occurrences of the word DEFAULT are changed to `client_id`.
3. The client definition procedure then searches for a machine class FIT file in the \IBMLAN\RPL\MACHINES\BB20.`nn`\`machine_class`.MC directory corresponding to the machine class that you specified for this client. If the file exists, it is appended to the \IBMLAN\RPL\FITS\`machine_id`.FIT file.

The first entry in a machine FIT file *must* be a UNC name specifying the name of the default network alias on which files can be found. For example, a

header record specifying \\BBSRV01\RPLFILES would indicate that the default network share is RPLFILES on server BBSRV01.

The rest of the file consists of file-name mapping records, with the same syntax as described in Section 3.1, "FIT Entry Syntax" on page 85. The following are examples of machine FIT file entries:

1. Defines the default Remote IPL server and network share name.

```
\\BBSRV01\RPLFILES
```

2. Translates all references to Z:\CONFIG.SYS to MACHINES\DEFAULT\CONFIG.SYS:

```
Z:\CONFIG.SYS MACHINES\DEFAULT\CONFIG.SYS
```

When this is resolved the default net name is automatically prefixed to the translated name.

```
\\BBSRV01\RPLFILES\MACHINES\DEFAULT\CONFIG.SYS
```

3. Translates all references to Z:\OS2 to BB20US\OS2:

```
Z:\OS2 BB20US\OS2
```

Except for explicitly mapped filenames (see line 6), this record results in all Z:\OS2\\*. \* references being translated to:

```
\\BBSRV01\RPLFILES\OS2
```

4. Subdirectory references from the previous example are also mapped. For example, Z:\OS2\DLL\ANYFILE.DLL is translated to:

```
\\BBSRV01\RPLFILES\OS2\DLL\ANYFILE.DL
```

5. The following is a comment.

```
; Go to a different share
```

6. Translates all references to Z:\OS2\SYSTEM\SWAPPER.DAT to \\BBSRV01\WKRFILES\DEFAULT\OS2\SYSTEM\SWAPPER.DAT:

```
Z:\OS2\SYSTEM\SWAPPER.DAT
\\BBSRV01\WKRFILES\DEFAULT\OS2\SYSTEM\SWAPPER.DAT
```

In this example, the default network alias at the top of the FIT file is overridden by specifying an explicit UNC name in the substitution text. Requests will be redirected to the client-specific file structure to which the client has read, write, create, delete, and execute access.

Note that in most cases, we recommend using local swapping on the client rather than remote swapping to the server, as indicated here.

7. The following translates all other references to be relative to \\BBSRV01\WRKFILES\DEFAULT:

Z:\ \BBSRV01\WRKFILES\DEFAULT

Note that the translated path is to the requester-unique file structure on the server for which the requester has read, write, create, delete, and execute authority.

### 3.2.1 Access Control Profiles

Any server alias or file name that you include in the substitution text must have an access control profile that allows the client workstation to access the resource. Since entries in the machine FIT file are typically accessed before the user logs on, such access uses the client's user ID rather than the end user's user ID. OS/2 Warp Server creates a user ID for each client workstation when you define it, and adds that user ID to the RPLGROUP group, which has read and execute access to the server's \BMLAN\RPL\BB20.US and \BMLAN\RPL\MACHINES\BB20.US directories. The client workstation's user ID is given read/write access to the \BMLAN\RPLUSER\*client\_id* directory. If you add other network aliases to the machine FIT file, you must ensure that the client workstation's user ID has the appropriate level of access to the resources referenced by those aliases.

Note however, that some FIT entries may not actually be used until after the user logs on to the client, even though they are defined in the machine FIT file. In such cases, the user's access control profile, not the client's access control profile, is used. You must therefore ensure that the user has the appropriate level of access to the network resources referenced by such entries.

### 3.2.2 Default Machine FIT Files

A number of default machine FIT files are provided with OS/2 Warp Server and WorkSpace On-Demand 2.0. These files are located in the \BMLAN\RPL\FITS directory.

Table 10. Default Machine FIT Files

FIT File Name	Description
DEFAULT20.FIT	Used for OS/2 2.0 clients
DEFAULT21.FIT	Used for OS/2 2.1 clients
DEFAULT30.FIT	Used for OS/2 Warp 3.0 clients
DEFAULT40.FIT	Used for OS/2 Warp 4.0 clients
DFBB10US.FIT	Used for WorkSpace On-Demand 1.0 U.S. English clients
DFBB20US.FIT	Used for WorkSpace On-Demand 2.0 U.S. English clients

For WorkSpace On-Demand clients, a default machine FIT file is provided as part of each language client installation. For example: DFBB20US.FIT is the default machine FIT for U.S. English clients, whereas DFBB20DE.FIT is the default machine FIT for German clients.

### 3.2.3 Adding Your Own Entries to the Machine FIT File

If you add applications or middleware that are loaded into the client's memory at boot-time, you will need to add redirection statements to allow the client to locate the necessary files on the server. You can do this by editing the machine FIT file to add the records you require.

The best way to make additions to the machine FIT file is to modify the default machine FIT file (DFBB20US.FIT for WorkSpace On-Demand clients) prior to defining the clients. In this way, your additional statements will be added to the client's machine FIT file when the default machine FIT file is merged with the machine class FIT file at client definition time.

**Note**

If you add your information to the DFBB20US.FIT file after you have defined a client, you must delete and recreate the client definition to make these changes effective for that client.

---

## 3.3 The User FIT File

WorkSpace On-Demand supports multiple levels of file index tables within a single client. The first level is the machine FIT file, from which the entries are loaded into memory at boot-time. In addition to the machine FIT file, you may also specify user-specific redirection statements in a user FIT file, which is loaded into memory by the PMLOGON shell when a user logs on to a client.

The entries in the user FIT file are inserted into the existing in-memory copy of the FIT (generated at boot-time), immediately following the required UNC reference in the first record of the machine FIT file, and prior to any of the other statements. This means that entries in the user FIT file will override any conflicting entries in the machine FIT file.

**Note**

User FIT files may only be used with the PMLOGON shell. If you replace the PMLOGON shell with a customized shell, that shell will not be aware of the user FIT file.

When a user logs on to a WorkSpace On-Demand client, the PMLOGON shell searches the \BMLAN\DCDB\USERS\*user\_id* directory for a user FIT file. The \BMLAN\DCDB\USERS\*user\_id* directory is created in the normal way when a user is added to the domain. As part of this process, the directory is given an ACL to permit only user access.

If a user FIT file is found, PMLOGON updates the in-memory copy of the client's FIT to include the user FIT file entries. If a user FIT file is not present, PMLOGON uses the default user FIT file, which is described in Section 3.3.2, "Default User FIT File" on page 93.

A user FIT file is normally generated manually by a system administrator when defining a user's user ID in the OS/2 Warp Server domain. However, it is often more efficient to use the default user FIT file, and to use variables to tailor the entries for each user. See 3.1.2, "Using Variables in FIT Files" on page 87 for more information on using variables.

The format of a user FIT file entry is the same as that for a machine FIT file entry. However, the user FIT file does not contain a default network alias as the first entry. A user FIT file, for example, might contain the following entry:

```
Z:\NETSCAPE\NETSCAPE.INI \\BBSRV01\WRKFILES\USER01\NETSCAPE\NETSCAPE.INI
```

*Figure 23. User FIT File - User-Based File Redirection*

In this case, adding a user FIT file for each user allows you to use a different NETSCAPE.INI file for each user. Alternatively, you can use variables to tailor a single user FIT file to handle multiple users (see Section 3.3.2, "Default User FIT File" on page 93).

As with machine FIT files, you can include a UNC name in the substitution text, or omit it. If you omit the UNC name, the redirector assumes the UNC name specified in the first record in the machine FIT file for the client workstation, and appends the substitution text to that UNC name.

The machine FIT must also include the following line, mapping to the location of the user FIT.

```
z:\DCDB\USERS \BBSRV01\IBMLAN$\DCDB\USERS
```

Figure 24. User FIT File Entry - User FIT File Redirection

### 3.3.1 Access Control Profiles

The entries in the user FIT file are loaded into memory when a user logs on to a client workstation. In most cases, the network resources referenced by these entries will be accessed using the user's access control profile. You must therefore ensure that the end user has the correct level of access to the network resources referenced by user FIT file entries.

For this reason, you must not try to access the IBMLAN\$ alias on your domain controller when a user is not logged on. If an access attempt is made before a logon, the client workstation name is used to authenticate the access; since the client ID connections are not re-initialized at logon or logoff time, you will not be able to access the user's FIT file after logon.

### 3.3.2 Default User FIT File

If you do not create a user FIT file for a particular user, WorkSpace On-Demand will use a default user FIT file for that user. This default user FIT file is named BB20USDU.FIT, and resides in the \IBMLAN\DCDB directory. The default user FIT file supplied with WorkSpace On-Demand is shown in Figure 25 on page 93.

```
;Default user FIT
;   ?: is replaced with the client's boot drive
;   <DCSERVER> is replaced by the server name of Domain Controller
;   <USER> is replaced by the user id
;Example: ?:\APPID1\DATA   \\

```

Figure 25. Default User FIT File - BB20USDU.FIT

The default user FIT file contains entries for DAX support. However, you can add redirection commands for any other applications, just as you would to a user-specific FIT file. These commands will then be used by all WorkSpace On-Demand clients that use the default user FIT file.

Note that you can use variables to tailor a default user FIT file to suit the needs of individual users. See Section 3.1.2, “Using Variables in FIT Files” on page 87 for more information on these variables.

---

### 3.4 The Application FIT File

WorkSpace On-Demand 2.0 introduces the concept of application FIT files. An application FIT file contains entries similar to those in the machine and user FIT files, but the application FIT file entries are loaded into memory only when the application’s primary executable is loaded, and are discarded from memory when the application is terminated.

The in-memory FIT is restricted to 64KB in size, and with WorkSpace On-Demand 1.0 and earlier OS/2-based RIPL implementations, this created problems when loading, and therefore redirecting, large numbers of files concurrently, since all FIT entries needed to be loaded at boot-time or user logon-time.

The introduction of application FIT files in WorkSpace On-Demand 2.0 means that only the following FIT entries are present in memory:

- Machine FIT file entries
- User FIT file entries
- Application FIT file entries for those applications that are currently active.

Application FIT files therefore allow you to reduce the size of the in-memory FIT, helping to reduce the incidence of this problem in a WorkSpace On-Demand 2.0 installation.

An application FIT file normally resides in the same directory as the application’s primary executable; that is, the EXE file that launches the application. You may create up to two different application FIT files for each application. See Section 3.4.2, “Pre vs. Post Application FITs” on page 95 for more information on each of these types.

Application FIT files provide a useful means of handling user- and application-specific FIT entries. We strongly recommend that you use application FIT files in preference to user FIT files whenever possible.

Note that you can use variables to tailor an application FIT file to suit the needs of individual users. See Section 3.1.2, “Using Variables in FIT Files” on page 87 for more information on these variables.



### 3.4.1 Access Control Profiles

The entries in the application FIT file are loaded into memory when the user launches the application from the desktop. The user is therefore logged on to the client workstation before the application FIT file is loaded into memory and therefore, the network resources referenced by these entries will be accessed using the user's access control profile. You must therefore ensure that the end user has the correct level of access to the network resources referenced by application FIT file entries.

### 3.4.2 Pre vs. Post Application FITs

There are two types of application FIT file:

- APPPRE.FIT is loaded into the in-memory FIT before the entries from the user FIT file, and therefore the application FIT file entries in APPPRE.FIT will override any conflicting entries in the user FIT file.
- APPPOST.FIT is loaded into the in-memory FIT after the entries from the user FIT file, and therefore the user FIT file entries will override any conflicting entries in APPPOST.FIT.

You may use both types of application FIT with the same application, by creating both application FIT files and placing them in the same directory as the application's primary executable.

If you wish, you can use names other than APPPRE.FIT and APPPOST.FIT for your application FIT files. You can do this by using the following environment variables on the **Parameters** page in the **Application Definition** notebook.

`NCC_PREFITS=filename`

`NCC_POSTFITS=filename`

You can specify the file name without a path, in which case the file is assumed to reside in the same directory as the application's primary executable, or use a fully qualified path and file name, including a UNC name.

Note that if the APPPRE.FIT and APPPOST files exist and you also specify the environment variables to refer to other FIT files, the files specified in the environment variables will take precedence, and these files will be loaded instead of the APPPRE.FIT and APPPOST.FIT files.

### 3.5 Nesting and Combining FITs

In order to reduce the amount of work involved in creating user FIT files, you can use a combination of user-specific and default user FITs. The user-specific FIT can contain entries unique to that user, and can then invoke the default user FIT. Figure 26 on page 96 illustrates this technique.

```
;User-specific FIT entries

Z:\TOOLS\IMAGEDIT          \\SERVER01\TOOLS\IMAGEDIT

#include BB20USDU.FIT
```

Figure 26. Nesting FIT Files

The user FIT shown in Figure 26 contains a single redirection request for a user-specific application, then uses a `#include` statement to invoke the default user FIT. Note that, regardless of where the `#include` statement is located in the user FIT file, the default user FIT entries are always loaded *after* the user-specific entries.

#### Note

If you do not place a `#include` statement in a user FIT file, the default user FIT file will *not* be read into memory, and *only* the user-specific entries in the user FIT file will be invoked.

Note that if a user-specific FIT file contains a `#include` statement referencing the WorkSpace On-Demand 1.0 default user FIT file (BB10USDU.FIT), but the user is logged onto a WorkSpace On-Demand 2.0 client, the client operating system will automatically find and load the WorkSpace On-Demand 2.0 default user FIT file instead.

You can use the `#include` statement to incorporate any FIT file, not just the default user FIT file. The FIT file to which the `#include` statement refers must reside in the `\BMLAN\DCDB` directory, and must conform to the standard 8.3 naming convention.

#### Note

There cannot be more than one optional `#include` statement in a single FIT file. This means that you must give careful consideration to how you structure your nested FIT files.

Given that there is a user-specific and a default user FIT file, an administrator has three options for implementing file remapping with FIT files at the user level.

1. Use only the default user FIT:  
No action required; this is how the system functions by default.
2. Use a user-specific FIT file:  
Create a user-specific FIT file, *user\_id.FIT*, for the user ID in the `\IBMLAN\DCDB\USERS\user_id` directory. The client now uses this FIT file for this user ID instead of the default user FIT file.
3. Use a user-specific FIT *and* the default user FIT:  
As in number 2 above, but also use the *#include* statement to append the default user FIT file to the user-specific FIT file.

#### Points to Consider

As supplied, the default user FIT file redirects DAX user files to the user's own subdirectory in the Domain Controller Database. This arrangement ensures that in a multi-server domain, a user's user FIT file will be available, regardless of which server validates the logon and access request, provided that the user logs on to the same domain

However, there are at least two points to consider with this arrangement:

- By placing user data in the DCDB directories, you increase the amount of network traffic due to the DCDB replication process inherent in a multi-server environment. This additional network traffic could negatively impact performance and responsiveness, particularly on heavily loaded networks and/or servers.
- Users will have write access to the directory that defines their user ID.

Alternatives include implementing user home directories or user-specific aliases, or directing user files to a subdirectory in another tree that is *not* within the DCDB directory structure (such as `\IBMLAN\RPLUSER`) where they can be kept separate from essential system files. Information on implementing these techniques can be found in *LAN Server Network Administrator Reference Volume 3: Network Administrator Tasks*, S10H-9680.

### 3.6 FITs in Memory

As already mentioned, the machine FIT is loaded into the client workstation's memory when the WorkSpace On-Demand client operating system is loaded. The machine FIT is maintained in memory at all times until the client workstation is shut down or rebooted.

When a user logs on to the client workstation using the PMLOGON shell, the WorkSpace On-Demand server attempts to locate a user FIT for that user. If a user FIT exists, it is downloaded to the client workstation. If a user FIT does not exist, the WorkSpace On-Demand server downloads the default user FIT to the client. If a user-specific FIT file exists and contains a #include statement for the default user FIT file, the server downloads both files.

When the client receives the user FIT, it merges its contents with those of the machine FIT already in memory. The user FIT entries are placed at the beginning of the in-memory FIT, followed by the machine FIT entries.

When the client starts an application that includes an application FIT file, the contents of the application FIT file are merged into the existing in-memory FIT in a similar way to user FIT entries. The entries from APPPRE.FIT are placed before the user FIT entries, and those from APPPOST.FIT are placed after the user FIT entries.

Figure 27 shows the order of the FIT file entries in the in-memory FIT.

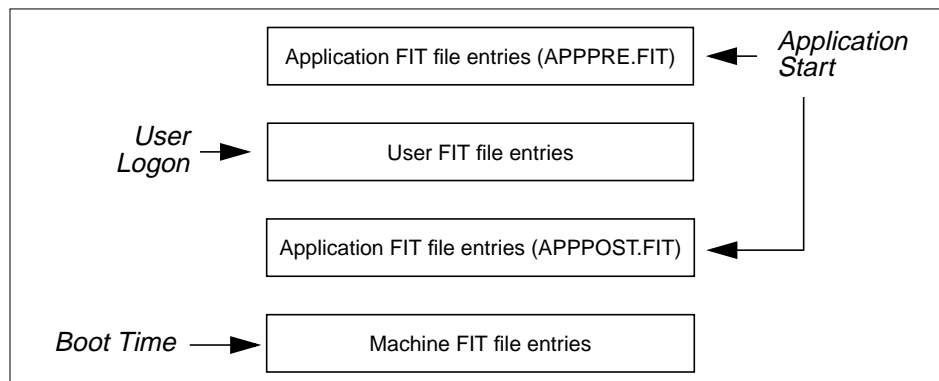


Figure 27. FITs in Memory

The in-memory FIT acts like a table (hence its name). Entries are examined in sequential order until a match is found, at which point the access request is resolved and the I/O is redirected accordingly. This means that if the user FIT

and the machine FIT contain conflicting entries, the user FIT entry will take precedence over the machine FIT entry.

You can display the in-memory copy of a client's FIT at any time using the RDRDEBUG tool, which is included on the CD-ROM that accompanies this redbook. You must copy the SMBTOOL.ZIP file from the CD-ROM to a temporary file on your server, then unpack the zipped file and copy the RDRDEBUG.EXE utility to a location where it can be accessed by the WorkSpace On-Demand client. For example, you might place it into the \BMLAN\RPL\BB20.US\OS2 directory.

To execute the program you will need to set up an OS/2 command prompt (CMD.EXE) as an application on the WorkSpace On-Demand client and give the user access to the command prompt application. While you would not normally grant such access for an end user, it is useful to do so for administrators and testing staff who will be debugging client and application definitions.

Start the OS/2 command prompt session on the WorkSpace On-Demand client and issue the following command:

```
RDRDEBUG /F | MORE
```

The **/F** parameter directs the RDRDEBUG program to display the information from the currently active FIT to the OS/2 window on the WorkSpace On-Demand client. Using **| MORE** allows you to more easily view the many pages of FIT information that will be displayed.

RDRDEBUG will display the FIT entries in both binary character and ASCII format.

Note that WorkSpace On-Demand 2.0 includes an additional tool named RDR\_TEXT.CMD, which resides in the \TOOLS directory on the WorkSpace On-Demand 2.0 CD-ROM. This tool will take the output from RDRDEBUG and display it in an easy to read text format.



## **Part 2. Installing WorkSpace On-Demand**

This part of the redbook discusses the management aspects of WorkSpace On-Demand.

Chapter 4, "WorkSpace On-Demand Planning" on page 103, describes the hardware and software prerequisites for installing WorkSpace On-Demand on an OS/2 Warp Server system along with other planning considerations relating to installing WorkSpace On-Demand on a single server.

Chapter 5, "Installing WorkSpace On-Demand" on page 115, describes how to install WorkSpace On-Demand on an OS/2 Warp Server system using the graphical installation procedure and how to automate the installation using the command line interface, response files and CID techniques.





---

## Chapter 4. WorkSpace On-Demand Planning

This chapter describes the required steps to plan for the installation of WorkSpace On-Demand on your OS/2 Warp Server system. During the course of this chapter, we cover the hardware and software prerequisites as well as some rule-of-thumb planning information.

Note that this chapter describes the planning process for an individual WorkSpace On-Demand server only and is intended to help you plan for a testing or "proof of concept" environment. Chapter 11, "Deploying WorkSpace On-Demand" on page 297, discusses the broader issues of deploying WorkSpace On-Demand in a large-scale production environment.

### Note

Check the README file in the root directory of the WorkSpace On-Demand 2.0 CD-ROM for the latest information on prerequisites and installation requirements. Information contained in the README file may supercede the information contained in this chapter.

For your planning convenience, a WorkSpace On-Demand home page is available to help you plan and manage your WorkSpace On-Demand implementation. This is the best place to find the most up-to-date information, and can be found at: <http://www.software.ibm.com/enetwork/workspace>.

---

### 4.1 Hardware Prerequisites

WorkSpace On-Demand places a number of requirements on both the server and client hardware. The following sections describe the minimum hardware requirements and, where appropriate, make recommendations.

#### 4.1.1 WorkSpace On-Demand Server

The WorkSpace On-Demand server must run OS/2 Warp Server, with the RIPL service installed and active, in order to support WorkSpace On-Demand clients. WorkSpace On-Demand therefore imposes a number of requirements on the server.

##### 4.1.1.1 Processor

We recommend a 90 MHz Pentium processor or equivalent as the *minimum* processor for a WorkSpace On-Demand server. While WorkSpace On-Demand will run on any Intel, or compatible, '486 CPU of at least 66 MHz,

the OS/2 Warp Server base code will exploit features of the Pentium hardware, if it is present.

In cases where multi-function server support is required (for example, your WorkSpace On-Demand server may also act as a file or database server) or where client-loading is heavy, processors with higher clock speeds should be considered. You should consider an SMP machine in cases where multi-function support is required *and* when the supported applications are written to exploit parallel processing, as in DB2/2, for example.

You can find a detailed study of processor performance and WorkSpace On-Demand at:

[www.software.ibm.com/enetwork/workspace/library/whitepapers/white\\_wos\\_capacity.html](http://www.software.ibm.com/enetwork/workspace/library/whitepapers/white_wos_capacity.html)

#### 4.1.1.2 Memory

Running the RIPL, DHCP, PXEPROXY and/or TFTP services within OS/2 Warp Server requires additional memory in the server. Table 11 shows the memory requirements for the RIPL service.

Table 11. Server Memory Requirements - RIPL Service

Client Support	RAM Required	Comments
DOS	+0.7 MB	Support for DOS clients only
OS/2 & WSOD	+0.8 MB	Support for OS/2 and WorkSpace On-Demand clients
All	+1.0 MB	Support for DOS clients, OS/2 clients and WorkSpace On-Demand clients

Note that the requirements given in Table 11 are *in addition* to the requirements for OS/2 Warp Server.

When using the DHCP, PXEPROXY and TFTP services to support DHCP PXE clients, you should use a server machine with at least 32 MB of RAM (16 MB per processor when using OS/2 Warp Server SMP).

#### 4.1.1.3 Disk Space

At the time of writing, installing all the features of WorkSpace On-Demand 2.0 requires approximately 230 MB of disk space. However, the disk space requirement may grow over time, depending upon factors such as:

- The number of client workstations that you plan to boot from each boot server.
- The number and size of any network applications that you will store on the boot server, including client-specific and user-specific configuration files.

- The space that each client workstation will use for swapping if swapping remotely over the network rather than to the client's local hard disk. The space required depends on the size and number of network applications that you plan to run concurrently on each client.

Table 12 shows the disk space requirements for DOS, OS/2 and WorkSpace On-Demand client operating system images.

Table 12. Disk Space Requirements - RIPL Service

Client Support	Disk Space Required	Comments
DOS	+16.7 MB	Includes LAN Support Program, Remote IPL copy of DOS LAN Services, DOS 6.3 and one DOS image
	+0.7 MB	For each additional DOS image
OS/2 & WSOD	+71.2 MB	Includes LAN Support Program, Remote IPL copy of User Profile Management, LAPS, OS/2 LAN Requester, Remote IPL copy of WorkSpace On-Demand client operating system and support for one client workstation
	+0.4 MB	For each additional client workstation
	+1 to 10 MB	Swap space for each client workstation configured to swap remotely

Again, note that the requirements given in Table 12 are *in addition* to the requirements for OS/2 Warp Server.

WorkSpace On-Demand introduces some additional components that you may choose to add when installing WorkSpace On-Demand. Table 13 shows the disk space requirements for these components.

Table 13. Disk Space Requirements - WSOD Features

Component	Disk Space Required
Command Line Interface	1 MB
Remote IPL Services Upgrade	4 MB
WorkSpace On-Demand client operating system image	68 MB
Predefined machine classes	4 MB
Printer driver support	12 MB
<b>TOTAL DISK SPACE REQUIRED</b>	<b>89 MB</b>

WorkSpace On-Demand also includes a number of client operating system components and applications that you may selectively install within the client operating system image on the server. Table 14 shows the disk space requirements for these components and applications.

Table 14. Disk Space Requirements - WSOD Client Components

Component	Disk Space Required
Java 1.1	23 MB
DOS Support	2 MB
WIN-OS2 Support	32 MB
REXX	2 MB
Multimedia Support	11 MB
Optional System Utilities	1 MB
TCP/IP (Required for DHCP boot)	43 MB
PCOMM Entry	13 MB
Netscape Navigator	12MB
<b>TOTAL DISK SPACE REQUIRED</b>	<b>139 MB</b>

**Note**

The figures given in Table 11 and Table 12 are not official and reflect the testing performed at the ITSO Austin Center during the preparation of this redbook, using pre-release versions of WorkSpace On-Demand 2.0. If you install each component separately, your findings may be slightly different from those listed here.

#### 4.1.1.4 Caching Requirements

At the time of writing, the WorkSpace On-Demand client operating system occupies approximately 14 MB of RAM. You should plan to add 16 MB of additional RAM to the WorkSpace On-Demand server to ensure that client requests are serviced from cache. Please note that we recommend using the HPFS386 file system configured with enough cache space to service at least 95 percent of all disk read requests from the disk cache.

## 4.1.2 Client Workstation

### 4.1.2.1 Processor

The WorkSpace On-Demand client operating system will run on any Intel-based system that is capable of running OS/2 Warp 4. However, for performance reasons, we recommend that you run WorkSpace On-Demand on a machine with an Intel 486 33 MHz or faster processor.

You should consider the processor requirements of your applications, as well as those of the WorkSpace On-Demand client operating system, when selecting hardware for your client workstations. Although applications are stored on the server, they run on the client using its processor and memory. If you plan to run CPU-intensive applications, you may need to use a faster processor on your client workstations.

### 4.1.2.2 Memory

The WorkSpace On-Demand operating system requires approximately 14 MB of RAM to contain its memory-resident working set. The minimum memory configuration supported by IBM for a WorkSpace On-Demand client is 16 MB. We therefore recommend that you install at least 16 MB of RAM in each client workstation.

However, the exact amount of RAM required in your client workstations will depend on the number and size of the applications that you intend to run concurrently in your clients. You will need to test your application suite with the WorkSpace On-Demand client operating system to determine the optimal RAM requirement for your installation.

Your choice of whether to use local or remote swapping in your clients will also affect the RAM requirement. We recommend that you use local swapping in order to reduce network traffic and improve performance, but if you decide to use remote swapping, you will need to add approximately 4 MB of RAM to your clients, making a total requirement of 20 MB, plus whatever RAM you require for your applications.

### 4.1.2.3 Disk Space

It is possible to use diskless workstations as WorkSpace On-Demand clients. However, we recommend that you use a client's local hard disk for swapping if the client hardware includes a hard disk.

The amount of disk space required for swapping is dependent upon the amount of RAM installed in your client workstations and upon the requirements of your applications. You must test your application suite with the WorkSpace On-Demand operating system to determine the amount of

disk space required for swapping. However, given the relatively large disk drives typically present in most PC hardware, and bearing in mind that the local disk drive is used only for swapping, it is unlikely that disk space on the client will become a constraining factor in a WorkSpace On-Demand environment.

#### 4.1.2.4 Predefined Clients

WorkSpace On-Demand includes predefined hardware configuration information for a number of different types of client workstation, as shown in Table 15.

Table 15. Predefined Machine Classes in WorkSpace On-Demand 2.0

Machine Class	Model / Bus	Video Type
GENGRADD	Generic ISA/PCI GRADD. Most PCs with an ISA / PCI bus and SVGA display adapter that is VESA 1.2 Compliant.	VESA Compliant
IBM300GL	IBM Personal Computer 300GL model 6282-38U	Cirrus Logic 5436/5446
IBM300G2	IBM Personal Computer 300GL model 6272-880	Cirrus Logic 5436/5446
IBM350	BM Personal Computer 350 model 6586-57H	S3 TRIO 64/64V+
IBM730	IBM Personal Computer 730 model 6877-KAZ	S3 TRIO 64/64V+
IBM750	IBM Personal Computer 750 model 6887-86B	S3 TRIO 64/64V+
ISAVGA	Any PC with an ISA bus and VGA display adapter that is supported by OS/2 Warp Version 4	Generic VGA
MCAVGA	IBM PCs (486 +) With Micro Channel BUS and standard VGA with SCSI/ESDI support.	VGA
MCAXGA	BM PCs (486 +) With Micro Channel BUS and XGA video with SCSI/ESDI support.	XGA

These sets of hardware configuration information are known as *machine classes*. You can use the standard machine classes supplied with WorkSpace On-Demand, modify these machine classes to meet your specific requirements, or create new machine classes to support different client machines and hardware devices. See Chapter 10, "Supporting Additional Hardware" on page 251, for a detailed discussion of how to create and modify machine classes.

You will note from Table 15 that WorkSpace On-Demand now supports GRADD. OS/2 Warp 4 introduced a new graphics adapter device driver (GRADD) architecture that makes it easier to support new hardware as it becomes available. Accelerated GRADD drivers provide enhanced performance for OS/2 Presentation Manager applications.

The GRADD device drivers provide seamless support for accelerated/unaccelerated display graphics and provide enhanced color depth and resolution. GRADD drivers conform to the OS/2 32-bit flat memory model and are designed to function as 32-bit Presentation Manager graphics display drivers under the OS/2 32-bit graphics engine. You can obtain more information about GRADD at

<http://service.software.ibm.com/os2ddpak/html/gradd>

---

## 4.2 Software Prerequisites

All WorkSpace On-Demand software is installed on the WorkSpace On-Demand server. The only exception to this statement is the WorkSpace On-Demand Administration Client component, which can be installed either on the WorkSpace On-Demand server or on a separate "fat client" workstation. In either case, there is prerequisite software that must be in place prior to installing WorkSpace On-Demand software.

### Installing Over WorkSpace On-Demand 1.0

You can install WorkSpace On-Demand 2.0 on an existing WorkSpace On-Demand 1.0 system. However, you should note that your existing clients will still use the WorkSpace On-Demand 1.0 client operating system. If you wish to migrate your clients to WorkSpace On-Demand 2.0, you must delete and redefine them.

### 4.2.1 WorkSpace On-Demand Server

You must install IBM OS/2 Warp Server, OS/2 Warp Server Advanced, or OS/2 Warp Server Advanced - SMP on the WorkSpace On-Demand server prior to installing WorkSpace On-Demand itself.

#### Note

Except where explicitly stated, any reference to OS/2 Warp Server in this redbook refers to *all* versions of the OS/2 Warp Server product.

When installing OS/2 Warp Server, you must install the following options:

- Remote IPL Support  
This is a required component and must be installed.
- File and Print Sharing  
This is a required component and must be installed.
- LAN Server Administration Tool  
Required if you plan to perform administration tasks on the server. If you will perform all administration from a client, you do not require the LAN Server Administration Tool.

If you are installing WorkSpace On-Demand on OS/2 Warp Server or OS/2 Warp Server Advanced, you must apply FixPak 22 or later prior to installing WorkSpace On-Demand. This FixPak contains base OS/2 fixes. The WorkSpace On-Demand installation process will apply required fixes for other components, such as LAN Server and MPTS.

**Note**

FixPak 22 is *not* required for OS/2 Warp Server SMP. But if you need year 2000 readiness, you have to apply—at least—the FixPak 32. The upcoming OS/2 Warp Server for e-Business does not need any fix to be Year 2000 compliant.

OS/2 Warp Server Advanced and OS/2 Warp Server Advanced SMP are the preferred platforms for a WorkSpace On-Demand server since they support the HPFS386 file system and therefore provide greater file system throughput. OS/2 Warp Server is not recommended for use with WorkSpace On-Demand, except under controlled and limited circumstances.

#### 4.2.2 Administration Client

The following software must be installed on the machine used for the WorkSpace On-Demand Administration Client:

- IBM OS/2 Warp Version 4
- File and Print Client. This is a required component.
- LAN Server Administration Tools



### 4.2.3 Workspace On-Demand Client Workstation

There is no user-installed software on the client workstation. However, the client machine must have a BIOS that allows it to boot from the network.

#### Hint

If your client's BIOS does not allow it to boot from the network, you can try the following technique:

- Use FDISK to remove any startable partitions on the local hard disk.
- Ensure that you do not have any other bootable devices such as diskette drives or CD-ROM drivers.
- Enable the Boot PROM on the network adapter.

The Boot PROM itself should now start the boot process from the network.

### 4.2.4 Year 2000 Readiness

All Workspace On-Demand components are Year 2000 enabled. However, the platforms on which you install Workspace On-Demand may require fixes in order to provide the necessary support:

- On the Workspace On-Demand Server, you must apply FixPak 32 for OS/2 Warp V3.0, prior to installing the Workspace On-Demand Manager.
- On an administration client, you must apply FixPak 6 or later to the base OS/2 Warp 4 operating system.

Note that the Workspace On-Demand client operating system provides Year 2000 support without any additional fixes.

For more information about the relationship between OS/2 Warp Family and Year 2000 and a step-by-step guide to be year 2000 compliant, check <http://www.software.ibm.com/os/warp/solutions/and/y2000/year2000.html>.

### 4.2.5 Euro-Currency Support

Some national language versions of Workspace On-Demand 2.0 include Euro-currency support. These versions incorporate the updates included in FixPak 6 for OS/2 Warp 4, which provide this support. The complete list of national language versions for Workspace On-Demand 2.0 is shown in Table 16 along with the FixPak level of the client operating system. Note that the Hebrew national language version does not provide Euro currency support.

Table 16. National Language Versions and FixPak Levels

Language	FixPak (w/ Euro Support)	FixPak (w/o Euro Support)
Arabic	XRAM006	
Chinese (Simplified)		FXM0505
Chinese (Traditional)		FXT0505
Czech		XRZM005
Danish	XRDM006	
Dutch	XRHM006	
English (US)	XR_M006	
English (Worldwide)	XRUM006	
Finnish	XRLM006	
French (Canadian)	XRCM006	
French (France)	XRFM006	
German	XRGM006	
Greek		XRKM005
Hebrew	OS/2 Warp 4 Hebrew refresh (FixPak 5+)	
Hungarian		XRYM005
Italian	XRIM006	
Japanese		FXJ0505
Norwegian	XRNM006	
Polish		XROM005
Portuguese (Brazilian)	XRBM006	
Portuguese (Portugal)	XRPM006	
Russian		XRRM005
Spanish	XRSM006	
Swedish	XRWM006	
Thai		XRVM005
Turkish		XR1M005

### 4.3 Supported Network Adapters

IBM is continually testing and adding support for new network adapters under WorkSpace On-Demand. For this reason, we do not provide a list of supported adapters in this redbook. For the latest list of supported adapters, see <http://www.software.ibm.com/enetwork/workspace/about/adapters.html>

### 4.4 Network Resource Requirements

The Remote IPL service requires additional network resources. Some of the configuration parameters in the WorkSpace On-Demand server's IBMLAN.INI and PROTOCOL.INI files must be changed. Table 17 on page 113 shows the parameters that may require modification.

Table 17. Network Resource Requirements

Parameter	Value	Description
Server Section in IBMLAN.INI		
NUMREQBUF	4 x n	Maximum value is 408
MAXOPENS	n x (65 + max. files opened per client)	This number includes any application code, DLL and data files opened by the client. Maximum value is 8000.  Note: This parameter is not required for a server running the HPFS386 file system.
MAXSESSOPENS	100 + max. files opened per client	This number includes any application code, DLL and data files opened by the client.  Note: This parameter is not required for a server running the HPFS386 file system.
Remote Boot Section in IBMLAN.INI		
MAXTHREADS	6	For large installations, increasing this parameter may decrease performance. You should specify a value no greater than 6.
Protocol Section in PROTOCOL.INI		
NCBS	4 x n	
SESSIONS	m	

In the values given in Table 17:

- $n$  is the number of WorkSpace On-Demand clients supported by this server.
- $m$  is the total number of DOS and WorkSpace On-Demand clients supported by this server.

Note that these guidelines take into account only the resource requirements necessary to support remote IPL. Add the resource requirements for Remote IPL to any other requirements, such as fault tolerance, local security, communication with local IPL requesters and other servers, and so forth.

For more information about these parameters and their interdependency on other parameters, refer to *LAN Server Network Administrator Reference Volume 2: Performance Tuning*, S10H-9681.

**Note**

A single WorkSpace On-Demand client logged on as GUEST with no applications configured for its desktop requires a single connection with four sessions and 81 open files. This information is based on a beta version of the product.

## Chapter 5. Installing WorkSpace On-Demand

This chapter describes the installation of WorkSpace On-Demand Manager and WorkSpace On-Demand Administration Client on machines that meet the hardware and software prerequisites described in Section 4.2, "Software Prerequisites" on page 109.

### 5.1 Before You Install WorkSpace On-Demand

The README file located in the root directory of the WorkSpace On-Demand installation CD-ROM contains the latest information regarding the installation process. Unless otherwise noted in this chapter, consider the information in the README file to override the guidance contained in this chapter. You should read the README file before starting your installation process.

The order in which you install OS/2 Warp Server and WorkSpace On-Demand is important. Following is the order in which products should be installed on a new server.

1. Install OS/2 Warp Server, including RIPL support.
2. Install LCCM (if you require LCCM in your installation).
3. Install FixPaks for OS/2 Warp (see Section 4.2, "Software Prerequisites" on page 109).
4. Install WorkSpace On-Demand.

#### FixPaks

WorkSpace On-Demand includes a number of FixPaks that are applied as part of the WorkSpace On-Demand installation. These are:

- LAN FixPaks:
  - IPx8409 - for OS/2 Warp Version 4
  - IPx8528 - for OS/2 Warp Server
- MPTS FixPaks
  - WRx8610 - OS/2 Warp Version 4 and OS/2 Warp Server

For your convenience, FixPak 32 for OS/2 Warp Server is contained on the WorkSpace On-Demand CD-ROM, but is not installed as part of the WorkSpace On-Demand installation process.

If you are installing WorkSpace On-Demand on an existing server, you may be able to omit one or more of the steps given above.

#### **Back Up Your System**

We strongly recommend that you back up your server before installing any WorkSpace On-Demand Manager components. The only way to return the server to the state it was in before WorkSpace On-Demand was installed is to restore the server from a backup.

---

## **5.2 OS/2 Warp Server Considerations**

If you intend to install OS/2 Warp Server and WorkSpace On-Demand 2.0 onto an entirely new server, you should consider the following points in order to avoid problems during installation.

Before installing OS/2 Warp Server and WorkSpace On-Demand, you should review Chapter 4, "WorkSpace On-Demand Planning" on page 103, if you have not already done so.

### **5.2.1 Choosing a Version of OS/2 Warp Server**

There are three versions of OS/2 Warp Server currently available from IBM:

- OS/2 Warp Server
- OS/2 Warp Server Advanced
- OS/2 Warp Server Advanced with Symmetric Multi-Processing (SMP)

We recommend that you use OS/2 Warp Server Advanced or OS/2 Warp Server Advanced with SMP as a basis for your WorkSpace On-Demand 2.0 server. Both of these products include the network-optimized 32-bit High Performance File System (HPFS386), which will provide enhanced file system throughput for your server, improving boot-time performance at the client.

### **5.2.2 Installing OS/2 Warp Server**

When installing OS/2 Warp Server, there are a number of points that you should consider. These are explained in the following sections.

### 5.2.2.1 Formatting Your Hard Disk

If you are installing OS/2 Warp Server onto a hard disk that already contains an operating system, particularly an existing installation of OS/2 Warp Server, we strongly recommend that you perform a "long" format on the hard disk prior to installation. If you allow the OS/2 Warp Server installation program to format the hard disk, it performs a "short" format which, although less time-consuming, does not always erase existing extended attributes and can cause problems later in the installation process.

To perform a long format, follow these steps:

1. When the OS/2 Warp Server installation program prompts you to accept or change the drive on which it will install OS/2 Warp Server, press **F3** to open a command line.
2. Change drives to the drive letter that corresponds to your CD-ROM drive.
3. Type `CD \OS2IMAGE\DISK_2`
4. Type `FORMAT d: /FS:HPFS /L`, where *d:* is the drive on which you wish to install OS/2 Warp Server. Note that the disk will take a significant period of time to format.
5. When the format completes, type `EXIT` to return to the OS/2 Warp Server installation program.

You may now proceed with the installation process.

### 5.2.2.2 Choosing Features

There are many optional features within OS/2 Warp Server. However, many of these features are not required to support WorkSpace On-Demand, and by omitting them from your installation, you can save both installation time and disk space on your server. You should review each of the components of OS/2 Warp Server with regard to your own requirements, and omit those that you do not require.

Note, however, that you *must* install the **Remote Boot Service for OS/2 Workstations** feature in order to support WorkSpace On-Demand clients. If you wish to provide remote boot support for DOS clients, you must install the **Remote Boot Service for DOS Workstations** feature.

By default, these features are not selected. You must select them by pressing the **More...** button to the right of the **File and Print Sharing** option on the ??? panel, and selecting the **Remote Boot Service for OS/2 Workstations** and **Remote Boot Service for DOS Workstations** check boxes on the File and Print Sharing Services panel, as shown in Figure 28 on page 118.

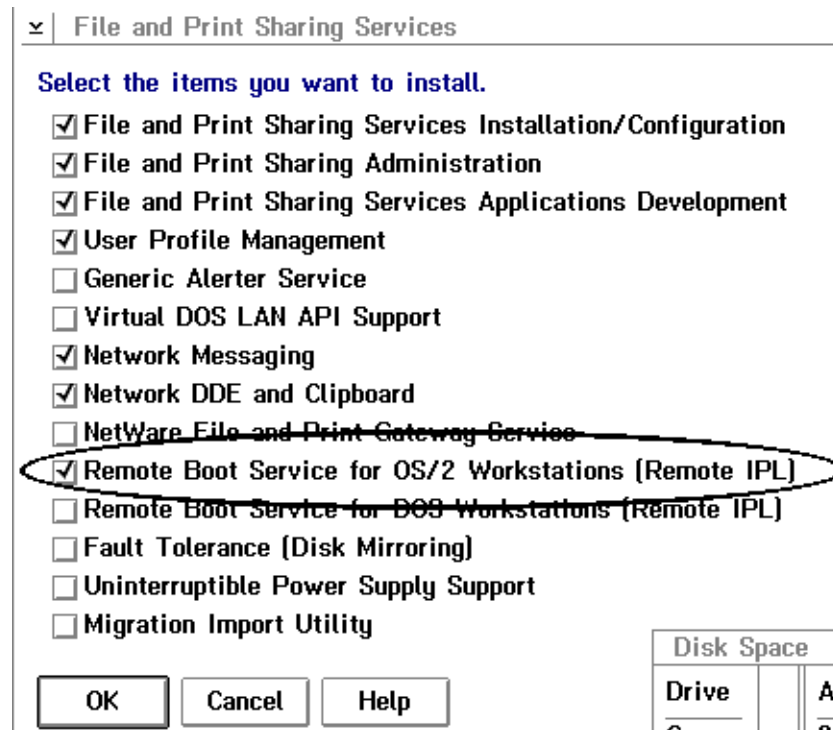


Figure 28. Selecting Remote Boot Service for OS/2.

### 5.2.2.3 HPFS386 Cache Size

In systems with more than 16 MB of RAM, the OS/2 Warp Server installation program sets the default HPFS386 cache size to 60 percent of the installed RAM in the system. In some cases, this settings may be satisfactory, but in most cases, it reserves an unacceptably large portion of the system's memory for caching.

You can improve the performance of your server by editing the HPFS386.INI file, which resides in the \IBM386FS directory on your server's boot drive, immediately after installation. Locate the `cachesize` parameter in the file, and reduce the cache size to a more acceptable level. We recommend that you use a minimum cache size of 16 MB, but you may need to increase the cache size depending on the number and size of the device drivers, middleware and application files that you add to the client operating system image for your installation.



### 5.2.3 Applying FixPaks to OS/2 Warp Server

WorkSpace On-Demand 2.0 includes FixPak 32 for the OS/2 Warp Server base code. This FixPak is Year 2000 compliant and provides the necessary support for WorkSpace On-Demand. However, you may find that subsequent FixPaks are available for OS/2 Warp Server. We recommend that you install the latest available FixPak for OS/2 Warp Server.

You can find the latest available FixPaks for OS/2 Warp Server and other OS/2 Warp releases at <http://service5.boulder.ibm.com/pspfixpk.nsf>.

---

## 5.3 Installing WorkSpace On-Demand on Your Server

You can install WorkSpace On-Demand by one of two methods:

- An attended installation uses a graphical user interface (GUI), where you will be prompted for information and options during the installation.
- An unattended installation uses the configuration, installation, and distribution (CID) architecture. After invoking an `install` command, a response file provides keywords and values so that user intervention is not required.

You can choose to install all of the WorkSpace On-Demand components or a smaller subset of components as part of a selective installation. In a selective installation, you can choose the specific WorkSpace On-Demand components to be installed on the server.

### 5.3.1 Attended Installation

You only need to install the WorkSpace On-Demand Manager code on a server from which the WorkSpace On-Demand clients are going to be booted.

**Attention**

If you install OS/2 Warp Server components after installing WorkSpace On-Demand, you must reinstall WorkSpace On-Demand because both products share some common code. WorkSpace On-Demand will be overwritten by older OS/2 Warp Server code, and there may be unpredictable results.

### 5.3.1.1 Phase One

The WorkSpace On-Demand CD-ROM includes Feature Install and all the necessary FixPaks. These components are required to perform the phase one portion of the installation and are automatically installed as part of the WorkSpace On-Demand installation if not already present and at the current levels.

#### Feature Install

IBM OS/2 Feature Install is an installation framework that is used to install some IBM Software Choice features. Feature Install provides OS/2 applications with a common set of installation functions necessary for most installation programs.

In addition to enhancing the consistency of the user interface among installation programs, Feature Install supports important administrative installation tasks including the distributed installation of features through a response file and the ability to install a feature onto a boot server. To find out more about Feature Install, check out the IBM Software Choice Web page at <http://service.boulder.ibm.com/asd-bin/doc/>

Before you begin the installation, you should stop all network services. You can do this by issuing the following command from a command line:

```
NET STOP REQ
```

If you forget to stop network services, the installation program will stop them automatically for you.

From an OS/2 command line of an OS/2 window or full screen session, enter the drive letter where the WorkSpace On-Demand CD is located, type `INSTALL` and press **Enter**. For example, if the WorkSpace On-Demand CD-ROM on your system is in drive G, enter `G: INSTALL`.

The installation program will display an IBM WorkSpace On-Demand installation screen. Click the forward arrow at the bottom of the screen, and the installation program will show the Ready to Install Upgrades screen, as shown in Figure 29 on page 121.

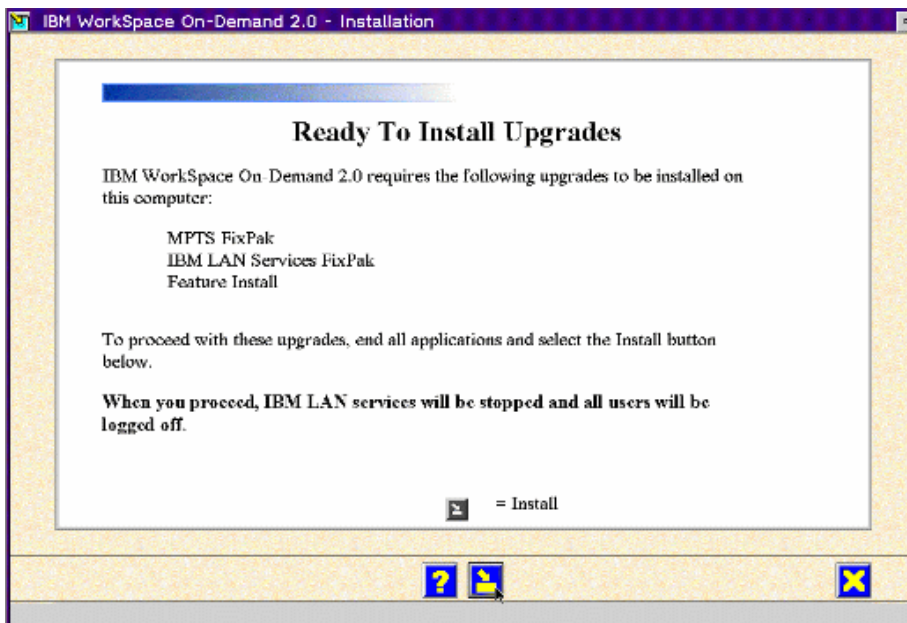


Figure 29. Ready to Install Upgrades

Click on the **Install** button to begin to copy files to your hard drive.

When the files have been copied to your hard drive, the installation program will prompt you to restart your server. Click on the **Restart** button, and the server will shut down and reboot automatically.

This completes the first phase of the installation.

### 5.3.1.2 Phase Two

After rebooting the machine, the second phase of the installation starts automatically. This phase uses Feature Install (installed during the first phase of the installation) to install the WorkSpace On-Demand components that you select.

Feature Install determines the options it needs to install by reading response files for Java and TCP/IP and calculates the disk space required to install WorkSpace On-Demand components. It then presents you with the Select Components screen, as shown in Figure 30 on page 122.

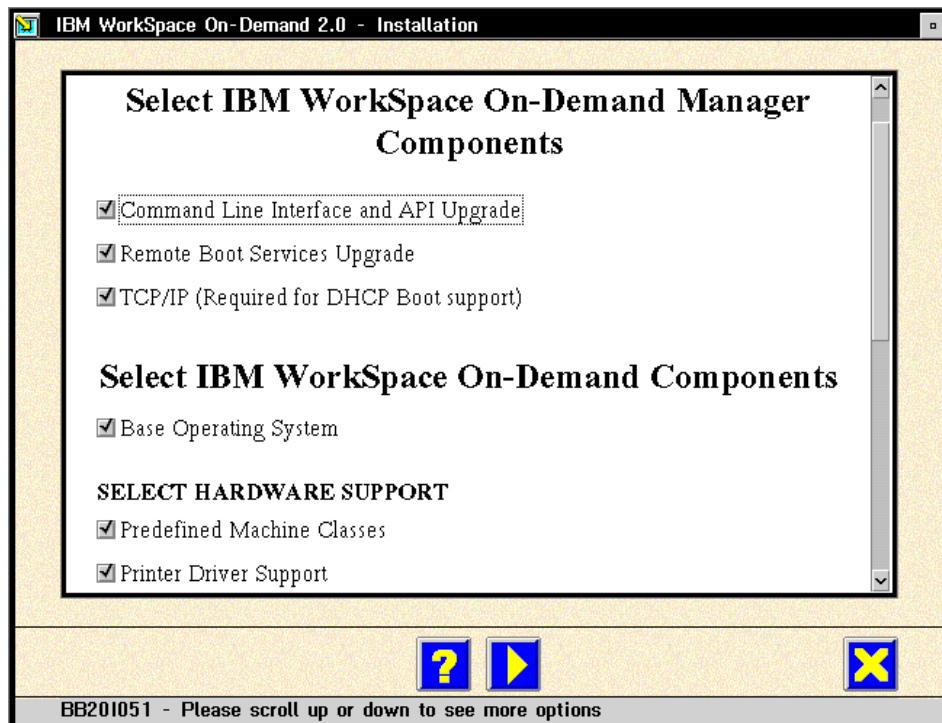


Figure 30. Select WorkSpace On-Demand Components (1)

Using the Select Components screen, select the features you need on your WorkSpace On-Demand server. If this is the first time you are installing WorkSpace On-Demand 2.0, we recommend that you install all of the components. Later, you can customize the server environment to suit your requirements and save disk space.

#### **WorkSpace On-Demand Manager Components**

WorkSpace On-Demand Manager is the set of utilities that run on your OS/2 Warp Server system and allow you to manage clients and applications.

- **Command Line Interface and API Upgrade**

Select this option to upgrade OS/2 Warp Server's GUI and command line interface to support WorkSpace On-Demand commands and functions.

**Note**

The name of this option is somewhat misleading. Even if you will only perform administrative tasks from the OS/2 Warp Server GUI, you *must* select this option in order to upgrade the LAN Server Administration Tool to support WorkSpace On-Demand 2.0 functions.

- **Remote Boot Services Upgrade**

Select this option to upgrade the REMOTEBOOT service on your server.

To select this option, you must select the **Command Line Interface and API Upgrade** option. You must also have installed the **Remote Boot Service for OS/2 Workstations** option when you installed OS/2 Warp Server. If the **Remote Boot Services Upgrade** option is not selectable, check that you have performed both of these tasks.

- **TCP/IP**

Select this option to upgrade the TCP/IP services on your OS/2 Warp Server system. You must select this option if you wish to use the DHCP PXE boot mechanism since upgrades are required to provide this support.

If you select this option, the installation program will install the TCP/IP base applications, DHCP/DDNS services and Network File System (NFS) support on the drive where your TCP/IP services are currently installed or on the server's boot drive if TCP/IP is not currently installed.

The installation program will also install the Java 1.1.6 Virtual Machine on the drive where JAVa 1.1.x is currently installed, or on the server's boot drive if Java 1.1.x is not currently installed.

**Note**

You can install TCP/IP and Java on a drive other than the server's boot drive, using one of two methods:

- Install TCP/IP and/or Java under OS/2 Warp Server, specifying a drive other than the servers' boot drive, before installing WorkSpace On-Demand.
- Redirect the installation under WorkSpace On-Demand by editing the TCPCID.RSP and/or JAVACID files, which are located in the \INSTALL directory on the WorkSpace On-Demand 2.0 CD-ROM. Naturally, you will need to copy these files to the \OS2\INSTALL directory on your server's hard disk before you can modify them.

### **WorkSpace On-Demand Components**

WorkSpace On-Demand is the term used in the installation program to refer to the WorkSpace On-Demand client operating system.

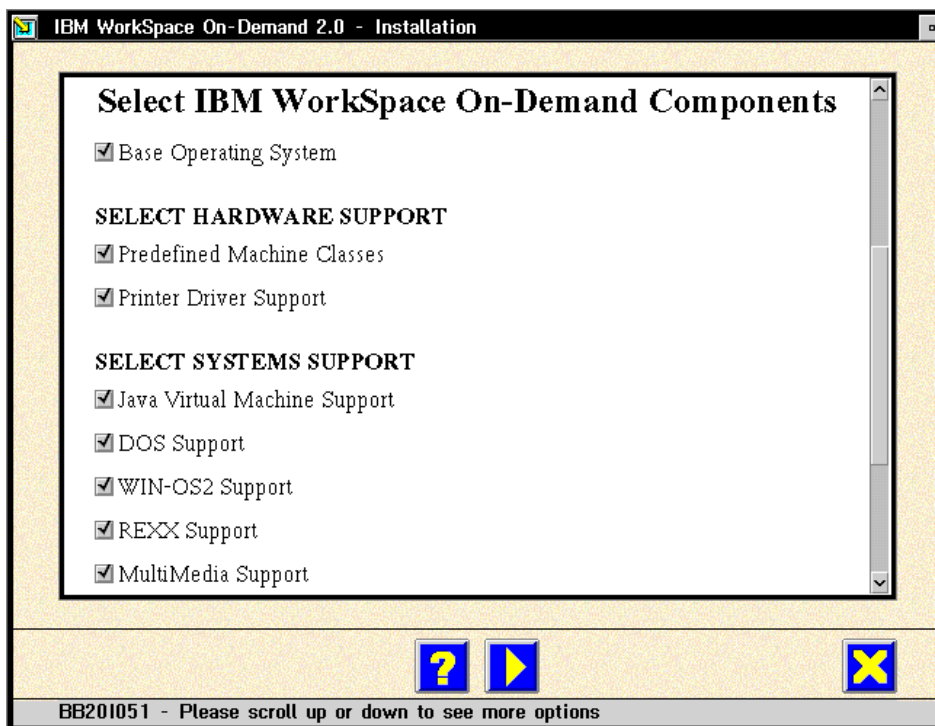


Figure 31. Select WorkSpace On-Demand Components (2)

The installation program allows you to customize the client operating system environment with a number of options.

- **Base Operating System**

Select this option to install the client operating system on your server. You must select this option if you wish to boot clients from this server.

- **Predefined Machine Classes**

Select this option to install hardware support for a specific set of predefined clients (see Section 4.1.2.4, "Predefined Clients" on page 108, for a list of the hardware supported by WorkSpace On-Demand 2.0).

If you select this option, you should ensure that you select the Remote Boot Services Upgrade option. The installation program sets up a number of file structures when installing the Remote Boot Services Upgrade that are required to support the predefined machine classes.

- **Printer Driver Support**

Select this option to install all of the supported printer drivers into the \IBMLAN\RPL\BB20.US\OS2\DRIVERS directory on your server. The drivers are grouped according to the type of printer they support (such as PostScript). If you require support for only certain types of printer, you can remove the unwanted types by manually deleting the drivers after installation.

Note that the Select Components screen contains a scroll bar. You will need to scroll down to view many of the components, as shown in Figure 32 on page 125.

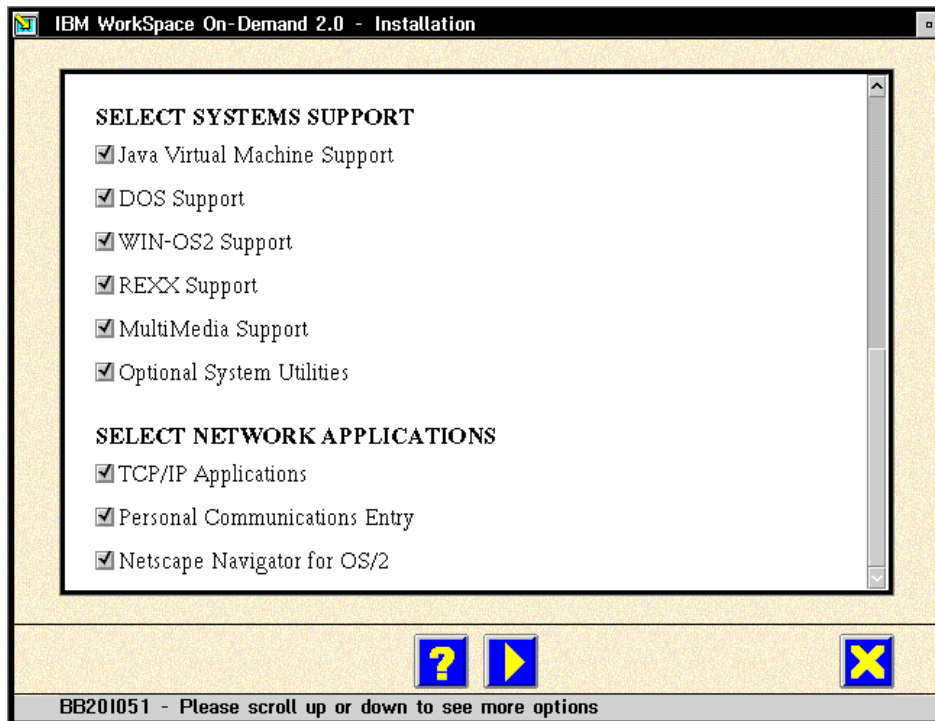


Figure 32. Select WorkSpace On-Demand Components (3)

- **Java Virtual Machine Support**

Select this option if you wish to run Java applications and applets on your clients.

- **DOS Support**

Select this option if you wish to run DOS applications on your clients.

- **WIN-OS2 Support**

Select this option if you wish to run 16-bit Windows 3.x applications on your clients. You must also select the DOS Support option if you wish to run Windows 3.x applications.

- **REXX Support**

Select this option if you wish to run REXX programs on your clients.

- **Multimedia Software Support**

Select this option if you wish to provide support for multimedia capabilities, such as Ultimotion video and sound, on your clients.

- **Optional System Utilities**

Select this option to install a number of command line utilities within your clients' boot image on the server. The system utilities included are:

ANSI	LABEL
ATTRIB	MODE
BLDLEVEL	MORE
BOOT	PSTAT
DISKCOPY	SETBOOT
FDISK	TREE
FIND	UNDELETE
FORMAT	XCOPY

- **TCP/IP Applications**

Select this option to install basic TCP/IP applications, such as FTP, Telnet and so on, within your clients' boot image on the server. Note that the TCP/IP protocol support is included within the Base Operating System option. You only need to select the **TCP/IP Applications** option if you explicitly need to run these applications on your clients.

- **Personal Communications Entry**

Select this option to provide entry-level, TCP/IP-based 3270/5250 terminal emulation on your clients.

- **Netscape Navigator**

Select this option to provide Web browser support on your clients.



After selecting the components you wish to install, start the installation process by selecting the **Install** button. The installation program will now copy the necessary files to the server's hard disk.

**Note**

If there is not enough space on the target drive(s) to install the selected components, the installation program displays the Insufficient Disk Space screen. You must select fewer options, free some space on the target drive(s), or redirect the installation of TCP/IP or Java to a different drive by editing their response files before continuing with the installation procedure.

**5.3.1.3 Registration**

When the second phase of the attended installation completes, the installation program will start the Axtive Registration Tool (ART). This tool allows you to register your WorkSpace On-Demand installation with IBM. If you wish to complete your registration at this time, you should follow the instructions given on the ART screens to fill in the necessary information. If you do not wish to complete your registration, you can select the **Cancel** button to exit from ART and complete the installation process.

Upon completion, the installation program will prompt you to shutdown and reboot your server. Note, however, that you must complete some additional steps before you can define and boot clients from your server. These steps are described in Section 5.3.1.4, "Post Installation Procedures" on page 127.

**If You Have Problems**

If you encounter any problems during the installation, check the WPINSTALL.LOG and BB20ERR.LOG files located in the \OS2\INSTALL directory on your server's boot drive. Try to rerun the `INSTALL` command from the WorkSpace On-Demand 2.0 CD-ROM. The installation program is often able to detect where it stopped the last time, and it will resume the installation from the previous point of failure.

**5.3.1.4 Post Installation Procedures**

After you install WorkSpace On-Demand 2.0 on your server, you must run the RIPL post-install utility (GETRPL) to initialize the server's RPL service and directory structures before you can define clients.

GETRPL migrates existing RIPL files, creates and updates system INI files, and moves files to the appropriate directories on the server. You must run the GETRPL utility on the server *after* you install WorkSpace On-Demand 2.0.

**Note**

Before running GETRPL, you must ensure:

1. The SERVER service is started.
2. The RPL service 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 currently started on your server by issuing the `NET START` command (without specifying a service name) from a command line.

If any of the above conditions is not met, the GETRPL utility will fail. In this case, you must terminate GETRPL, rectify the problem and then restart GETRPL.

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

```
GETRPL /I /O:BB20.cc
```

and press **Enter**.

- The `/I` parameter tells the server to build a new OS2.INI file for the boot code.
- The `cc` refers to the national language code of the WorkSpace On-Demand version you are installing.

For more information about GETRPL and its optional parameters, see the online *WorkSpace On-Demand Administrator's Guide* that is provided on the WorkSpace-Demand CD-ROM.

### 5.3.2 Unattended Installation

You can install WorkSpace On-Demand in an unattended mode using a response file to provide the necessary installation choices. You can carry out an unattended installation in one of two ways:

- By starting the install program from an OS/2 command prompt on your server and specifying the necessary parameters.

- By running a REXX script to distribute the necessary files and start the installation. In this way, you can install WorkSpace On-Demand remotely using CID techniques, without actually being present at the server when you start the install.

Both of these methods require a response file that contains the installation choices that you would otherwise make using the GUI in an attended installation. WorkSpace On-Demand includes a sample response file named BBCID.RSP, which resides in the \INSTALL directory on the WorkSpace On-Demand CD-ROM. You can copy this response file and customize it to meet your own requirements.

### 5.3.2.1 Using the Command Line Interface

You can start an unattended installation from an OS/2 command prompt by issuing the `INSTALL` command, with the following parameters:

```
INSTALL /B:boot_drive /L1:path_log_file /L2:path_log_hst  
/R:path_response_file /R2:path_user_response_file
```

For example:

```
INSTALL /B:C /L1:D:\LOGS\WSODR2.LOG /L2:D:\LOGS\WSODR2.HST  
/R:D:\UTIL\NCINSTALL.RSP /R2:D:\UTIL\BBCID.RSP
```

The above example installs WorkSpace On-Demand 2.0 onto the server's C-drive, writes its log files to the D:\LOGS directory, and instructs the installation program to search for Netscape Navigator or Communicator in the D:\NETSCAPE directory. The installation options are contained in the following response files:

- NCINSTALL.RSP contains the options related to Feature Install for phase one of the installation.
- BBCID.RSP contains the options related directly to WorkSpace On-Demand for phase two of the installation.

In the example above, both of these response files reside in the D:\UTIL directory.

Note that there are two additional response files that are not mentioned explicitly in the `INSTALL` command:

- TCPCID.RSP controls the installation of TCP/IP services
- JAVACID.RSP controls the installation of Java 1.1.6

These response files reside in the \OS2\INSTALL directory on the server's boot drive, or in the same directory on the CD-ROM.

### 5.3.2.2 Using CID

You can install WorkSpace On-Demand remotely over a network using CID techniques. In this case, you will require a REXX script to run the installation after the files are distributed to the server. Note that you will require approximately 250 MB of disk space on the distribution server for the entire WorkSpace On-Demand directory tree and its files.

#### Note

Do not try to install WorkSpace On-Demand into a directory other than \IBMLAN\RPL\BB20.US. The installation will work correctly, but WorkSpace On-Demand will not work because this path is hard-coded in the product.

Since WorkSpace On-Demand uses Feature Install to carry out the installation, there are actually a number of response files:

- The first response file provides the information for phase one of the installation—that is, installing Feature Install itself. The default name for this file is NCINSTAL.RSP.
- The second response file provides the installation options within WorkSpace On-Demand and is read by Feature Install. The default name for this file is BBCID.RSP.
- The third response file contains the installation options for TCP/IP services, and is named TCPCID.RSP.
- The fourth response file contains the installation options for Java, and is named JAVACID.RSP.

### 5.3.2.3 Post-Installation Procedures

After the installation is finished, the machine must reboot again. In a CID environment, this will happen automatically because the installation program returns a code of xFE00, indicating a successful program termination with a required reboot. After this reboot, the GETRPL utility must be run to complete the WorkSpace On-Demand environment. To run this utility unattended, you may call the program with the /L parameter for the log file.

```
GETRPL /L:d:\WSOD\GETRPL.LOG /O:BB20.US.
```

You must be logged on as an administrator to run this command.

---

## 5.4 Installing the WorkSpace On-Demand Administration Client

Just as OS/2 Warp Server allows you to perform administration tasks from a client on the network, WorkSpace On-Demand also allows you to manage your servers and clients from an OS/2 Warp 4 client. To do this, however, you must install the WorkSpace On-Demand Administration Client component on your OS/2 Warp 4 system.

### 5.4.1 Attended Installation

You can manage an OS/2 Warp Server domain from an OS/2 Warp Version 4 client by using the LAN Server Administration tool. This is a selectable option during the installation of the OS/2 Warp Version 4 client. An administrator can remotely manage any domain (or servers in that domain) without actually sitting at the domain controller or server. This is also true for the enhancements provided by the WorkSpace On-Demand Manager. To be able to perform remote management of the WorkSpace On-Demand Manager, you need to add the WorkSpace On-Demand Manager support to your OS/2 Warp Version 4 client.

Installing the WorkSpace On-Demand Administration Client component is very similar to installing WorkSpace On-Demand 2.0 on the server. In fact, you will use the same installation program. The procedure is almost identical except that during the second phase of the installation, you only need to select the **Command Line Interface and API Upgrade** option. This option updates the LAN Server Administration Tool on OS/2 Warp Version 4 client to support WorkSpace On-Demand 2.0 functions.

#### 5.4.1.1 Phase One

The WorkSpace On-Demand CD-ROM includes Feature Install and all the necessary FixPaks. These components are required to perform the phase one portion of the installation and are automatically installed as part of the WorkSpace On-Demand installation if not already present and at the current levels.

Before you begin the installation, you should stop all network services. You can do this by issuing the following command from a command line:

```
NET STOP REQ
```

While the installation program will attempt to automatically stop network services, experience has shown that this is not always successful. We recommend that you issue the `NET STOP REQ` command to stop network services manually.

Follow the instructions given in Section 5.3.1.1, "Phase One" on page 120, to complete the initial phase of the installation.

#### 5.4.1.2 Phase Two

After the installation program reboots your server and commences phase two of the installation process, it will display the Select Components screen as shown in Figure 33 on page 132.

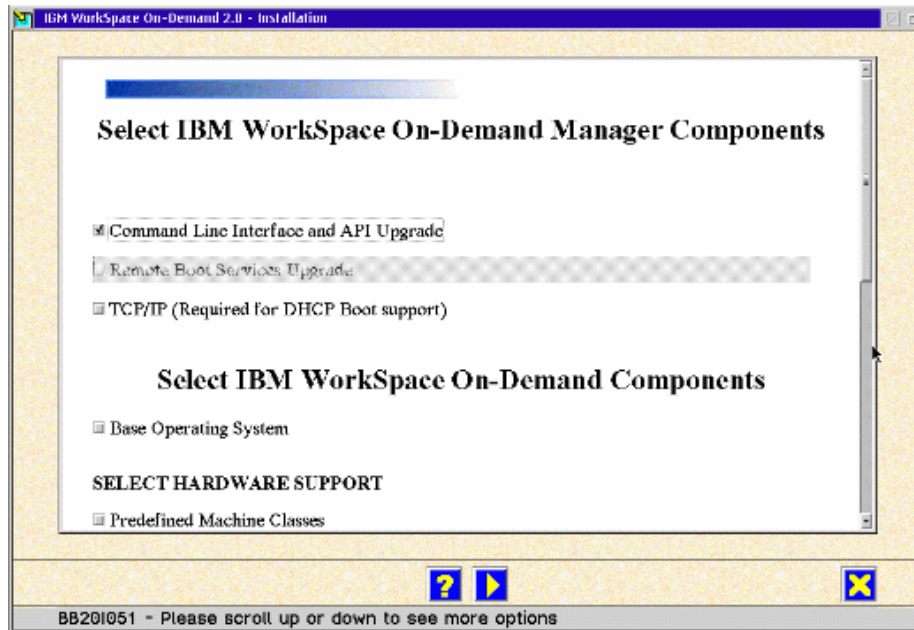


Figure 33. Installing the Administration Client

To install the Workspace On-Demand Administration Client on your OS/2 Warp 4 system, you should select the **Command Line Interface and API Upgrade** option, and ensure that all other options are *deselected*.

To start the install on the client, select the **Install** button. Feature Install will proceed to copy the Workspace On-Demand Manager files associated with the Workspace On-Demand Administrator Client component from the Workspace On-Demand install CD-ROM to the client hard drive.

When you have completed the installation, you must shut down and reboot the client before using the Workspace On-Demand Administration Client utilities. When you reboot the client and start the LAN Server Administration Tool, the GUI will contain the additional objects required to support Workspace On-Demand.

**Note**

If you install new OS/2 Warp Version 4 features after installing the the WorkSpace On-Demand Administration Client, you must reinstall the Administration Client again. This will ensure that any files common to both OS/2 Warp Version 4 and WorkSpace On-Demand are at the correct levels to support WorkSpace On-Demand.

### 5.4.2 Unattended Installation

You can easily install the WorkSpace On-Demand Administration Client using CID techniques in a software distribution environment. The steps necessary to carry out an unattended installation of the WorkSpace On-Demand Administration Client are identical to those described for a WorkSpace On-Demand server in Section 5.3.2, "Unattended Installation" on page 128. However, you must edit the appropriate response file to ensure that only the **Command Line Interface and API Upgrade** option is installed.

---

## 5.5 Uninstalling WorkSpace On-Demand

WorkSpace On-Demand 1.0 included an UNINSTAL utility that removed the WorkSpace On-Demand 1.0 files and directories from a server. This utility is no longer supported in WorkSpace On-Demand 2.0. If you wish to remove WorkSpace On-Demand 2.0 from your server, you can do so manually by following the instructions given in the on-line *WorkSpace On-Demand 2.0 Administrator's Guide*.

**Note**

The WorkSpace On-Demand Manager installation process installs the WorkSpace On-Demand components and FixPaks. The only way to return the server to the state it was in before WorkSpace On-Demand was installed is to restore the server from a backup. That is why we strongly recommend that you back up your target machine before installing any WorkSpace On-Demand Manager components as was noted in Section 5.1, "Before You Install WorkSpace On-Demand" on page 115.





## **Part 3. Managing WorkSpace On-Demand**

This part of the redbook discusses the management aspects of WorkSpace On-Demand.

Chapter 6, "Working with Client Workstations" on page 137, describes how to define client workstations on your WorkSpace On-Demand server by using both the GUI administration tools and the command line interface.

Chapter 7, "Working with Network Applications" on page 167, describes how to define network applications for use by end users on client workstations

Note that we do not include an in-depth discussion of user management in the OS/2 Warp Server domain. End users log on to WorkSpace On-Demand clients using their user IDs and passwords from the OS/2 Warp Server domain in the same way as they log on to their existing "fat client" workstations. This redbook only describes how to grant access for a user to network applications since this procedure has been significantly enhanced with the advent of WorkSpace On-Demand.



---

## Chapter 6. Working with Client Workstations

This chapter describes how to add and delete clients on your WorkSpace On-Demand server using both the OS/2 Warp Server GUI and the command line interface and how to query client information for the clients defined on your server.

---

### 6.1 Defining a Client Workstation

There are two ways to define a client:

- Using the Client Definition notebook within the OS/2 Warp Server Administration GUI. This method is described in Section 6.1.1, "Using the GUI" on page 137.
- Using the `NET RIPLMACH` command from an OS/2 command line interface. This method is described in Section 6.1.2, "Using the Command Line Interface" on page 147.

The GUI method is a useful way to define clients during testing and for small production environments. For larger production environments, however, where you may be defining a large number of clients on a single server, you will probably wish to automate the definition process using the command line interface, possibly in conjunction with REXX scripts and response files.

#### 6.1.1 Using the GUI

The OS/2 Warp Server GUI offers an easy way to define a WorkSpace On-Demand client. At your Remote IPL server, perform the following steps:

1. Open the **LAN Services File and Print** folder on the OS/2 Warp Server desktop.
2. Select the **Local Workstation** folder within the LAN Services File and Print folder.
3. Open the **Remote IPL Requesters** folder within the Location Workstation folder.

Figure 34 on page 138 shows an example of an open **Remote IPL Requesters** folder.

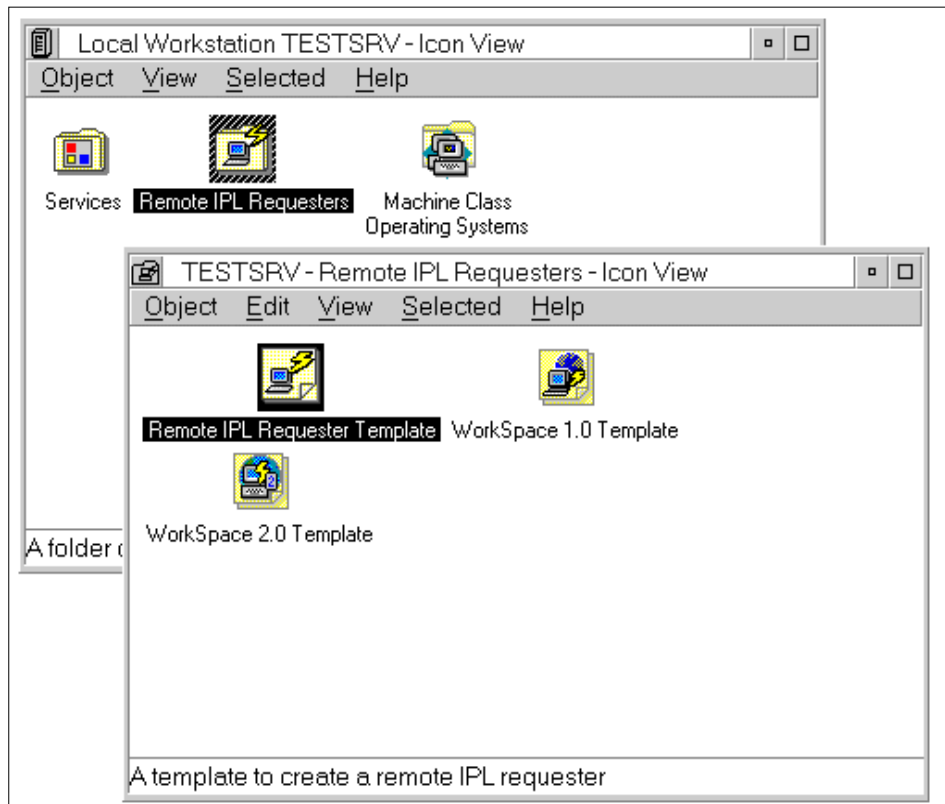


Figure 34. Defining a Client in the Remote IPL Requesters Folder

By dragging the **WorkSpace 2.0 Template** icon to a free place and dropping the template within the same folder, you can define a new client workstation using the Client Definition notebook. The OS/2 Warp Server GUI will automatically open the Client Definition notebook when you drop the icon on a free space within the Remote IPL Requesters folder.

The sections that follow describe each of the pages in the Client Definition notebook. After you have completely entered all the required information, select **Set** to save the changes and close the notebook, or select **Apply** to save the changes without closing the notebook.

### 6.1.1.1 Identifying the Client

You must first identify the client using the **Identity** page of the Client Definition notebook, as shown in Figure 35.

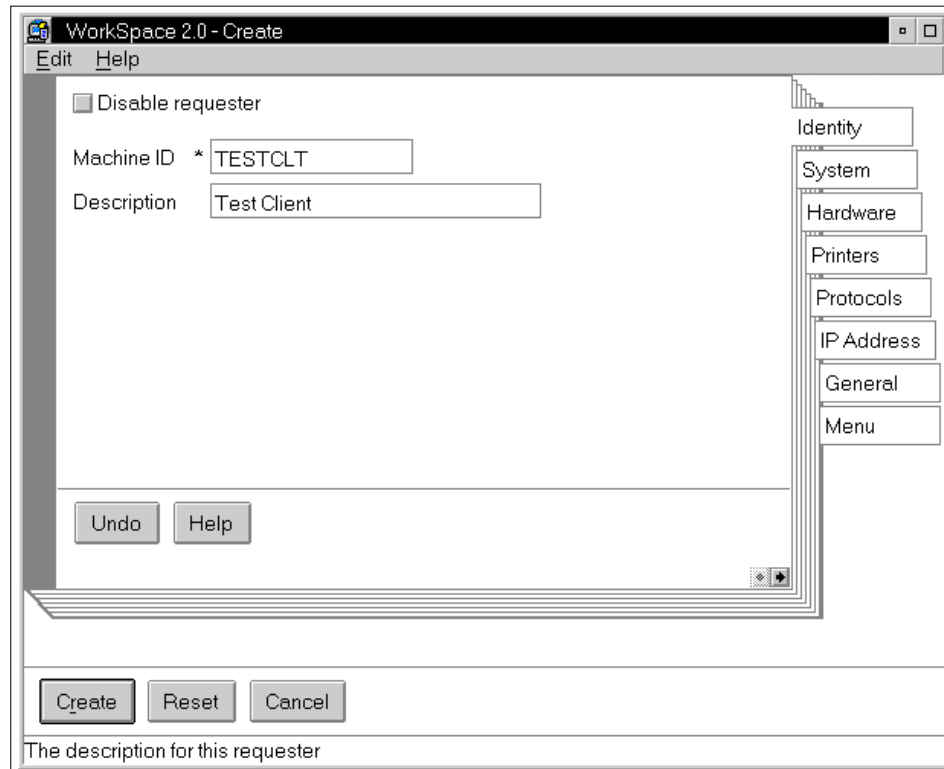


Figure 35. Defining the Client - Identity Page

1. You will usually want to enable the client when creating it. If you do not want to enable it, select **Disable requester**. Disabling the client makes it unavailable for loading from the server; that is, the server will simply not respond to a boot request from that client.
2. Complete the **Machine ID** field. The machine ID is an alphanumeric string of up to 15 characters in length (eight characters if the server's boot drive uses the FAT file system) that is used by the administrator to identify the client. In our example, we have chosen TESTCLT as the Machine ID.
3. Complete the **Description** field. The description is an alphanumeric string of up to 48 bytes that is used by the administrator to describe the client.

### 6.1.1.2 Specifying the Operating System Parameters

Next, you must complete basic operating system parameters on the **System** page in the Client Definition notebook, as shown in Figure 36.

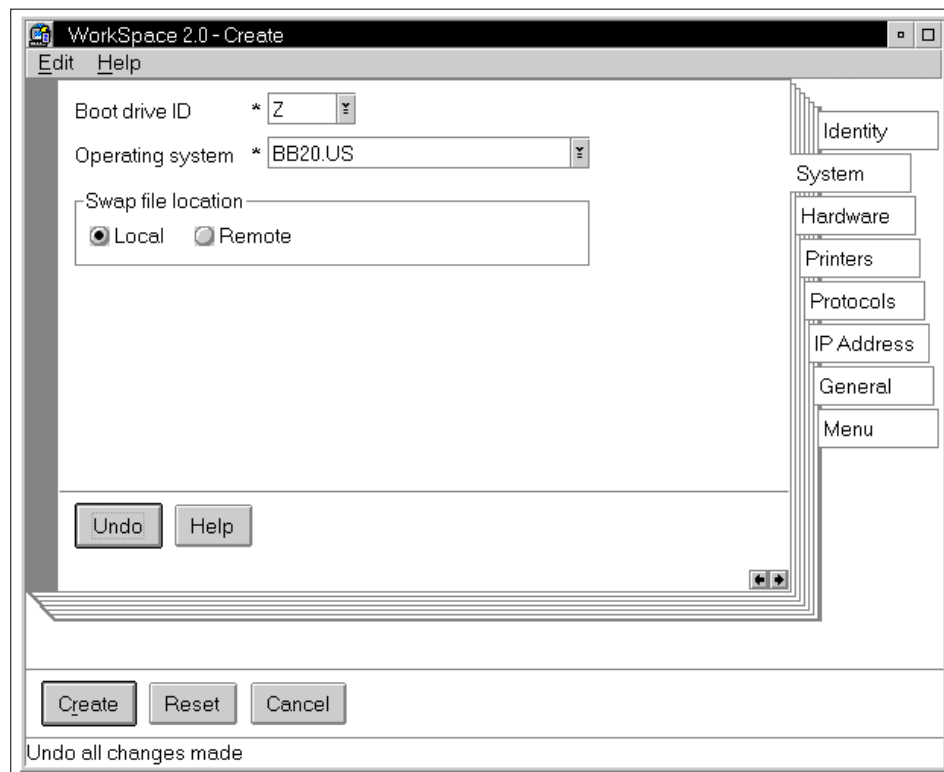


Figure 36. Defining the Client - System Page

1. Select the **Boot drive ID** from the drop-down list. The default is **Z**.

#### Note

If you select Z, you will not be able to use the "next available drive" option when assigning drives.

2. Select the **Operating system** from the drop-down list. Note that OS/2 and DOS are not listed here. If you wish to define an OS/2 or a DOS client workstation, use the **Remote IPL Requester Template**.
3. Choose the location for the swap file, either **Local** (on the client) or **Remote** (on the server). If you choose **Local**, your client *must* have a local hard disk that is partitioned and formatted.

### 6.1.1.3 Specifying the Hardware Settings

You must specify the network adapter address and hardware configuration for the client, using the **Hardware** page, as shown in Figure 37.

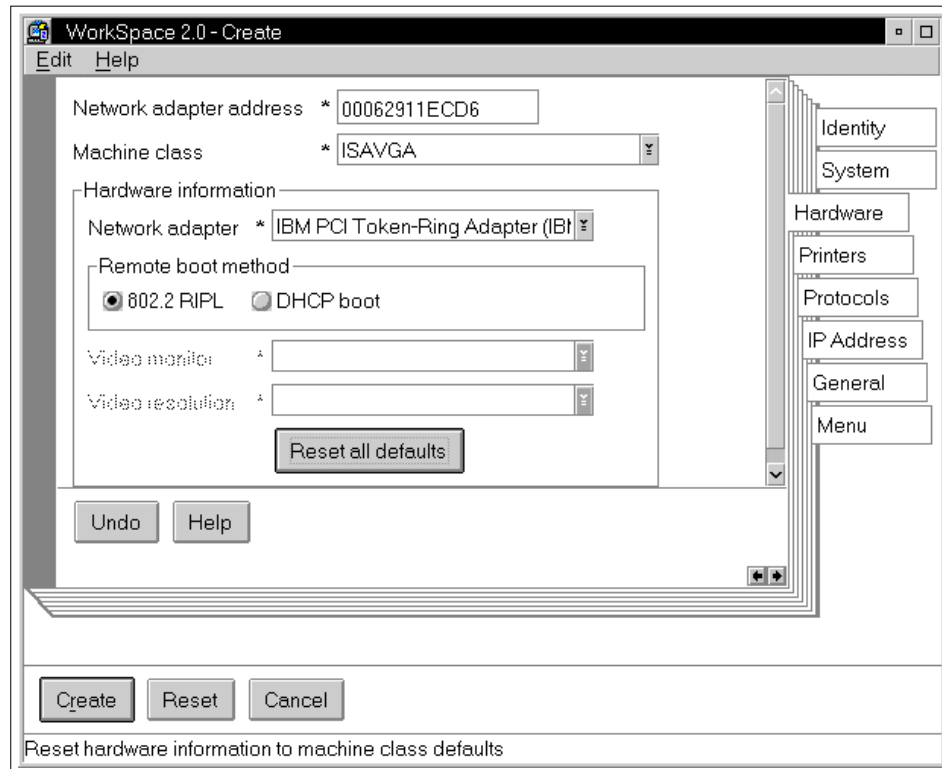


Figure 37. Defining the Client - Hardware Page

1. Complete the **Network adapter address** field by entering the Universally Administered Address of the client's network adapter.

The adapter address is a unique twelve-digit hexadecimal address for the network adapter. You can obtain this address by turning on the client and waiting for it to begin broadcasting on the network. When it does so, find the AA entry on the display; the adapter address is the twelve-character string immediately following this field.

2. Select the correct machine class by selecting the appropriate entry from the **Machine class** drop-down list. The machine class is the description of the hardware environment for a particular client. In the example shown in Figure 37, we are using the ISAVGA machine class.

3. Choose an entry from the **Network adapter** drop-down list. In our example, we have selected the IBM PCI Token-Ring Adapter.
4. Select the remote boot method (IEEE 802.2 RPL or DHCP) by selecting the **IEEE 802.2 RIPL** or **DHCP boot** radio button in the **Remote boot method** group.

If you select **DHCP boot**, you must perform a number of additional steps:

- Configure the DHCP service on the server and the PXE Proxy service if it does not run on the same server as your primary DHCP service.
- Configure the BINL service on the server.

You must define the client to all the required services before you can boot a client using the DHCP PXE boot mechanism. You must perform these steps by using the **DHCP Server Configuration** and **BINL Server Configuration** objects within the **DHCP Server Services** folder or by editing the configuration files manually. See Section 2.4, “The DHCP PXE Boot Mechanism” on page 66, for more information on these services and their configuration files.

**Note**

You can select the **DHCP boot** radio button only if the network adapter that you specify in the **Network adapter** drop-down list supports the PXE specification. If you select an adapter that does not support this specification, the **DHCP boot** radio button is disabled.

5. To change the video monitor, choose an entry from the **Video monitor** drop-down list.
6. To change the video resolution, choose an entry from the **Video resolution** drop-down list.

**Note**

For certain machine classes, you can change the video resolution and the video monitor. Note, however, that the ISAVGA class does not allow you to change either of these options.



### 6.1.1.4 Defining Printers

To define a printer for the client, you must complete the information on the **Printers** page as shown in Figure 38. From this page, you can define network and local printer access for the client.

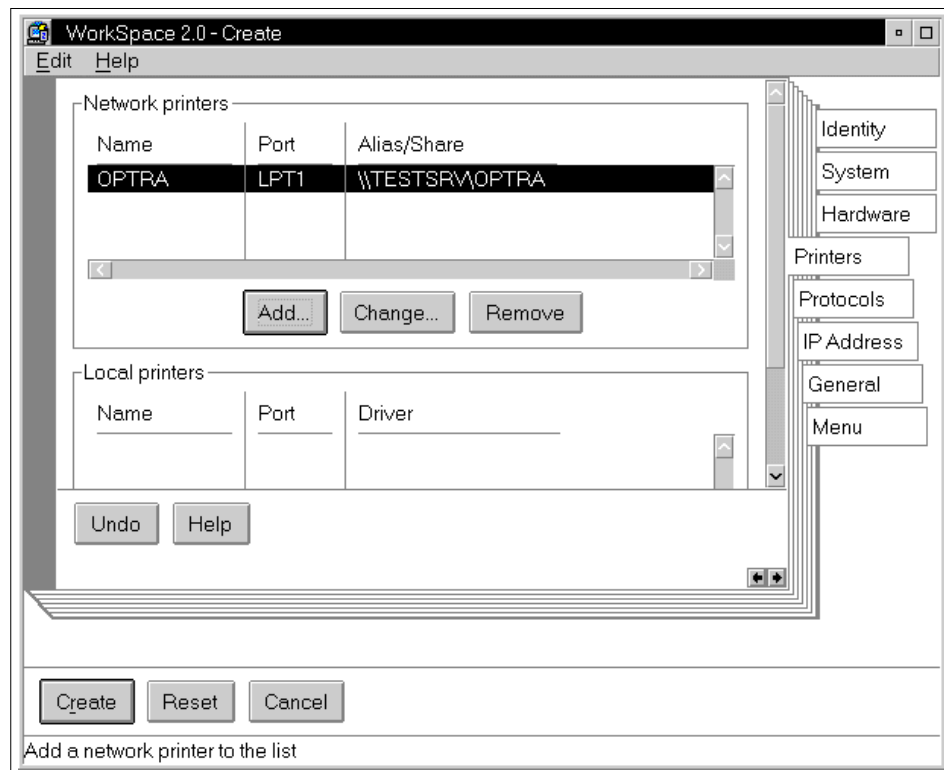


Figure 38. Defining the Client - Printer Page

**Note**  
Depending on the size and resolution of your screen, the **Add**, **Change...** and **Remove** buttons for the **Local printers** list may not be visible. Use the scroll bar to scroll down until the buttons appear.

When you initially select this page, both the **Network printers** and **Local printers** lists will be empty. To add a network printer, complete the following steps.

1. Select **Add**. The **Add Network Printer** pop-up will appear, as shown in Figure 39 on page 144.

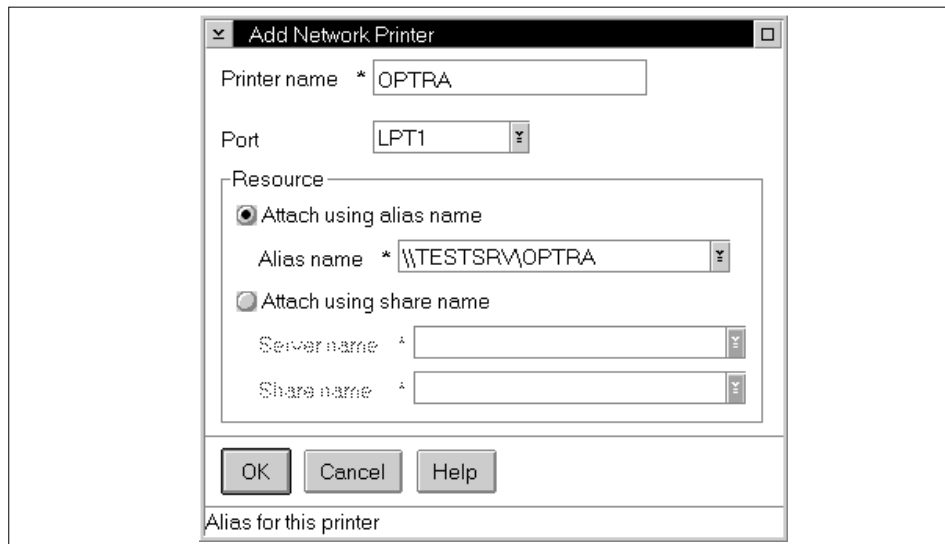


Figure 39. Defining the Client - Adding a Network Printer

2. On the **Add Network Printer** pop-up:
  - Complete the **Printer name** field.
  - Select the desired **Port** from the drop-down list.
  - Select the desired **Attach using alias name** or **Attach using share name** radio button. If **Attach using alias name** is selected, select the printer **Alias name** from the drop-down listbox, or enter a valid alias name. If **Attach using share name** is selected, select the **Server name** and **Share name** from the drop-down list, or enter a valid name in each field.
  - Click on **OK**.

If you have a local printer attached first, ensure that the parallel or serial port that the local printer is attached to is enabled. Then complete the following steps to access the local printer from the client:

1. Select **Add**.
2. On the Add Local Printer pop-up:
  - a. Complete the **Printer name** field.
  - a. Select the desired **Port** from the drop-down list.
  - a. Select the desired **Printer driver** from the list.
  - a. Select **OK**.

### 6.1.1.5 Configuring Network Protocols

You may configure your client to boot using NetBIOS (NETBEUI) or NetBIOS over TCP/IP (TCPBEUI) protocols. In either case, you can configure the client to support both protocols for application use. To configure your primary boot protocol, use the **Protocols** page as shown in Figure 40.

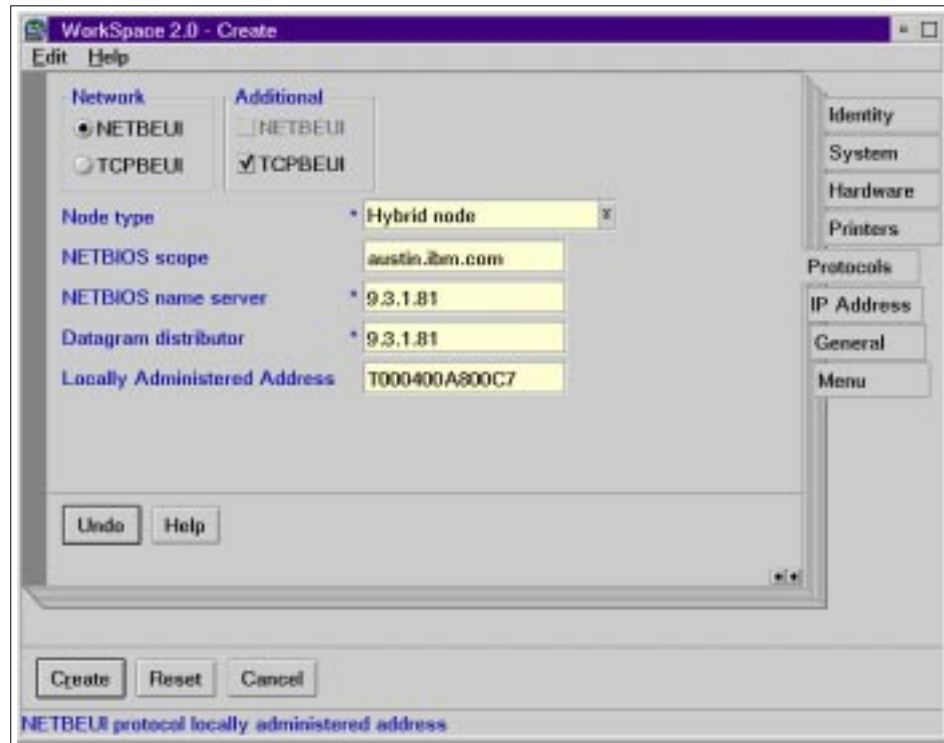


Figure 40. Defining the Client - Protocols Page

Choose your boot protocol by selecting either the **NETBEUI** or **TCPBEUI** radio button in the **Network** group. If you wish to support both protocols, select the remaining protocol in the **Additional** group.

If you wish to support TCPBEUI, select a node type from the **Node type** drop-down list.

In many cases, you will want to select a node type other than the default broadcast node. If you select any entry other than Broadcast node, you must specify the NetBIOS scope and the IP addresses of your NetBIOS name server and datagram distributor in the appropriate fields.

If the client's network adapter supports locally administered addresses (LAAs), you can specify an LAA rather than using the burned-in universally administered address (UAA). Some organizations use LAAs as a means of identifying nodes within the corporate network for management purposes.

To specify a locally administered address for your client, enter an identifier (I for IEEE standard notation, Ethernet format and T for Token-Ring), followed by a twelve-digit hexadecimal address in the **Locally Administered Address** field.

#### 6.1.1.6 Configuring the TCP/IP Protocol

If the WorkSpace On-Demand client will use TCP/IP to run network applications after the boot process is complete, configure the TCP/IP settings using the **IP Address** page, as shown in Figure 41.

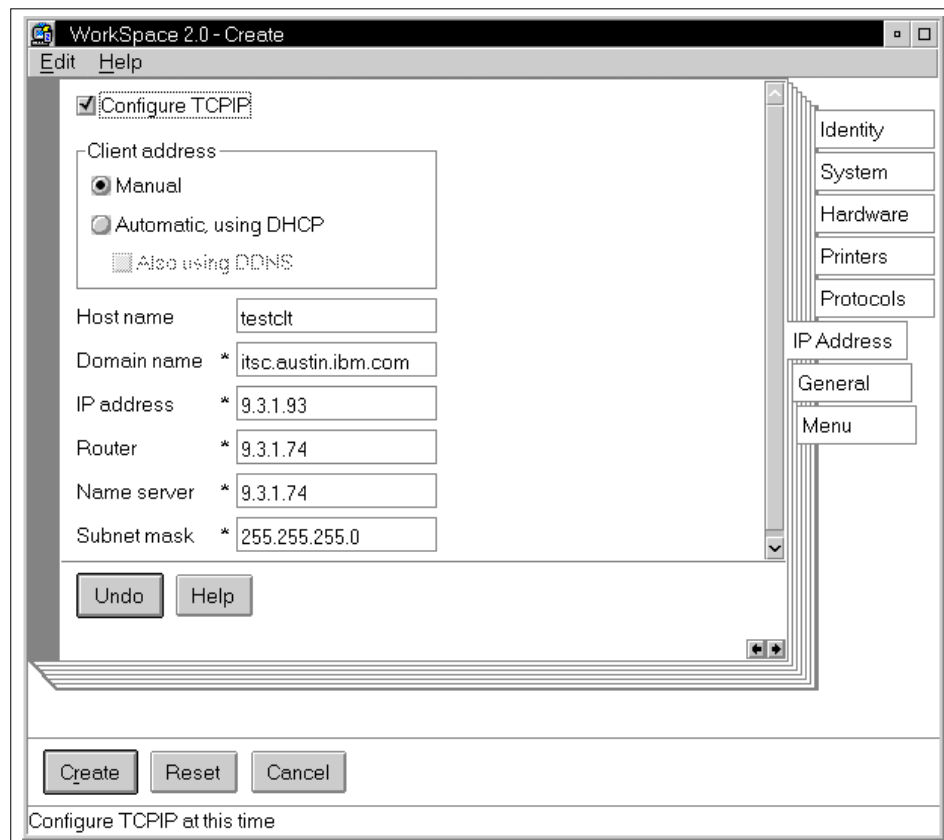


Figure 41. Defining the Client - IP Address Page

1. Select **Configure TCP/IP** if you want to perform TCP/IP configuration. All other fields are disabled until you select this checkbox.
2. For client address, select **Automatic, using DHCP** if your network uses automatic addressing through the dynamic host configuration protocol, or select **Manual**. If automatic is selected, you may also optionally select **Also using DDNS** if you want to use Dynamic Domain Name Services.  
If you select **Manual**, you must complete the **IP address** and **Subnet mask** fields.
3. Complete the **Host name** field. This field specifies the host name of the RIPL requester machine and can contain up to 32 alphanumeric characters.
4. Complete the **Domain name** field. This field specifies the domain name of the RIPL requester machine and can contain up to 40 alphanumeric characters.
5. Complete the **IP address** field. This field specifies the IP address of the RIPL requester machine and can contain an IP address of up to 15 numeric characters (for example 123.123.123.123).
6. Complete the **Router** field. This field specifies the router address of the RIPL requester machine and can contain an IP address of up to 15 numeric characters (for example 123.123.123.123).

Select the **Create** button to create the new client. After a short amount of disk activity, you should be ready to remote boot the client machine.

To understand the steps taken by the server when you select the Create button in the Client Definition notebook, see Section 6.1.3, "The Client Definition Process" on page 150.

### 6.1.2 Using the Command Line Interface

The `NET RIPLMACH` command allows you to define new clients. You can use response files to create multiple new Workspace On-Demand clients. Figure 42 on page 148 shows the command syntax of the `NET RIPLMACH` command.

```
NET RIPLMACH /ADD /RS(P):response_file_pathname
              [/SE(RVER):ripl_server_name]

/ADD
    Specifies the add operation. This is a required
    keyword to add or create the machine.

/SE(RVER):
    Specifies the server name of the RIPL server
    where the RIPL requester definition is created.
    If you are on the server, this is an optional
    parameter and the default is the local machine.
    If you are on the administrator client, this is
    a required parameter.

/RS(P):
    Specifies the fully qualified path to a response file.
    The path can be a UNC name.
```

Figure 42. NET RIPLMACH Command Syntax

You can use the `NET RIPLMACH` command to create a complete client definition from the command line. However, the most useful aspect of the `NET RIPLMACH` command is its ability to accept a response file as a parameter. Using a response file, you can create many client definitions on a with a single invocation of the `NET RIPLMACH` command.

**Note :**

You must provide a fully qualified path name for the response file in the `/RS:` parameter of the `NET RIPLMACH` command (UNC names are allowed). If you do not provide a fully qualified path name, the `NET RIPLMACH` command will not accept the response file.

Figure 43 on page 149 shows an example of a response file used with the `NET RIPLMACH` command to create clients. Although this response file contains definitions for only two clients, you can include as many clients as you wish in a response file of this type.

```
[WSODCLT1]

/os:bb10.us
/cl:isavga
/mac:10005a123451
/rem:"This is WorkSpace client 1"
/ada:"ibm t-r shared ram family (UP/SMP, IBMTOK.OS2)"
/dhcp:y
/tcpn:rplclient1
/tcpd:austin.ibm.com

[WSODCLT2]

/os:bb10.us
/cl:isavga
/mac:10005a123452
/rem:"This is WorkSpace client 2"
/ada:"ibm t-r shared ram family (UP/SMP, IBMTOK.OS2)"
/dhcp:y
/tcpn:rplclient2
/tcpd:austin.ibm.com
```

Figure 43. Response File to Create WorkSpace On-Demand Clients

You can also use the `NET RIPLMACH` command to modify client configurations using a response file.

**Important !**

Each line in the response file must end with a valid character and not with spaces. The processing of the response file will fail if there are extra spaces at the end of any line. The `NET RIPLMACH` command will fail with a message such as:

```
WSCLT4: NET3952: You entered a value that is not valid forthe /OS option.
```

### Invalid Parameter

In this example, we did not use the `/RES` parameter to define monitor resolution in the response file. However, other testing has shown that only lower case x's are valid as separators for video resolution. For example, to express an 800 by 600 by 1024 screen at 75 Hertz refresh rate, the correct parameter input would be:

```
/RES:800x600x1024@75Hz
```

Using an upper case X will result in an error message indicating that `NET RIPLMACH` has an invalid parameter.

It is possible to further automate the client definition using additional tools such as TME 10 Netfinity and some custom-built REXX scripts. A discussion of this process is included in Section 6.2, "Defining Client Workstations Automatically" on page 151.

### 6.1.3 The Client Definition Process

When you select the **Create** button in the Client Definition notebook or run the `NET RIPLMACH` command to create the new client definition, the server does the following:

- Adds a new workstation record to the server's RPL.MAP file, as shown in Figure 4 on page 44.
- Creates a machine FIT file for the client in the `\IBMLAN\RPL\FITS` directory.
- Creates two directories for the client-specific files:

```
\IBMLAN\RPL\MACHINES\client_name  
\IBMLAN\RPLUSER\client_name
```

Files that are required by the client are placed in the above directories. See Section 2.2.6, "Client-Specific Files and Directories" on page 57, for a listing of the files that are placed in these directories and how they are constructed.

- Creates a "user ID" for the client. This enables the server to grant access to the above directories. For example, the client's user ID has RWCXDA rights to the `\IBMLAN\RPLUSER\TESTCLT` directory.
- Adds the client's user ID to the RPLUSER group, which has RX access rights to the `\IBMLAN\RPL\` directory. This allows the client to access the client operating system image at boot time.



---

## 6.2 Defining Client Workstations Automatically

In a large enterprise network, you may need to manage thousands of clients spread over several hundred locations. To create the client definitions in the WorkSpace On-Demand servers, you must know the burned-in network adapter address (universally administered address) of each client in order to define it to the server. Since your servers and clients may reside in different locations, it may require considerable effort to query the adapter information, create the new client definitions, and apply the correct client name to a new machine.

In order to define a new client on a WorkSpace On-Demand server, you need the following information:

- The name of the new client
- The type of network adapter
- The network adapter's universally administered address
- The machine class

In an enterprise environment, the name of a new client is typically predetermined in accordance with some kind of organizational naming convention. The information about the type of network adapter can also be easily determined since client machines are typically ordered from the manufacturer in a predefined configuration.

In this section of the redbook, we describe a process to automatically detect new clients being added to the network and to dynamically query the necessary information and define these clients on a WorkSpace On-Demand server. The sequence of steps in this process is illustrated in Figure 44 on page 152.

### Note

The automated client detection and configuration process described in the remainder of this section is *not* a supported part of the WorkSpace On-Demand product. It has been developed during the production of this redbook and has undergone some rudimentary testing. You must test it to determine whether it meets the requirements for use in your organization. The process and the accompanying utilities are provided without any warranty, either expressed or implied.

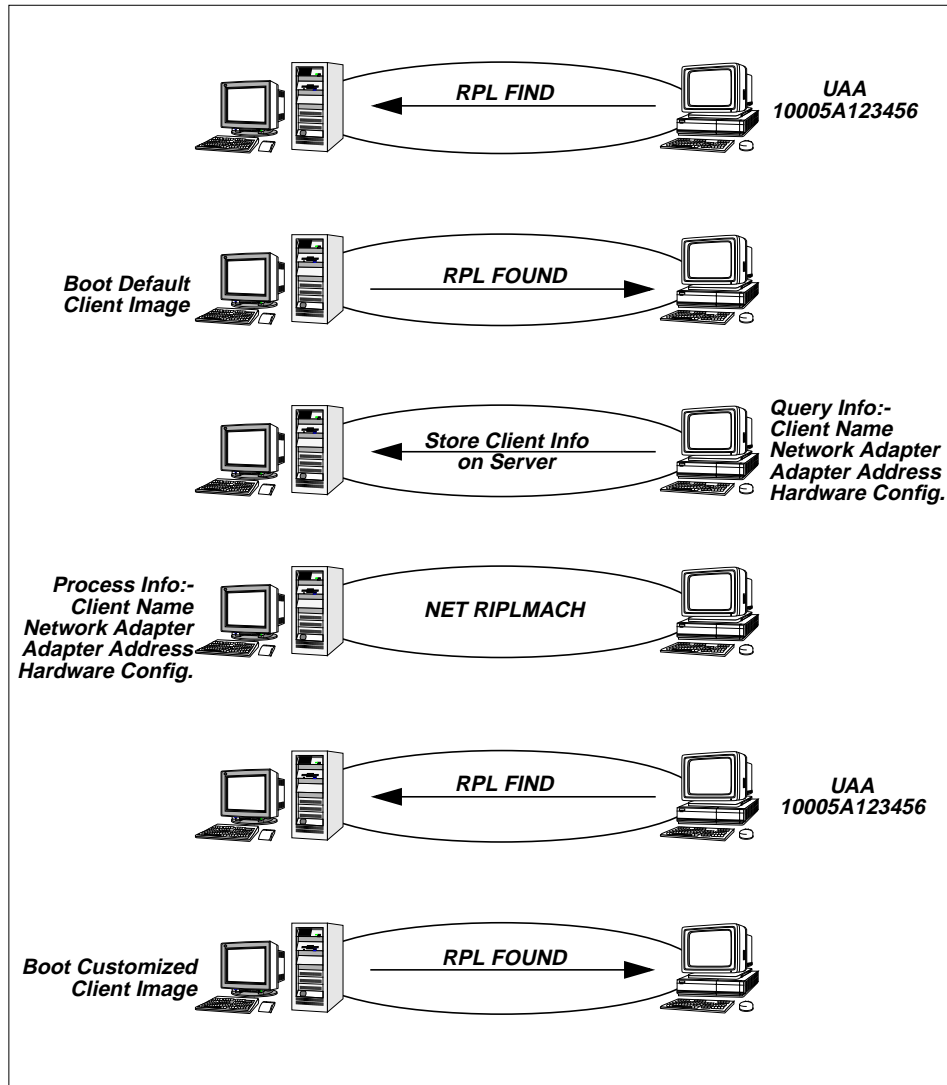


Figure 44. Automated Client Detection and Definition

### 6.2.1 The Process

Before you can run the automatic definition process, you must install the necessary components on your WorkSpace On-Demand server. The following sections describe the steps that you must follow.

### 6.2.1.1 Install TME 10 Netfinity

The automatic definition process uses TME 10 Netfinity to gather information from the client. You must therefore install TME 10 Netfinity on the WorkSpace On-Demand server.

### 6.2.1.2 Create a Directory on the Server

You must create a directory on the server with write access for the new clients to store their data. In our testing, we created a directory D:\NEWCLTS to store the configuration data. We created a network alias for this directory named NEWCLTS using the following command:

```
NET SHARE NEWCLTS=D:\NEWCLTS.
```

This alias is necessary because we want to refer to this directory using a FIT file. You must create an access control profile to provide the RPLGROUP user group with read, write, create, execute, and delete access to this directory using the following command:

```
NET ACCESS NEWCLTS /ADD RPLGROUP:RWXCD.
```

This enables every new client to write its configuration data to this directory.

### 6.2.1.3 Create a Query Program

You must now create a program to query the client's hardware configuration. This program will invoke the TME 10 Netfinity System Information Tool and write the resulting data to a file in the directory that you created in Section 6.2.1.2, "Create a Directory on the Server" on page 153.

In our testing, we created a REXX script named QUERYCFG.COM to handle these tasks. The source code for this program is included in Appendix A.1, "Client Hardware Query Program – QUERYCFG.COM" on page 321.

When the client boots for the first time, the boot sequence will invoke QUERYCFG.COM, which will immediately ask the user for the client name. We assume that the person setting up the client machine will be provided with written information about the name of each client.

After obtaining the client name and verifying that it is unique in the network by comparing it against a file of known names, QUERYCFG.COM invokes the TME 10 Netfinity System Information Tool through the SINFG30 program. This program has command line options that allow you to store the data in a given file. If you wish, you can use a DB2/2 table to store the data, but in this example, we chose to use a simple text file named *machine\_id*.DATA, where *machine\_id* is the name of the client.

Note that the TME 10 Netfinity System Information Tool does not query the network adapter address, which is required in order to define the client. QUERYCFG.CMD therefore appends the contents of the client's LANTRAN.LOG file to the *machine\_id*.DAT file in order to include the adapter address.

After storing the hardware configuration data, QUERYCFG.CMD then waits for the completion of a server process that creates the final WorkSpace On-Demand Client definition. When this process completes, QUERYCFG then shuts down the system.

You can add more function to a batch process like this. One possible extension would be to test for the presence of a local disk drive and possibly format this drive if no valid file system is found. For this example, however, we wanted to give an example for the general technique used and therefore made QUERYCFG as simple as possible.

#### **6.2.1.4 Create a Model Client Definition**

You must now create a model client definition for a new client on the WorkSpace On-Demand server. This model client definition is used to boot a client workstation for the first time when it is initially inserted into the network. You will modify the boot process for this model client so that it invokes the QUERYCFG.CMD program to query and record the client's hardware configuration.

You can create the model client definition by defining a client to WorkSpace On-Demand using an adapter address that contains wild cards. For example, we defined an entry with the adapter address 00062???????. Each wildcard ? position is not tested when the server detects a RPL FIND frame. Hence, a match will be found for any network adapter that has an address beginning with 00062.

If you have many different network adapter types, you can create a model client definition for each adapter type that you use. Since the UAAs are vendor- and type-specific in the first six numbers, you can create model definitions with adapter addresses that have positions seven through twelve replaced by the ? wildcard specification.

In our testing, we created client definitions for the IBM PCI Auto Wake On LAN and the Madge Token-Ring adapters. The resulting RPL.MAP file is shown in Figure 45 on page 155.

```

; workstation records
1000FFFFFFFF DEFAULT40 ~ FITS\DEFAULT40 DELLSRV Z ~ ~ ~ , , , ~ R_240_OTK ~ ~
00062??????? MODEL ~ FITS\MODEL DELLSRV Z ~ ~ ~ , , , ~ R_BB20_US_OTKTRP ~ ~ ~
0000F??????? MDGMODEL ~ FITS\MDGMODEL DELLSRV Z ~ ~ ~ , , , ~ R_BB20_US_OTKMDGSMT ~ ~ ~

```

Figure 45. RPL.MAP File with Model Client Definitions

Every new PC with an adapter address matching the specified positions from the model definition will start a RIPL process and boot a system as configured for the model.

**Note**

We did not test a large number of adapter types; so, while it might appear that different versions of a similar adapter (for example, PCI and ISA versions) have the same adapter address in the first six positions, they may in fact use different drivers. You should test this approach for the types of adapters you intend to use in your enterprise.

**Change the Model Client's Boot Process**

You must modify the model client's boot process so that it will invoke the QUERYCFG.CMD program. You can do this by changing the CONFIG.SYS file in the \IBMLAN\RPL\MACHINES\MODEL directory, replacing the default RUNWORKPLACE statement with a command line (CMD.EXE) and appropriate parameters as shown in Figure 46.

```

SET RUNWORKPLACE=Z:\OS2\CMD.EXE /k z:\util\querycfg.cmd
DEVICE=Z:\NETFIN.40\BIN\SYSINFO.SYS

```

Figure 46. Model Client CONFIG.SYS Changes

The change of the RUNWORKPLACE statement forces the client to process the QUERYCFG.CMD instead of PMLOGON.EXE to obtain the hardware configuration.

You must add the DEVICE=SYSINFO.SYS statement to this CONFIG.SYS file to support the TME 10 System Information Tool that gathers the hardware information.

**Change the Model Client's FIT File**

You must modify the model client's machine FIT file as shown in Figure 47 on page 156 to add redirection commands for the TME 10 Netfinity System

Information Tool, the \UTIL directory where QUERYCFG.COM resides and to the directory where QUERYCFG.COM will store the client configuration data.

```

; Netfinity and Utility support for automated detection of new clients
z:\NETFIN.40      BB20.US\NETFIN.40
z:\UTIL          BB20.US\UTIL
z:\*.data        \\DELLSRV\NEWCLTS
z:\*.ok          \\DELLSRV\NEWCLTS

```

Figure 47. Model Client FIT File Changes

Note that you can change these directories to point to other locations if you wish, provided they match the locations of the programs and directories that you have created on your server.

#### May Also Run with OS/2 Warp Definition

A model client definition can possibly be made for an OS/2 Warp client with the same functionality. Although this has not been tested. The REXX scripts shown here are specific to WorkSpace On-Demand.

### 6.2.1.5 Create a Client Generation Program

After QUERYCFG.COM has written the client configuration data to your server, you must create another program that creates "real" client definitions for each client. In our testing, we created a program named ADDNEWCL.COM. The source code for this program is included in Appendix A.2, "Client Definition Program – ADDNEWCL.COM" on page 322.

The ADDNEWCL.COM program has the following parts:

- A main process that runs every 60 seconds
- The ScanClt procedure, which scans the *machine\_id*.DATA file and searches for an entry that contains a universally administered address. The value for the network adapter address is stored in a global variable named ADDRESS.
- The QueryAdp procedure, which fills the variable *adpType* with the name of the network adapter type based on the list in the *AdpList* STEM variable.
- The LogMsg procedure, which is used to log the program execution.
- The CreateClt procedure, which is used to create the new clients. It scans through the RPL.MAP file and checks for a workstation record with the adapter address of the new client.

Any client that does not have an entry specified in the RPL.MAP file, but for which the adapter address fits the global mask defined for the model client, will be booted using the model client definition, and an entry is automatically created in the RPL.MAP file, as shown in Figure 48.

```

; workstation records
1000FFFFFFFF DEFAULT40 ~ FITS\DEFAULT40 DELLSRV Z ~ ~ ~ , , ~ R_240_OTK ~ ~
00062??????? MODEL ~ FITS\MODEL DELLSRV Z ~ ~ ~ , , ~ R_BB20_US_OTKTRP ~ ~ ~
0000F??????? MDGMODEL ~ FITS\MDGMODEL DELLSRV Z ~ ~ ~ , , ~ R_BB20_US_OTKMDGSMT ~ ~ ~
0000F62A5B29 MDGMODEL ~ FITS\MDGMODEL DELLSRV Z ~ ~ ~ , , ~ D_BB20_US_OTKMDGSMT ~ ~ ~
000629844293 MODEL ~ FITS\MODEL DELLSRV Z ~ ~ ~ , , ~ D_BB20_US_OTKTRP ~ ~ ~

```

Figure 48. RPL.MAP File with Model Client Definitions

This entry is initially disabled. This line must be deleted from the RPL.MAP file, or the NET RIPLMACH command will fail with an error saying that there is already an entry with a matching adapter address in the RPL.MAP file.

The program calls the QueryAdp procedure to determine the type of network adapter in the client. In our example, QueryAdp simply checks the name of the model client definition and references this against a list of known adapters. In a production environment, however, you might wish to use a more flexible method like that shown in Figure 49.

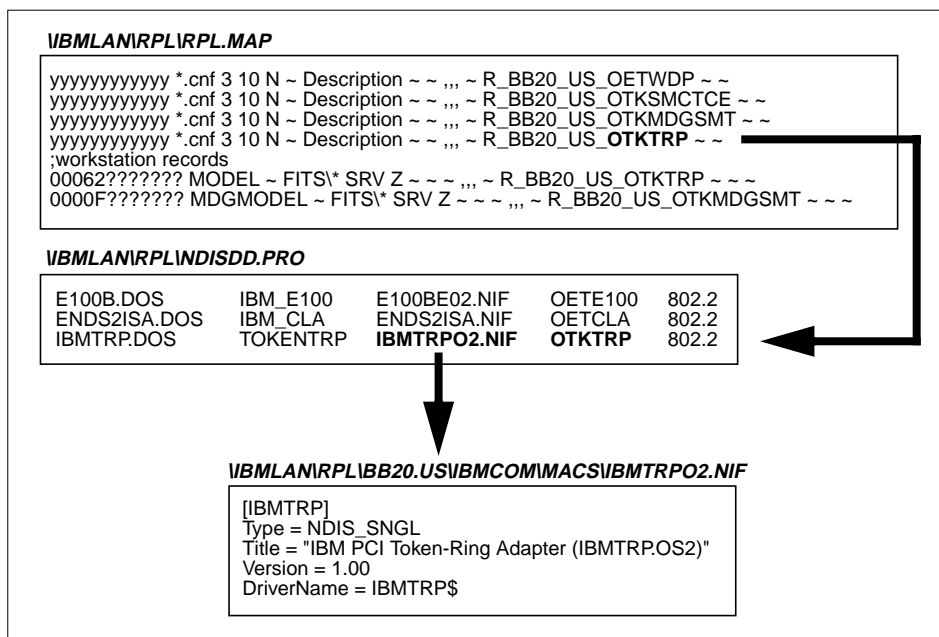


Figure 49. Dynamic Detection of Network Adapter Name

If you scan the client line with the model definition, you find the pointer to the server record. Using the information in the activated server records of the RPL.MAP file, the corresponding entries in the NDISDD.PRO and finally the text from the .NIF file, you can obtain the names for the supported network adapters.

This is the safest way to have the correct values for the `/ADA` parameter of the `NET RIPLMACH` command because you can scan through the files during the execution of the program. The values must not be hardcoded in the program. They will be set dynamically.

After determining the necessary parameters, the program calls the `CreateClt` procedure to create a new client definition. `CreateClt` sets the parameters for the `NET RIPLMACH` command to the following values:

- The machine class is set to `ISAVGA`.
- The operating system is set to `BB20.US`.
- The adapter address is set to the value of the variable `address`.
- The client name is set to the value of the variable `cltname`.
- The remark is set to *WorkSpace On-Demand client* + the client name.
- The adapter type is set to the value of the variable `adptype`.
- The stdout and stderr output is redirected to append to the Log file.

`CreateClt` then invokes the `NET RIPLMACH` command with these parameters. The RPL.MAP file then contains a "real" workstation record for the client, as shown in Figure 50.

```

; workstation records
1000FFFFFFFF DEFALT40 ~ FITS\DEFALT40 DELLSRV Z ~ ~ ~ , , ~ R_240_OTK ~ ~
00062??????? MODEL ~ FITS\MODEL DELLSRV Z ~ ~ ~ , , ~ R_BB10_US_OTKTRP ~ ~ ~
0000F??????? MDGMODEL ~ FITS\MDGMODEL DELLSRV Z ~ ~ ~ , , ~ R_BB20_US_OTKMDGSMT ~ ~ ~
0000F62A5B29 MDGMODEL ~ FITS\MDGMODEL DELLSRV Z ~ ~ ~ , , ~ D_BB20_US_OTKMDGSMT ~ ~ ~
000629844293 ITSOLAB2 ~ FITS\ITSOLAB2 DELLSRV Z ~ ~ ~ , , ~ R_BB20_US_OTKTRP ~ ~ ~

```

Figure 50. RPL.MAP File after Client Definition

As you can see, one client is already defined, and another one of the type `MDGMODEL` is not yet processed. This entry is still disabled by the `D` instead of the `R` at the beginning of the workstation type field of that line.

After finishing the `CreateClt` procedure, the main program renames the `machine_id.DATA` file to `machine_id.OK`. The existence of this file is the signal for the `QUERYCFG.CMD` program running on the client that the client



definition is done on the server. QUERYCFG.CMD then reboots the client, and when the reboot completes, the client will have a fully functional WorkSpace On-Demand client definition.

#### **Avoid Duplicate NetBIOS Names**

Every new client that is not yet defined will be booted with a client name as defined in the model configuration. We have not tested an environment with multiple machines that had the same network adapter installed, but it is likely that only one model client at a time can become active and run through this process. As soon as the client is rebooted, the next machine can be processed. You may extend the procedures in your environment to resolve this conflict.

### **6.2.2 Possible Extensions**

The automatic definition process described here is only a small subset of possible functions. The intention of the example is to provide you with an illustration of how you can automate the client definition procedure. You can take this example and add additional features and levels of functionality as you wish.

By using standard tools such as the TME 10 Netfinity System Information Tool, you can customize the WorkSpace On-Demand environment to meet your enterprise's requirements. For example, TME 10 Netfinity provides information on the hardware configuration of your client machines. If you use IBM hardware, the exact model description is shown in the output data file. In one of our tests, we used an IBM Personal Computer 350, Machine Type/Model 6586/5SH. The output provided by the TME 10 Netfinity System Information Tool for this machine is similar to that shown in Figure 51 on page 160.

```
***** Model and Processor Information *****
Model Name           : IBM Personal Computer 350
Machine Type/Model  : 6586/5SH
Processor            : Pentium( tm )
Math Coprocessor    : Pentium( tm )
Number of Processors : 01
Processor Speed     : 100 MHz
System Board Speed  : 50 MHz
Internal Processor Cache : Enabled
Model                : FC
Submodel             : 01
BIOS Revision       : 00
BIOS ROM Date       : 10-26-1995
Build ID            : LPKT53A
Dedicated IRQ Levels : 0 1 2 8 12 13 14 15
Shared IRQ Levels   : 11
```

Figure 51. TME 10 NetfinityNetfinity System Information Tool Output

This shows that the tool obtains the correct Type/Model ID for IBM PC hardware. If you create machine classes as described in Chapter 10, “Supporting Additional Hardware” on page 251, you can easily create client definitions that match the machine class for the hardware that you are using.

The TME 10 Netfinity System Information Tool also shows the memory size installed in the client. You may therefore wish to customize the swap-file location (local or remote) depending on the memory size of the client. This parameter can be set in the parameter list of the NET RIPLMACH command.

For non-IBM hardware, you should build the machine classes corresponding to the video adapters installed because the video adapter is the most complex configuration item for a machine class. Figure 52 on page 161 shows an extract of the output from TME 10 for the video configuration of a non-IBM PC.

```
***** VGA-compatible controller *****  
Manufacturer           : CIRRUS LOGIC  
Display controller     : VGA-compatible controller  
Class Code             : 0003  
PCI Bus Number 0      : Device Number 02  
Vendor ID              : 1013  
Device ID              : 00A8  
Revision ID           : 00FC
```

Figure 52. TME 10 Netfinity System Information Tool Video Adapter Detection

The specific video adapter is shown in the Class Code, Vendor ID and Device ID fields. If you create and name your machine classes based on the information that the System Information Tool provides, you can create machine class definitions for non-IBM hardware almost as easily as for IBM hardware.

---

### 6.3 Deleting a Client

There are two ways to delete a client definition from a WorkSpace On-Demand server:

- Using the OS/2 Warp Server GUI
- Using the command line interface and the NET RIPLMACH command

The GUI interface provides a useful means of deleting a single client or a small group of clients during testing or for day-to-day administration. To delete large groups of clients however (for example, when reconfiguring a server), the command line interface provides a more efficient method.

Note that when you delete a client from the server, you delete not only the client definition, but all the client-specific directories and files that relate to that client. See Section 6.1.3, “The Client Definition Process” on page 150 for a list of these client-specific directories and files.

#### 6.3.1 Using the GUI

Deleting a client from your WorkSpace On-Demand server is similar to deleting any other object from the Workplace Shell desktop:

1. Locate the client’s icon in the **Remote IPL Requesters** folder (see Figure 34 on page 138)
2. Right-click on the icon to display a context menu.

### 3. Select the **Delete** menu option.

WorkSpace On-Demand Manager will then delete the client definition along with its client-specific files (such as its machine FIT file) and directories.

## 6.3.2 Using the Command Line Interface

You can use the NET RIPLMACH command to delete WorkSpace On-Demand clients from a server, by specifying the /DELETE parameter. Note however, that you cannot use a response file when using the /DELETE parameter with the the NET RIPLMACH command. If you wish to delete multiple clients with a single invocation of the NET RIPLMACH command, you must therefore use a REXX script such as that shown in Figure 53 to automate this operation.

```

/* Sample REXX script to delete a number of clients
   given by a file with the client names */

parse upper arg dellist
call stream dellist, 'c', 'open read'
if result <> "READY:" then
do
  say "Error opening file"
  exit x2d('0812',4)
end
clt.0=0
i=0
do while lines(dellist)
  i=i+1
  clt.i = linein(dellist)
  clt.0 = i
end
call stream dellist, 'c', 'close'
i=1
do clt.0
  "net riplmach /delete "clt.i
  i=i+1
end

```

Figure 53. Deleting Multiple Clients - Sample REXX Script

This REXX script accepts a file name as a parameter. This file contains the list of clients to be deleted. The REXX script then iterates through the list and deletes each client in turn using the NET RIPLMACH command. An example of this file is shown in Figure 54 on page 162.

```

WSODCLT1 WSODCLT2 WSODCLT3 WSODCLT4 WSODCLT5
WSODCLT6 WSODCLT7 WSODCLT8 WSODCLT9 WSODCLT10

```

Figure 54. Deleting Multiple Clients - Input File

## 6.4 Querying Client Information

You can also use the NET RIPLMACH command to gather information about a defined WorkSpace On-Demand client. Using NET RIPLMACH without any parameters shows you a list of defined WorkSpace On-Demand clients with their associated operating system and status (enabled or disabled).

You can issue the NET RIPLMACH command with a specific client name that you wish to query, and the command will provide output as shown in Figure 55 on page 163.

```
RIPL requester name          TESTCLT
Remark                       Test Client
Requester definition enabled  Yes
Network adapter address      00062911ECD6
Locally administered address T000400A800C7
Network adapter type         IBM T-R Shared RAM Family (UP/SMP,
                             IBMTOK.OS2)
Operating system version     BB20.US
RIPL machine class name      IBM300GL
Boot drive                    Z
Location of SWAPPER.DAT      Local
Video monitor                 IBM 17V/17S Color Monitor 17 inch, IBM
                             Corporation
Video resolution              1024 x 768 x 256 x 72Hz
DHCP-enabled                  No
DDNS-enabled                  No
Machine IP address            9.3.1.93
Router IP address             9.3.1.74
Nameserver IP address         9.3.1.74
Subnet mask                    255.255.255.0
IP hostname                   testclt
IP domain name                itsc.austin.ibm.com
Boot protocol                  802.2
Network protocol              NETBEUI
Additional protocol            TCPBEUI
NetBIOS datagram distributor  9.3.1.81
NetBIOS name server           9.3.1.81
Node type                      Hybrid
NetBIOS scope

Local printers attached to this requester:
  None

Network printers available to this requester:
  None

The command completed successfully.
```

Figure 55. Querying Client Information Using NET RIPLMACH

This command is useful if you want to query the WorkSpace On-Demand topology of a server remotely.

You can periodically run a program as shown in Figure 56 on page 164 to gather the information for the currently defined WorkSpace On-Demand clients attached to a particular server.

```
/* Sample REXX script to gather client information */

cltlist='clients.lst'
cltdata='clients.dat'

'del 'cltlist
'del 'cltdata

clt.0=0
i=0
do while queued() > 0
  pull entry
end /* do */

/* Query clients */

call stream cltlist, 'c', 'open write'
if result <> 'READY:' then
do
  say "Error opening file : "cltlist
  exit x2d('0812',4)
end /* do */

'net riplmach | rxqueue' /* Querying info to pipe */

do while queued() > 0
  pull entry
  i=i+1
  clt.i=entry
  say "Clnt."i" = "clt.i
  clt.i = translate(clt.i)
  if pos('ENABLED',clt.i) = 0 & pos('DISABLED',clt.i) = 0 then
  do
    say "No valid client line : "clt.i
    i=i-1
  end
  else
  do
    clt.i=word(clt.i,1) /* reduce line to the client's name */
  end
end /* do */

clt.0=i
i=1
do clt.0
  say 'Querying information for client : 'clt.i
  'net riplmach 'clt.i ' 1>>'cltdata
  i=i+1
end
```

Figure 56. Gathering Client Information - Sample REXX Script

---

## 6.5 Printing

WorkSpace On-Demand provides for the use of local printers, networked printers, or both, from the same client workstation. This section explains the steps that you must carry out before defining a network printer for access by a WorkSpace On-Demand client, and explains what happens when you define a local printer as part of the client workstation definition.

### 6.5.1 Network Printing

To use a network printer from a WorkSpace On-Demand client, you must ensure that the printer is shared on the network, and that the users of this client workstation have access to it. You must also ensure that you have created a printer object for the printer on your server. When you have carried out these tasks, you can complete the steps described in Section 6.1.1.4, "Defining Printers" on page 143.

The WorkSpace On-Demand shell allows the user to manipulate the printer object on the desktop by means of mouse button 2 and the context menu. For example, a user who has access to several printer objects can choose which is to be the default. Users can also see details of, or cancel, their own print jobs.

### 6.5.2 Local Printing

Defining a local printer in the Client Definition notebook causes a local printer icon to appear on the client's desktop. The setup utility may prompt you to install additional drivers for the printer, if no printers of that type have previously been defined on that server.

This will create a directory for the printer driver in the \OS2\DRIVERS subdirectory of the \BMLAN\RPL\BB20.US subdirectory. For example, the driver for a printer that emulates an HP Laserjet will go in the subdirectory \BMLAN\RPL\BB20.US\OS2\DRIVERS\LASERJET. You should bear this structure in mind when installing updated versions of printer drivers on the WorkSpace On-Demand server.





## Chapter 7. Working with Network Applications

This chapter explains how to install applications in your OS/2 Warp Server domain to be accessed on the WorkSpace On Demand 2.0 environment by the network users, how to define them as network public applications, and how to grant access to applications for your users.

### 7.1 Application Structure Under Workspace on Demand

In Section 2.1.1, "What is an Operating System?" on page 39, we mentioned that an operating system can be regarded as a collection of files that must be loaded into a client workstation's memory and executed. The same can be said for an application; it is simply a set of files that contain the application code and associated configuration information that must be loaded into memory and executed.

The files that comprise an application are similar in nature to those that comprise an operating system. However, an application introduces an additional level of complication since we must not only deal with configuration data that is specific to a client workstation but also that which is specific to an individual user. Table 18 shows the types of files that comprise an application, along with some examples.

Table 18. File Types - Application Software

File Type	Read-Only	Read/Write
Generic	95% of application code	Not Applicable
Client-Specific	SNA network configuration	Temporary work files
User-Specific	Lotus Notes ID file	Lotus Notes desktop

It is a simple process to load an application into a client's memory from local storage since each client has its own copy of the application software. In most cases, each user has a unique copy of the configuration files, provided that the client is not shared among multiple users.

When you load an application from a server, however, the process becomes more complicated. A single copy of the application code is now shared between multiple clients and multiple users. Some of the files can be shared, but others are unique to a particular client or user. The server must store multiple copies of such files and allow each client and user to access the correct files. An application's files must therefore be divided into different groups and placed in different locations on the server.

However, most applications' installation programs assume that you are installing the application onto a local drive at the client workstation from which it will be run. They copy files into a single directory structure on a single logical drive. This means that you cannot simply run the installation program on the server and immediately access the application from a client.

When setting up a network application, you must perform a number of steps to ensure that clients and users can correctly access the application's files:

1. Determine the required file structures for the application on the server.
2. Copy the application's files into these file structures.
3. Ensure that file I/O requests are correctly redirected to the server.

In order to successfully carry out these steps, you will need to understand the structure of the OS/2 Warp Server and WorkSpace On-Demand server environment, as well as the behavior of your application itself. There are a number of tools that can help you in analyzing your application's behavior, and these are discussed at the appropriate places in the text.

### 7.1.1 Application Types

Before starting to set up an application for access by WorkSpace On-Demand clients, you should understand the basic type of application with which you are working. Applications can be split into three basic types:

- **Client-specific applications** are typically "middleware" applications that are launched during the client workstation's boot sequence. They are composed of generic application code, with certain configuration data that is unique to each client workstation, such as network addresses, SNA PU IDs and so on. An example of such an application is IBM Personal Communications/3270 Access Feature.
- **User-specific applications** are typically launched by the end user from the client's desktop. They are composed of generic application code with certain configuration data that is unique to each end user. An example of such an application is Lotus Notes.
- **Hybrid applications** exhibit characteristics of both the client- and user-specific categories. These applications include certain code that is launched during the client's boot sequence as well as individual application components that are launched by the end user. An example of such an application is Lotus SmartSuite for OS/2 Warp 4.

The type of application you are installing will affect the types of files within the application and the file structures into which they must be placed.

## 7.1.2 Generating Directory Structures

As illustrated in Table 18 on page 167, there are three basic types of files used by an application:

- Generic code includes programs, dynamic link libraries or basic configuration files that are shared by all users of an application, and all clients on which the application is run. These files are always accessed in read-only mode by an application running on a client.
- Client-specific files are configuration files that define a client's SNA address and so on. These files are unique to a client workstation, but are still shared by those end users who use that client. In most cases, these files will be accessed in read-only mode, but in some circumstances, a client-specific file may require write access by the application.
- User-specific files are configuration files that define an individual user's working environment, mail preferences and so on. These files are unique to each end user. In most cases, these files will be accessed in read/write mode to enable users to customize their working environment

The WorkSpace On-Demand server provides a default set of directory structures in which you can store these different types of files. Figure 57 shows the directory structures that are available on the WorkSpace On-Demand server.

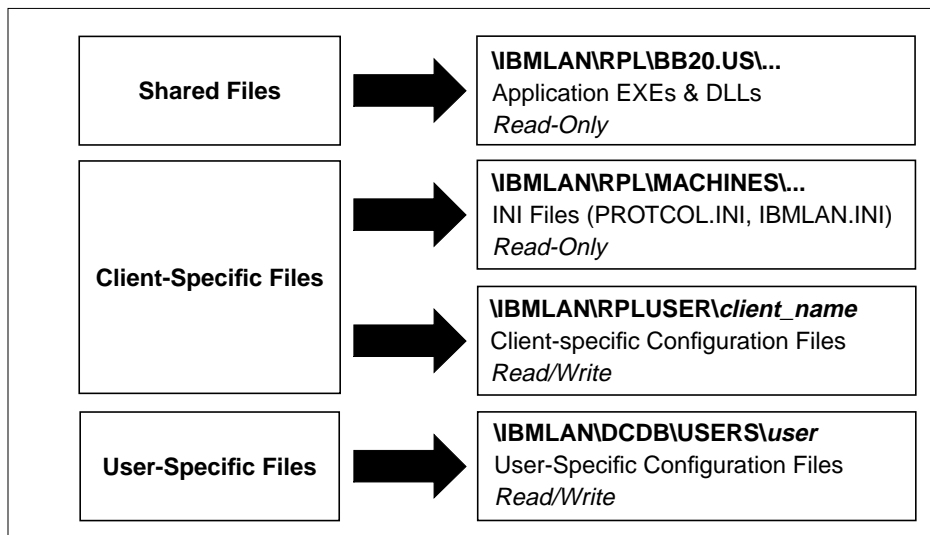


Figure 57. Application Directory Structures

### 7.1.2.1 Shared Files

Generic shared files, to which the client workstation requires only read access, can be located in one of two places on the server:

- In a subdirectory within the \BMLAN\RPL\BB20.US directory tree. The application's directory is thus treated as part of the client's boot-drive, and is immediately accessible to the client both at boot-time and after user logon. This is the preferred method of sharing application files and indeed is the only method by which you can easily access application files during the boot sequence.
- In a shared directory to which the client workstation and/or end user is granted access. This works satisfactorily for applications which do not require device drivers or other modules to be initialized at boot-time, but requires more work to define and grant access to users. All the applications can share the same alias in order to optimize network resources.

We recommend that you create a subdirectory for your application's shared files within the \BMLAN\RPL\BB20.US directory tree. This directory tree contains the operating system image and is the best location for shared application files that are common to all clients and users. It is available to all clients at boot-time as part of the RPLFILES alias.

Many middleware applications require device drivers or other application code to be loaded during the boot sequence. You may therefore need to add statements to your client's CONFIG.SYS file.

### 7.1.2.2 Client-Specific Application Files

With many applications, certain configuration files are specific to the individual client. For example, the Personal Communications/3270 Access Feature includes configuration files containing SNA physical and logical unit IDs that are unique to each client.

The WorkSpace On-Demand server provides two directory structures in which you can place such files:

- The \BMLAN\RPL\MACHINES directory tree contains operating system and application files specific to a particular client and to which the client requires only read access. This structure is available to all clients at boot time as part of the RPLFILES alias.
- The \BMLAN\RPLUSER directory tree contains operating system and application files to which the client requires read/write access, such as log files. This file structure is available to all clients at boot time as the WRKFILES alias.

Each client has its own unique subdirectory within each directory tree. You can place client-specific files within these subdirectories using the `\BMLAN\RPL\MACHINES\client_name` subdirectory for read-only files and the `\BMLAN\RPLUSER\client_name` subdirectory for those files that require write access.

When deploying WorkSpace On-Demand in a large-scale production environment, you may wish to automate the creation of these client-specific files using the following technique:

- Place a default version of each client-specific configuration file in the `\BMLAN\RPLUSER\BB20.US` directory structure. WorkSpace On-Demand automatically replicates this directory structure, including subdirectories and files, to the `\BMLAN\RPLUSER\client_name` directory for each client when the client is defined.
- Use REXX command files to modify the contents of each client-specific configuration file and add the required unique entries for each client at client definition time. The information for these entries must, of course, be gathered as part of the preinstallation planning performed prior to deploying WorkSpace On-Demand.

See Chapter 11, “Deploying WorkSpace On-Demand” on page 297, for more information on deployment techniques.

### **7.1.2.3 User-Specific Application Files**

Some applications may incorporate files that are specific to an individual user. For example, Lotus Notes includes files, such as a user’s ID and `DESKTOP.DSK` files, that define the working environment for a specific user rather than for the client workstation itself.

In a WorkSpace On-Demand environment, you must place such user-specific files in a location on the server from which they can be accessed after the user logs on. This location will of course be different for each user. We recommend that you place these files in the `\BMLAN\DCDB\USERS` directory structure. This directory is created on the domain controller when you define a user ID, and the user has read/write access to the files in the directory.

Note that this directory structure is located on the domain controller for your OS/2 Warp Server domain. In a single-server environment, this is the same physical machine as your WorkSpace On-Demand server. However, if you define your WorkSpace On-Demand server as an additional server in your OS/2 Warp Server domain, the files in this directory structure will reside on a different physical machine than the shared and client-specific files.

### 7.1.3 Copying Files to the Directory Structures

When you have created the directory structures necessary to support your application's files on the server, you must copy the files to the correct locations within these directory structures. Determining which files must be placed in which locations, and ensuring that each file resides in the correct location, is a critical element in setting up a network application.

At this point, you will need to understand the behavior of your application and the files that it requires in order to load and run correctly. We have developed a methodology that will help you to investigate your application's behavior and place its files in the correct locations on the server. Section 7.2, "Installing an Application on the Server" on page 174, describes this methodology in detail.

### 7.1.4 Redirecting File I/O Requests

Chapter 3, "Understanding File Index Tables" on page 85, discusses the concept of the file index table (FIT). The WorkSpace On-Demand client operating system maintains a FIT in memory for use by applications and the operating system itself. When an application running on a client requests access to a file on the server, the client operating system uses the FIT to redirect file I/O requests to the correct location on the server. Figure 58 shows the process by which this redirection occurs.

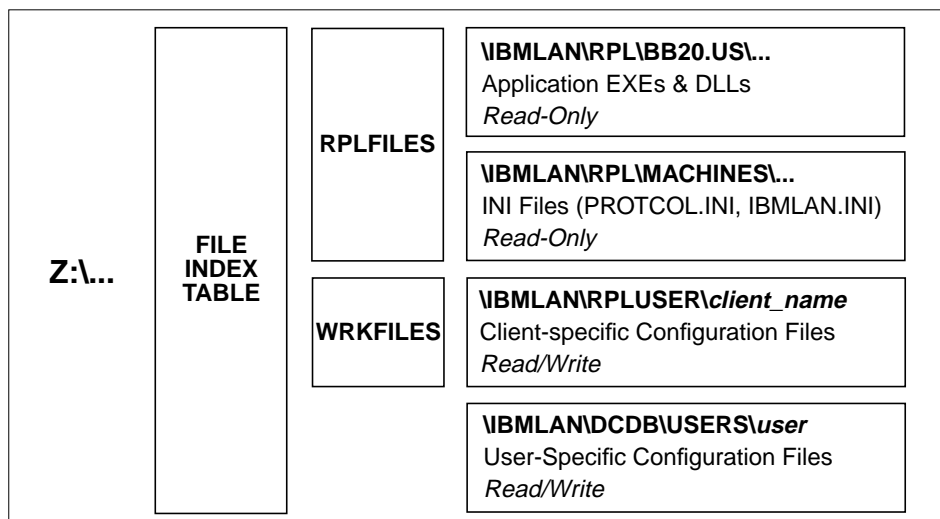


Figure 58. Redirecting Application File I/O Requests

In most cases, the client will regard all application files as residing on its boot drive. The client operating system uses the FIT to resolve the directory and

file name requested by the application to an alias, directory and file name on the server, and requests that file from the server over the network.

When you place application files in the directory structures on the WorkSpace On-Demand server, you must ensure that the FIT used by the client includes redirection statements for these files. You can place these statements in either of two areas:

- For client-specific applications that are launched during the client's boot sequence, place the redirection statements in the machine FIT file.
- For user-specific applications that are launched by the end user, place the redirection statements in the application FIT file.

For client-specific and user-specific files, you can use the variables allowed in the application FIT file to redirect file I/O requests to a specific client's or user's directory structures. See Section 3.1.2, "Using Variables in FIT Files" on page 87, for more information.

### 7.1.5 Access Control Profiles

When placing files on the server, you must ensure that the client and/or the user has the necessary permissions to access the files. Remember that the client workstation itself has a user ID created when you define the client, and this user ID is used to authenticate all file access requests until an end user logs on to the client.

After an end user logs on to the client, the end user's own user ID is used to authenticate access requests for resources that *have not* already been accessed by the client prior to logon.

*Table 19. Application/File Types - Access Control Profiles*

Application Type	File Type	Access Control Profile
Client-Specific	Generic	Client
	Client-specific	Client
User-Specific	Generic	User
	User-specific	User
Hybrid	Generic	Either
	Client-specific	Client
	User-specific	User

In many applications, this does not present a problem since all the application's files are accessed either before or after logon. However, some applications exhibit characteristics of both the client-specific and user-specific types; these are the hybrid applications shown in Table 19. These applications may access certain files at boot-time, before a user logs on, and other files after a user logs on. Lotus SmartSuite for OS/2 Warp 4 is an example of such an application.

It is important to understand that if a network alias is first accessed prior to user logon, the client's user ID and access control profile will be used for *all* file I/O requests to that alias, both before *and after* user logon. If an alias is not accessed until after user logon, the user's user ID and access control profile will be used to authenticate all file I/O requests for that alias. You must therefore ensure that you organize your FIT file entries, and the aliases to which they redirect requests, in such a way that the application can access its files on the server using the correct access control profile.

---

## 7.2 Installing an Application on the Server

As discussed in Section 7.1, "Application Structure Under Workspace on Demand" on page 167, you cannot simply install an application on your WorkSpace On-Demand server and expect it to be immediately accessible to WorkSpace On-Demand clients. You must alter the directory structures and other environment settings created by the application's installation program, to reflect the fact that the application will be loaded from the server but executed on the clients. The following sections show you how to determine the changes that are required and how to apply these changes to your application's environment.

### 7.2.1 Hardware Requirements

In order to install and test your application in a WorkSpace On-Demand environment, you need to provide a test hardware environment as shown in Figure 59.

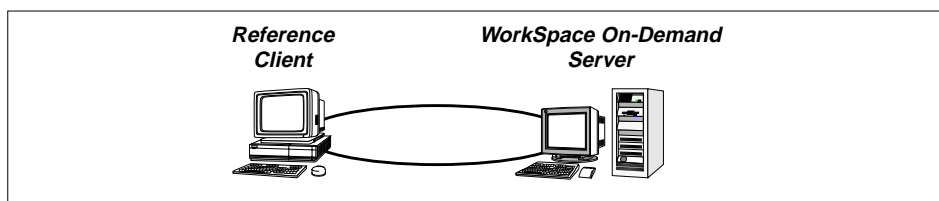


Figure 59. Creating a Machine Class - Test Hardware Environment



First, you will require a WorkSpace On-Demand server. This is required in all cases, regardless of the type of application you wish to install.

For testing, and in some cases for installation too, you will need a client, known as a *reference client*. Some applications will require you to install the application on the reference client, then transfer the files to the correct locations on the server. For this reason, the reference client should include the following:

- A BIOS that allows it to boot from the network as well as from local storage devices.
- A network adapter that is supported by WorkSpace On-Demand 2.0 or for which you have already generated the appropriate definitions as described in Chapter 9, "Supporting Network Adapters" on page 241.
- Its own local disk drive, with sufficient disk space to install OS/2 Warp 4.

When creating and testing your application definitions, you will use the reference client as both a WorkSpace On-Demand client and a traditional "fat" client in stand-alone mode.

### 7.2.2 Create a Temporary Environment

Create a temporary environment on the server using the MKTMPENV utility provided with WorkSpace On-Demand 2.0. Issue the following command:

```
MKTMPENV client_name environment_name /ADD
```

where *client\_name* is the name of the client, and *environment\_name* is any name not beginning with BB or OS2.

The MKTMPENV utility creates copies of several directory structures as shown in Table 20. These three directory trees represent the three basic ways in which an application may share its files, as discussed in Section 7.1.2, "Generating Directory Structures" on page 169.

Table 20. MKTMPENV Utility Directory Copies

Original	Copy
\BMLAN\RPL\BB20.US	\BMLAN\RPL\ <i>environment_name</i>
\BMLAN\MACHINES\ <i>client_name</i>	\BMLAN\MACHINES\ <i>client_name</i> .BAK
\BMLAN\RPLUSER\ <i>client_name</i>	\BMLAN\RPLUSER\ <i>client_name</i> .BAK

This process may take several minutes. After these directories are copied, you can install the application into the copies on the server

Note that certain applications do not install correctly on a server using the backup directory trees created by the MKTMPENV utility. In such cases, you may need to boot your reference client using OS/2 Warp 4, and install the application on the reference client's local hard disk. The reference client allows you to install the application in the standard way and analyze the additions and changes made to the reference client's file structures by the installation process. These detailed differences will allow you to transfer the application's files to the correct locations on the server, and configure the application to be used by users on WorkSpace On-Demand clients.

### 7.2.3 Install the Application

The means by which you install the application will differ depending on whether your application allows you to use the MKTMPENV utility or whether you are forced to use a reference client. Each method is explained in more detail below.

#### 7.2.3.1 Using MKTMPENV

Boot the client and logon using an existing user ID. To avoid any problems with access control profiles at this time, we recommend that you use a user ID with administrator authority.

Install the application on the boot drive (usually Z:) following the application installation instructions. If possible, do not configure any client-specific or user-specific features.

When you have installed the application, run the DIFFTREE utility (provided with WorkSpace On-Demand 2.0) from an OS/2 command line on the server, using the following command:

```
DIFFTREE source_tree target_tree
```

You will need to run the DIFFTREE utility three times (once against each of the directory trees copied by the MKTMPENV utility):

The DIFFTREE report shows the files that were added or changed when you installed the application. With sufficient knowledge of the application, you should be able to place the read-only files and read/write files in the correct locations. At this stage, most files will be read-only. You should copy these files to the proper areas within the \IBMLAN\RPL\BB20.US directory tree.

#### Note

There are other tools that allow you to compare directory trees. One tool we would recommend is PMDMatch 1.09, a graphical Presentation Manager application that allows you to view differences between directory trees at a glance. We have included this tool on the CD-ROM that accompanies this redbook.

Configure the application to include your required settings for both client-specific and user-specific information and generate a new DIFFTREE report. This report should now reflect the changes made during the configuration process.

Most files changed at this point will be client- or user-specific. You must decide whether you wish these files to be stored in a read-only form, thereby preventing the user from changing them, or in a read/write area, where the user may change them as required. You should place read-only files in the \BMLAN\MACHINES\*client\_name* directory and read/write files in the \BMLAN\RPLUSER\*client\_name* directory.

#### 7.2.3.2 Using a Reference Client

If you are using a reference client, you must capture the client's system state, both before and after installing the application. This will allow you to determine which files and directories have been added or changed during installation.

Before starting the installation, turn off the archive attribute bit for all files on the reference client using the following command from the root directory of the client's boot drive:

```
ATTRIB *.* -A /S
```

This allows you to later determine which files are accessed during the application installation process, by querying the archive attribute.

Make copies of existing system configuration files, including:

```
CONFIG.SYS  
OS2.INI  
OS2SYS.INI  
WIN.INI  
SYSTEM.INI
```

The installation process may modify these files, and keeping copies allows you to determine where changes have been made.

Install and configure the application on the reference client according to the application's installation instructions.

When the installation is complete, search for all added or modified files on the client by issuing the following command:

```
DIR *.* /A:A /S >ATTRIB.TXT
```

This command scans the client's hard drive for all files that have the archive attribute set—that is, all files that have been opened or modified during the installation process—and writes the resulting directory listing to a file named ATTRIB.TXT.

#### 7.2.4 Identify Client- and User-Specific Files

It is very important that you understand the application's behavior in order to determine which files and subdirectories require only read access and which require read/write access. Typically, application code can be made read-only and shared by all clients and users, but client- and user-specific files must be separated into client- and user-specific subdirectories and accessed in read/write mode.

At this point, you will require some knowledge of the way in which your application operates since you must now determine which application files can be shared by all client workstations and/or users and which files are specific to a particular client or user. Generic files can be located within the WorkSpace On-Demand boot image and shared by all clients and users, but client- or user-specific files require separate copies to be made and placed in different directories. Generic application files that can be shared by all users are normally located within the client's boot image on the server—that is, within the \IBMLAN\RPL\BB20.US directory tree.

For the files that are specific to a client or user, you must identify those files that are specific to a client and determine whether each file can be accessed in read-only mode or whether it requires write access during application execution. Client-specific, read-only files can be placed in the \IBMLAN\RPL\MACHINES\*client\_name* directory, but client-specific, read/write files should be placed in the \IBMLAN\RPLUSER\*client\_name* directory, where *client\_name* corresponds to the name of the client.

For the user-specific files, you must perform a similar task to determine which files can be accessed in read-only mode and which files require write access.

You must create your own directories to contain user-specific, read-only files since WorkSpace On-Demand does not provide a directory for such files. You can place user-specific read/write files in the \IBMLAN\DCDB\USERS\*user\_id* directory on the domain controller. In most cases, however, you can simply place all user-specific files in the \IBMLAN\DCDB\USERS\*user\_id* directory on the domain controller.

## 7.2.5 Copy Application Code to the Server

Next you must translate the application's directories on the server or the reference client to your current WorkSpace On-Demand client boot image. The steps you must follow will differ depending on whether your application allowed you to use the MKTMPENV utility or whether you were forced to use a reference client.

### 7.2.5.1 Using MKTMPENV

If you are installing from a temporary environment that you created on the server using the MKTMPENV utility, XCOPY the contents of the \IBMLAN\RPL\*environment\_name*\*appl\_name* directory to a new directory named \IBMLAN\RPL\BB20.US\*appl\_name*, where *appl\_name* is the base directory for the application (for example, NOTES for Lotus Notes or NETSCAPE for Netscape Navigator). You should ensure that the subdirectories are also copied.

### 7.2.5.2 Using a Reference Client

If you installed the application on a reference client, you should create a directory tree within the \IBMLAN\RPL\BB20.US directory corresponding to the application's directory tree on the reference client. You can use a compression tool such as Infozip's PKZIP.EXE to create a single file containing the application's entire directory structure. You can then move this file to the server and unzip it into the \IBMLAN\RPL\BB20.US directory.

## 7.2.6 Copy Client- and User-Specific Files to the Server

You must now copy any client- and user-specific files to the appropriate locations on the server.

- Copy any client-specific or user-specific read-only files to the \IBMLAN\RPL\MACHINES\*client\_name* directory. In cases where these files contain unique settings for each client, you will need to devise a method of generating each file with the unique settings and placing it in the correct directory for the client to which it belongs.
- Within the \IBMLAN\RPLUSER\*client\_name* directory tree, create an empty directory for the application and also for any subdirectories of that

application directory that will be mapped using FIT entries. These directories will contain client-specific files that are updated by the application during execution. In some cases, these files may not be created until the application is run; so the directories must be created even though they are left empty. This is another instance where knowledge of the application's behavior becomes very important.

- Within the `\BMLAN\DCDB\USERS\user_name` directory, create an empty directory for the application. You should then copy any required configuration files to this directory. For example, you might copy the user's initial Lotus Notes ID and DESKTOP.DSK files here so that these files can be accessed by redirection through the FIT and modified as required by the user.

The exact directory structures you use on your server will depend upon your own preferences and the requirements of the application, since some applications require a specific directory structure for their files.

### 7.2.7 Modify Operating System Configuration Files

Some applications, particularly middleware applications, will modify operating system configuration files such as `CONFIG.SYS` and possibly even `OS2.INI` or `OS2SYS.INI`. Windows applications commonly make changes to the `WIN.INI` and `SYSTEM.INI` files. You must ensure that these changes are reflected in the corresponding files within the client's boot image.

Note the changes made to the system configuration files by the application installation process, and make the necessary additions or modifications to the corresponding files in the client's boot image. While each client has its own unique set of system configuration files, which are located in the `\BMLAN\RPL\MACHINES\client_name` directory, it is more efficient to make the changes to the template files for a client's machine class, which are located in the `\BMLAN\RPL\MACHINES\BB20.US\machine_class.MC` directory. Any changes that you make to this file are automatically incorporated into each new client's configuration files when you define the client.

Some applications may modify binary files such as `OS2.INI`. In these cases, you must use a utility, such as `INIDIFF.CMD`, to determine the changes made to the INI files, and then use the `INIMERGE.CMD` utility and possibly an INI file editor, such as `FM/2`, to apply these changes to the INI files in the client's boot image. These utilities are included on the CD-ROM that accompanies this redbook.

In WorkSpace On-Demand 2.0, you can define certain INI file entries dynamically using the **Parameters** page in the **Application Definition** notebook when you define create a network application definition. See Section 7.8, “Customizing INI Files” on page 203, for more information on how to do this.

The method described above incorporates the changes into the configuration files for every new client that you define using a particular machine class. If you wish to run an application only on certain clients, and therefore want the necessary modifications to be included only for those clients, you may wish to create a new machine class and make the modifications to the templates for that machine class only. See Chapter 10, “Supporting Additional Hardware” on page 251, for more information on machine classes.

Note that the more application code you include in your client's boot image to be loaded at boot-time, the larger the working set of your client becomes. Since this can affect the boot performance of your clients, you need to find the right balance between loading code at boot-time or alternatively, waiting until an application is needed before loading its modules. This balance will depend heavily on the types of applications that you use in your particular installation.

### 7.2.8 Redirect Required Files

You must ensure that all application files are addressed by entries in the client's file index table (FIT). You can place the required entries in the machine, user or application FIT files, depending on the time at which the file will be accessed.

Any files used by middleware applications that will be accessed at boot time should be placed in the machine FIT file, irrespective of whether they are generic, client- or user-specific files, since the machine FIT file is loaded into the client's memory during the boot process. Note that you must place these files in a location on the server to which the client has the appropriate access using the client's own access control profile.

Application files that are accessed by applications launched from the desktop by the end user should be placed in the application FIT file. You can use the variables available under WorkSpace On-Demand 2.0 to tailor a single application FIT file to support multiple clients and users. These variables are:

- ? is resolved as the client's boot drive (typically Z:).
- <DCSERVER> is resolved as the OS/2 Warp Server domain controller.
- <USER> is resolved as the user name.

- <RPLBOOTSERVER> is resolved as boot server.
- <OS2VERSION> is resolved as the current OS/2 version being booted on the client workstation. This enables for different OS/2 versions to be managed on different clients by the same boot server.
- <MACHINE> is resolved as the client workstation name.

Note that the user FIT file is not widely used under WorkSpace On-Demand 2.0 since the application FIT file performs many of the same functions and allows you to load and discard FIT entries when starting and stopping an application, thereby saving memory space in the in-memory FIT.

You may wish to use a user FIT file, however, if you have a hybrid application that is loaded at boot-time and therefore has entries in the machine FIT, but also requires access to user-specific files. Since entries in the user FIT override similar entries in the machine FIT, you can use the user FIT to redirect certain files to the correct, user-specific locations.

See Chapter 3, “Understanding File Index Tables” on page 85, for a more complete discussion of FIT files and the variables they may contain.

### 7.2.9 Test the Application Definition

Test the network application by running it from a different client workstation. Ensure that the application runs correctly and that all configuration files can be accessed and, where necessary, modified. For user-specific applications, you should also log on using several different user IDs, and ensure that the user-specific settings are loaded correctly by the application.

### 7.2.10 Clean Up the Installation Environment

Delete the NetBIOS sessions between the server and the client by issuing the following command from the server’s command line with administrator privilege:

```
NET SESS \\client_name /DEL
```

If you used the MKTMPENV utility to create a temporary environment on the server, delete the temporary environment by issuing the command:

```
MKTMPENV client_name /DELETE
```

---

## 7.3 Defining the Application Using the GUI Interface

Defining an application for use by a WorkSpace On-Demand client is much the same as setting up a network public application with OS/2 Warp Server.



However, WorkSpace On-Demand adds a number of enhancements to the public application facility. These enhancements allow more environment information to be saved with the application, thus allowing for application roaming (see Section 7.9, "Application Roaming" on page 206).

The example we use here is an OS/2 application named PMCAMERA. This application consists of the following files:

PMCAM200.DLL  
PMCAM200.EXE  
PMCAM200.HLP  
PMCAM200.INI

This is a very simple application, but it illustrates the principles of application definition quite well since it contains several different file types.

You must first decide where to install your application on the server. We recommend installing your applications within the \IBMLAN\RPL\BB20.US directory tree. For this example, we will create a subdirectory named PMCAMERA within the \IBMLAN\RPL\BB20.US directory. We will then create another directory named PMCAMERA\DLL for the DLL files, and a third directory named PMCAMERA\HLP for help files.

Installing PMCAMERA is very easy; you must simply XCOPY the application files to their locations:

```
XCOPY PMCAM200.DLL C:\PMCAMERA\DLL
XCOPY PMCAM200.EXE C:\PMCAMERA
XCOPY PMCAM200.HLP C:\PMCAMERA\HLP
XCOPY PMCAM200.INI C:\PMCAMERA
```

Begin by opening the **Public Application Definitions** folder for your OS/2 Warp Server domain, as shown in Figure 60.

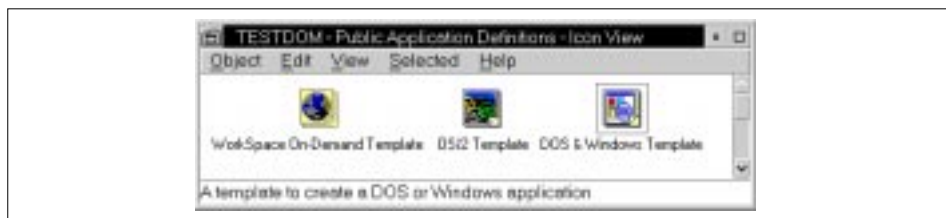


Figure 60. Public Application Definitions Folder

### 7.3.1 Identify the Application

In the Public Applications Definitions folder, drag and drop a **WorkSpace On-Demand Template** to a free area of the folder. The OS/2 Warp Server GUI displays the **Identity** page of the Application Definition notebook, as shown in Figure 61.

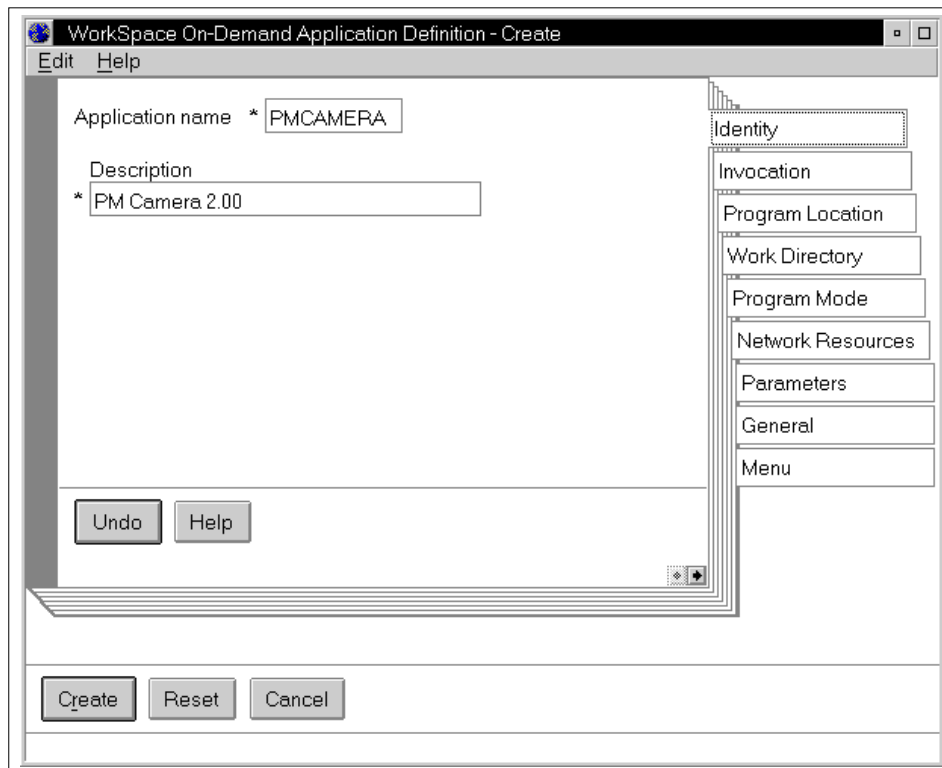


Figure 61. Application Definition - Create Settings Notebook

On this page, enter a name for the application in the **Application name** field, and enter a description in the **Description** field.

### 7.3.2 Specify the Program Name

On the **Invocation** page, specify the command used to launch the application. In our PMCAMERA example, the command is PMCAM200.EXE. You can also specify additional parameters on this page, or instruct the client to prompt the user for additional parameters when starting the application.

### 7.3.3 Specify the Program Location

On the **Program Location** page, you need to specify the **Alias** directory that you had defined earlier for the application. The path to the application's main executable also needs to be specified, as shown in Figure 62.

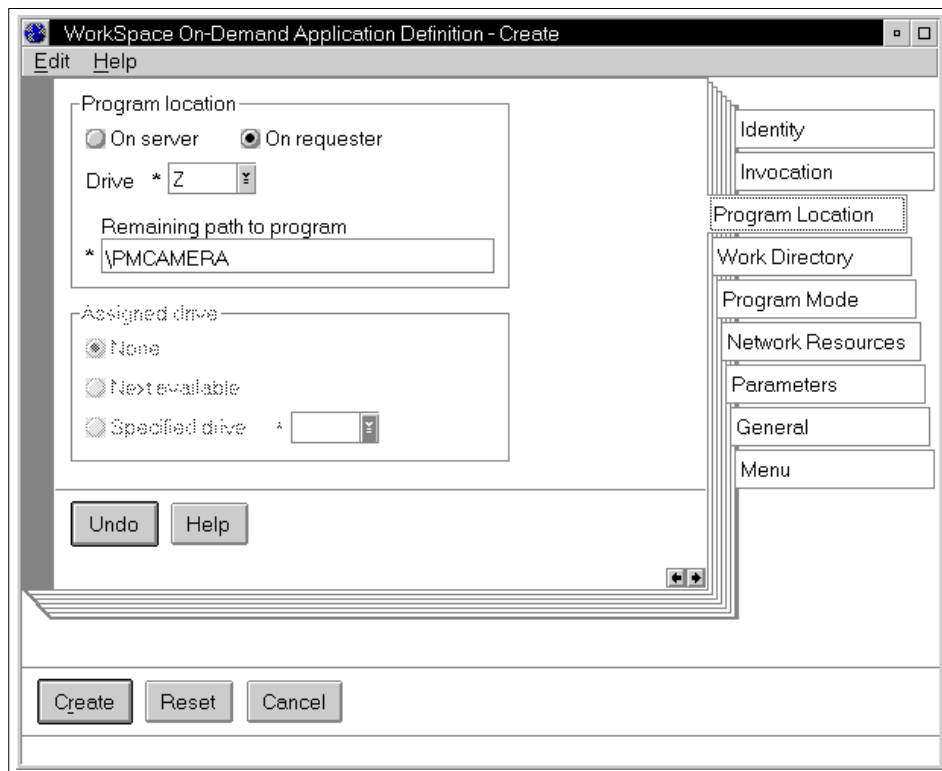


Figure 62. Application Definition - Program Location Page

You can see from the example in Figure 62 that we have selected the **On requester** radio button. This is because, even though the application physically resides on the server, we will access it from the client workstation's viewpoint as though it resides on the client's boot drive (Z:). This is the preferred method of accessing network applications since it reduces the number of network aliases, which are potential drive letter assignments, required on the server.

If you wish, however, you can select the **On server** radio button. The OS/2 Warp Server GUI then replaces the **Drive** field with an **Alias** field into which you can enter the name of the network alias on which the application resides.

Most applications do not require a drive letter assignment to the directory alias where the program resides. The default is not to assign a drive on the **Program Location** page. However, some application programs require a drive assignment and will not start correctly unless one is assigned. To avoid this situation, you can assign a drive letter to the directory alias as a current assignment and specify this drive.

### 7.3.4 Assign a Work Directory

Some programs require a work directory to store temporary files and other working data. You can assign a working directory using the **Work Directory** page, as shown in Figure 63.

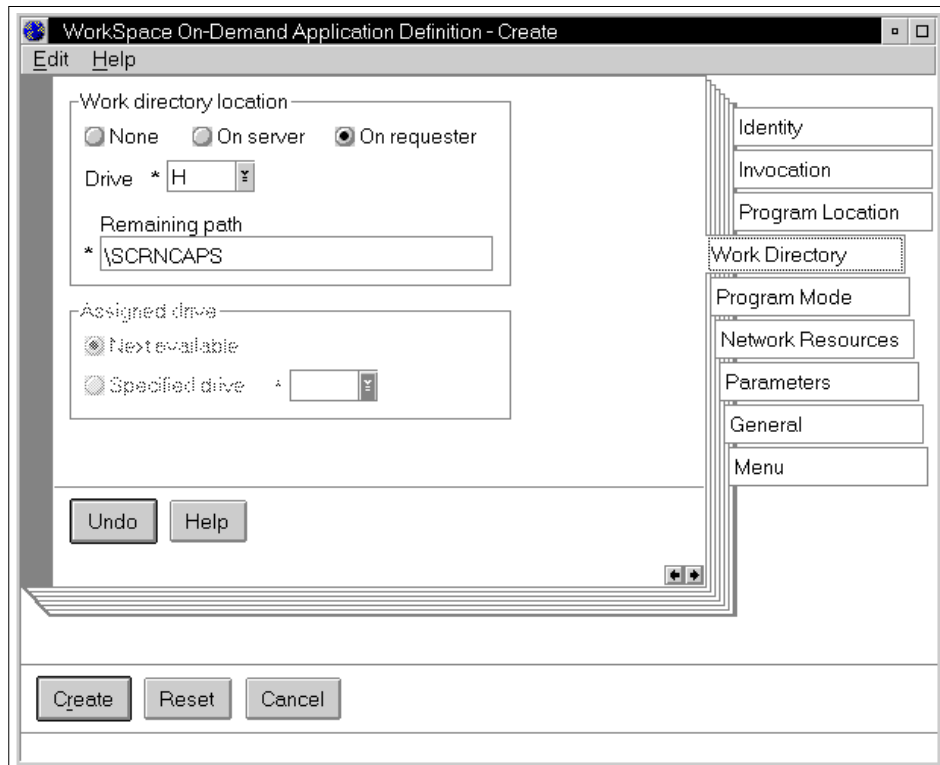


Figure 63. Application Definition - Work Directory Page

Here we have specified the work directory to be the user's home directory, which is accessed as drive H: from the client's viewpoint. This example assumes that you have defined a home directory for each user who will run PMCAMERA and that this directory is accessed as drive H: by the user.

### 7.3.5 Specify the Program Mode

On the **Program Mode** page, you need to specify what type of application you are defining. In our PMCAMERA example, the application is an OS/2 Presentation Manager (OS/2 PM) application.

### 7.3.6 Allocate Network Resources

Figure 64 shows the **Network Resources** page. Using this page, you can dynamically assign network resources, such as printers, shared serial ports or additional directory aliases, when the application is started by the user. Note that the user must have the appropriate level of access to the resource.

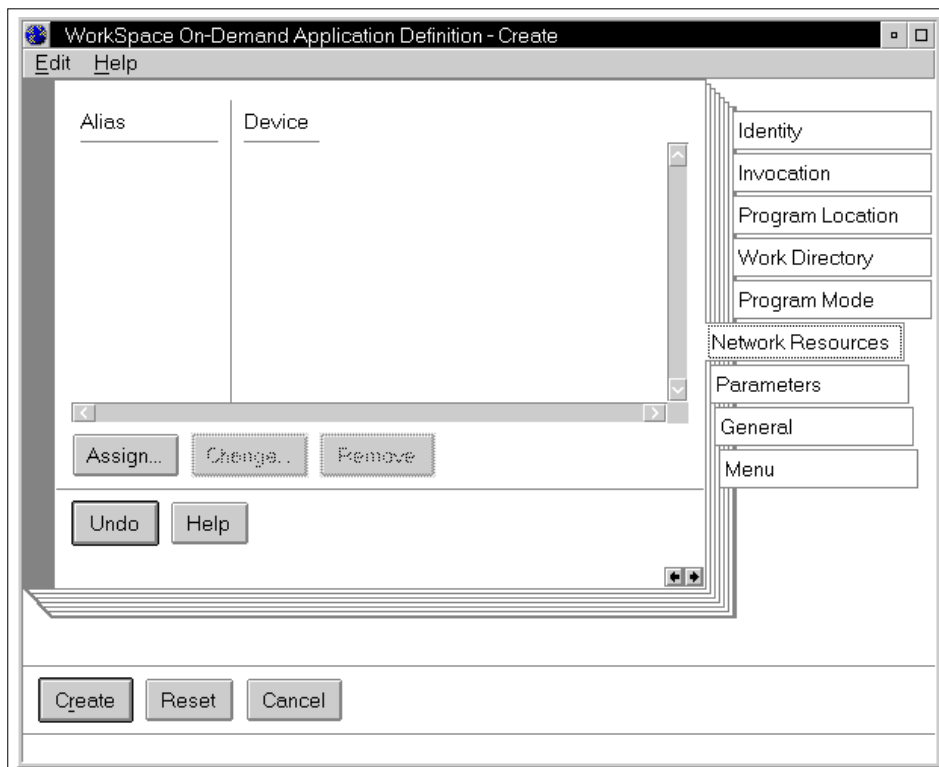


Figure 64. Application Definition - Network Resources

In our PMCAMERA example, we do not require any additional network resources; so the **Network Resources** page is left blank.

### 7.3.7 Define Required Environment Variables

The **Parameters** page allows you to define the application's execution environment dynamically when the application is started. This page is shown in Figure 65.

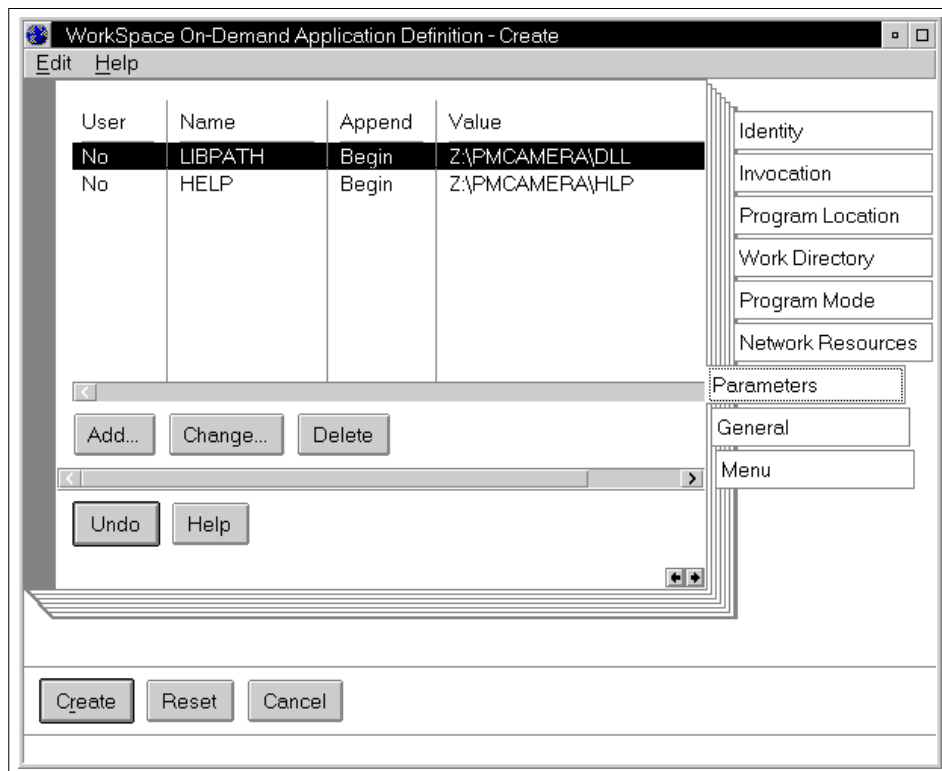


Figure 65. Application Definition - Parameters

The **Parameters** page displays the environment variables including their names, the location where the value is appended to the variable, the value, and whether the parameter is user-specific. From here, you can add, change or delete parameters. You can also read a set of parameters from a file instead of manually entering them each time.

#### 7.3.7.1 Adding Parameters

To add an application parameter:

1. Select the **Add** push button. The Add Application Parameters pop-up is displayed, as shown in Figure 66.

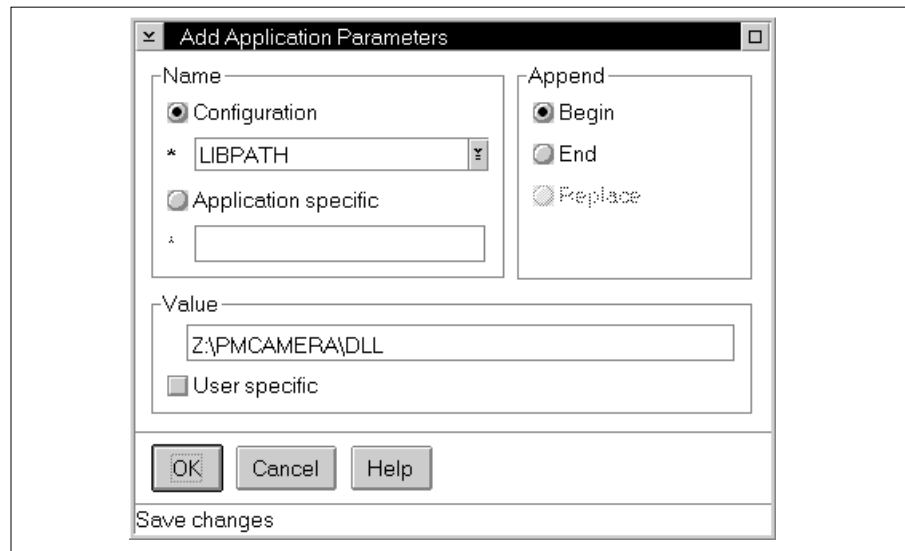


Figure 66. Application Definition - Adding Configuration Parameters

2. Choose either the **Configuration** radio button to display a drop-down list of known environment variables from which to select an item, or choose the **Application specific** radio button and enter an application-specific environment variable name.
3. Choose the position for appending the environment parameters. Select **Begin** to place the parameter value at the beginning of a currently existing environment variable of the same name. Select **End** to place the parameter value at the end of a currently existing environment variable of the same name. Select **Replace** to replace an existing environment variable with the value you specify.

You can specify a parameter name more than once if it is defined with a different append value. Parameters specified more than once that have different append values are processed in the following order:

1. REPLACE value
2. BEGIN value
3. END value

If you repeat a parameter name with the same append value, an error message is displayed that instructs you to correct the problem before adding that parameter.

You can specify most environment variables as application parameters; however, `LIBPATH` is an exception. `LIBPATH` cannot be changed or replaced.

`LIBPATH` can only be specified with the `BEGIN` or `END` append flag to allow adding to the front or back of the existing `LIBPATH`. Additionally, `BEGINLIBPATH` produces the same results as specifying `LIBPATH` with the `BEGIN` append flag, and `ENDLIBPATH` produces the same results as specifying `LIBPATH` with the `END` append flag. See the *IBM OS/2 Command Reference* for more information on using `LIBPATH`.

#### Hint

In order to launch an application more quickly, you can specify `PATH` and `LIBPATH` parameters using the **Begin** radio button. In this way, the system will find the required application files without needing to search through the operating system directories first.

4. Enter the value of the parameter in the **Value** field.
5. In our `PMCAMERA` example, the following parameters must be added:
  - `LIBPATH, Begin, \\WSODSRV1\OS2APPS\DLL`
  - `HELP, ?, \\WSODSRV1\OS2APPS\HLP`

#### User Settings

In the `PMCAMERA` example, we assume that all users will use a common INI file. If you wish each user to maintain a separate INI file, you must include an entry for the INI file in the application FIT file, and use the `<USER>` variable to redirect the file access to a separate user-specific directory.

6. Select the **User specific** checkbox if you wish this parameter to be overridden for each user. In such cases, the value that you specify here becomes the default. See Section 7.7.3, "User-Specific Application Parameters" on page 202, for information on how to override the default value.
7. Select **OK** to add the parameter.

The Add Application Parameters pop-up will disappear, and the new parameter you have added will appear in the list on the Parameters page in the Application Definition notebook.

If you wish to add another parameter, select the **Add** button again. Similarly, you can modify or delete an existing parameter by selecting the **Change...** or **Delete** buttons, respectively.



### 7.3.7.2 Parameter Types

There are two basic types of parameters that you can select using the Add Application Parameters pop-up:

- Configuration parameters are the environment variables `PATH`, `LIBPATH`, `DPATH`, and `HELPE`
- Application-specific parameters are special parameters that apply only to a specific application. If your application sets its own environment variables, you can define them here. There are also a number of parameters that are defined by WorkSpace On-Demand itself, but you can specify to customize the behavior of your application on the client. A number of these are described in the following sections.

### 7.3.7.3 Launch Parameters

WorkSpace On-Demand supports several special parameters that define the way an application is launched. You can specify these parameters on the Parameters page of the Application Definition notebook, just as you can any other application-specific parameter.

`WSOD_LAUNCH_MINIMIZED`

When you specify this parameter with a value, the application is launched minimized. Figure 67 shows the Add Application Parameters pop-up when adding the `WSOD_LAUNCH_MINIMIZED` parameter.

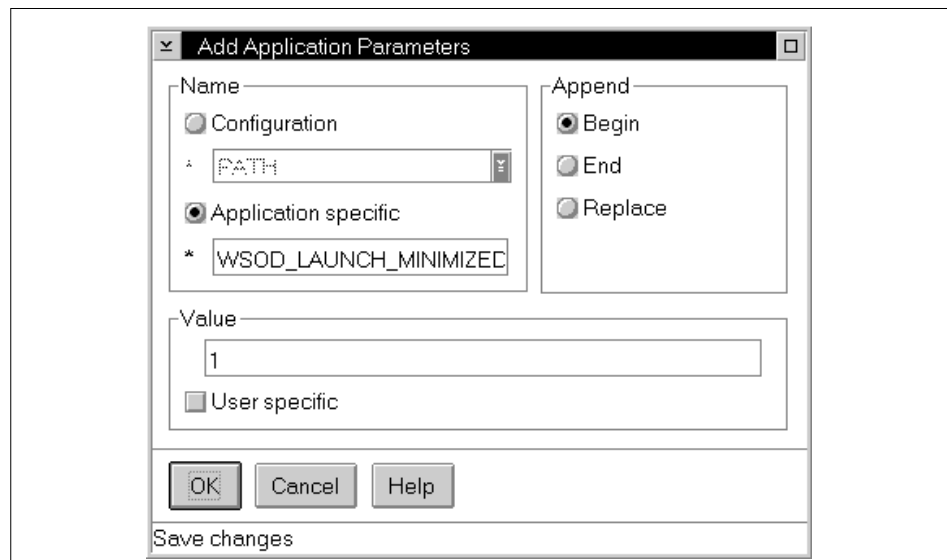


Figure 67. Application Definition - Adding Application-Specific Parameters

## WSOD\_LAUNCH\_SESSION

This parameter gives the administrator full control over the type of session in which WorkSpace On-Demand will launch the application. Use this with care because the application may not work at all if you choose the wrong type of session. Table 21 shows the values that `WSOD_LAUNCH_SESSION` can take and the session types that it will launch.

Table 21. *WSOD\_LAUNCH\_SESSION Values*

Value	Session Type
1	Full-screen OS/2 session
2	Windowed OS/2 session
3	OS/2 Presentation Manager application
4	Full-screen DOS session
7	Windowed DOS session
10	WIN-OS/2 real mode session
12	WIN-OS/2 auto mode selection
15	WIN-OS/2 standard mode, seamless, separate VDM
16	WIN-OS/2 standard mode, seamless, common VDM
17	WIN-OS/2 enhanced mode, seamless, separate VDM
18	WIN-OS/2 enhanced mode, seamless, common VDM
19	WIN-OS/2 enhanced mode, full-screen session
20	WIN-OS/2 standard mode, full-screen session

## WSOD\_LAUNCH\_NODROP

When you specify this parameter with a value, the client operating system leaves network resources, such as application FIT entries, attached when the application's primary executable terminates. This parameter is useful when you wish to launch an application containing a number of executables that must be run in sequence.

If you specify the `NODROP` parameter with a value, the application launcher will keep the application FIT file entries in the in-memory FIT when the application's initial executable terminates. If you do specify the `WSOD_LAUNCH_NODROP` parameter, the application launcher removes the application FIT entries from the in-memory FIT.

When specified with a value, this parameter causes a DOS or OS/2 VIO window not to close when the application terminates. This may prove useful in determining the cause of a problem with the application definition because you will be able to see any error messages the program generates. It is usually best removed once the program is working correctly.

### 7.3.8 Create the Application Definition

Add a title on the **General** page of the Application Definition notebook (for example, PMCAMERA); then select the **Create** button at the bottom left-hand corner of the Application Definition notebook to create the application definition.

---

## 7.4 Defining the Application Using the Command Line Interface

How you administer your applications and users depends on your administrative structure and the knowledge of the people in your remote locations. Many users may need to use the same set of public applications; so it may make sense to define applications and grant access to users through an automated process. Alternatively, the administrators (if any) in your remote locations may not be familiar with such tasks.

Another advantage in using the command line interface is the standardization of your application definitions. By using default scripts to define your applications, you avoid having different application definitions across your different locations. In large enterprise environments where you may have many locations and many users, the use of standardized batch processes or REXX scripts helps to provide a consistent end-user environment.

You can define applications using the `NET APP` command from a command line or a REXX script. The `NET APP` command has the same capabilities as the Application Definition notebook, but has the advantage that you can run it remotely in a full screen session, or you can run it unattended using response files. The command line interface therefore allows you to support remote administrators or to perform all administration from a central site.

WorkSpace On-Demand 2.0 enhances the command line interface to support the new WorkSpace On-Demand application parameters as well as the standard OS/2 and DOS public applications using the `NET APPARM` command. You can use the `NET APP` and `NET APPARM` commands to create WorkSpace On-Demand public applications, to modify the application-specific

parameters, to assign applications to users, or to make application parameters user-specific.

### 7.4.1 OS/2 Presentation Manager Application

You can use the `NET APP` command to create an application definition for an OS/2 PM application. For example, the following creates an application definition for `PMCAMERA`:

```
NET APP PMCAMERA /APPDIR:Z:\PMCAMERA /COMMAND:PMCAM200.EXE
/REMARK:"PMCamera" /TYPE:WSOD /INTERFACE:PM /ADD
```

### 7.4.2 OS/2 Command Line Application

You can use the `NET APP` command to create an application definition for an OS/2 command line (VIO) application that will run in a windowed or full screen OS/2 session. For example, the following command creates a windowed OS/2 command line session:

```
NET APP OS2WCOMD /APPDIR:Z:\OS2 /COMMAND:CMD.EXE /REMARK:"Windowed OS/2
Command Prompt" /TYPE:WSOD /INTERFACE:PM /ADD
```

You can create a definition that directly launches an application program by including the name of the application's EXE file in the `/COMMAND:` parameter. To create a full screen OS/2 session, use the same command with the `/INTERFACE:FS` parameter.

### 7.4.3 WINOS2 Application

To create an application definition for a general WIN OS/2 session, use `PROGMAN.EXE` as the invocation command. *Do not* use `WIN.COM` or `WINOS2.COM`. When you create an application definition for any application that resides in the `WINOS2` directory of the client boot image, the program location (on the Invocation page) must be specified as being on the requester, with the drive and path of the program being the boot drive and `\OS2\MDOS\WINOS2`.

To create an application definition for a `WINOS/2` windowed session (Win3.1 Enhanced Mode), type the following command from an OS/2 command line:

```
NET APP WINOS2W /APPDIR:Z:\OS2\MDOS\WINOS2
/COMMAND:PROGMAN.EXE /REMARK:"Windowed WINOS/2 Command
Prompt" /TYPE:WSOD /INTERFACE:PM /ADD
NET APPPARM WINOS2W /ADD WSOD_LAUNCH_SESSION=17 /PLACEMENT:R
```

When using WINOS/2 full-screen sessions, if the session appears to be unresponsive to the mouse or keyboard input, or both, enter the following command from an OS/2 command line:

```
NET APPARM WINOS2FS /ADD VIDEO_SWITCH_NOTIFICATION=1
/PLACEMENT:R
```

To create an application definition for a WINOS/2 full-screen session, use the same command with the `/INTERFACE:FS` parameter; then use the `NET APPARM` command with the `WSOD_LAUNCH_SESSION=19` parameter (for Windows 3.1 Enhanced Mode).

#### 7.4.4 DOS Application

You can use the `NET APP` command to create an application definition for a DOS application, as shown in the following example:

```
NET APP DOSWIND /APPDIR:Z:\OS2\MDOS /COMMAND:COMMAND.COM
/REMARK:"Windowed DOS Command Prompt" /TYPE:WSOD /INTERFACE:PM /ADD
```

You can then use the `NET APPARM` command to specify DOS settings for the application. For example, the following example configures a DOS application to use 4 MB of EMS memory and 8 MB of XMS memory:

```
NET APPARM DOSWIND /ADD MOUSE_EXCLUSIVE_ACCESS=1 /PLACEMENT:R
NET APPARM DOSWIND /ADD XMS_MEMORY_LIMIT=4096 /PLACEMENT:R
NET APPARM DOSWIND /ADD EMS_MEMORY_LIMIT=8192 /PLACEMENT:R
```

#### 7.4.5 Using Response Files

You can use response files to create applications, set the application parameters to the default values, and assign applications to your users. Response files provide a powerful way to automate your WorkSpace On-Demand administration and apply necessary standards across a number of servers. Appendix B.1, "Application Definition Script – ADDAPPL.CMD" on page 329, contains a REXX script that creates WorkSpace On-Demand applications and sets the application parameters using the command line interface.

You can use the response file shown in Appendix B.2, "Application Definition Response File – ADDAPPL.RSP" on page 333, to create WorkSpace On-Demand applications and assign these applications to users.

The applications shown in Appendix B.2 are:

- An application definition that simply invokes a command line
- An application definition for PM Camera

The application parameters are not provided; so some applications, such as PM Camera, will need more application parameters. The response file is included to show the basic capabilities of the command line interface in conjunction with response files.

---

## 7.5 DOS and WIN-OS/2 Applications

The same general principles apply to setting up a DOS or Windows application under WorkSpace On-Demand as to any other. There are a few special considerations, however.

You must pay particular attention to the way your application handles the Windows INI files. In many cases, you must make a policy decision on whether to allow your users unrestricted access to their own copies of the INI files by means of FIT file extensions or to allow only read access to a common set of standard INI files. Once again, there is no substitute for knowing your application!

There is no need to use OS/2 Warp Server's DOS and Windows Template to create an application definition. Although this will work, it is better to use the WorkSpace On-Demand 2.0 Template, which gives you the option to specify parameters for the application.

### 7.5.1 DOS and WIN-OS/2 Settings

WorkSpace On-Demand supports the use of DOS and WIN-OS/2 settings although it has no equivalent to OS/2's Add Programs facility to define them automatically. You can define any setting as an application-specific entry on the **Parameters** page of the Application Definition notebook simply by entering the name of the setting and the required value.

For example, you can enter the `IDLE_SECONDS` setting with a value of 60, the `IDLE_SENSITIVITY` setting with a value of 100, a `FOLDER` setting with a value of `<WP_APPSFOLDER>`, and so on with all the DOS or WINOS2 settings.

To change these values, you must use the `SET` command, followed by `keyname=value` in the value field on the **Parameters** page in the Application Definition notebook. For example:

```
SET DOS_FILES=45;SET DOS_HIGH=1;
```

Note that when defining DOS settings on an OS/2 Warp 4 client, you would typically select **On** or **Off** radio buttons for some settings. When specifying these settings on the Parameters page in the Application Definition notebook, you must use the Boolean values 1 for On and 0 for Off. For example:

```
SET COM_HOLD=1;
```

will set the `COM_HOLD` setting to On.

Some settings, such as `DOS_VERSION`, may already have default values. You must be careful when specifying these settings since any value that you supply is treated as a replacement (even if you are using the `UPDATEIFEXIST` flag). Therefore, if you wish to add one item to the list of items in the `DOS_VERSION` setting, you should also include all of the existing values.

Note that some settings may not apply to particular clients due to their hardware configuration. For example, the `VIDEO_8514A_XGA_IOTRAP` setting is only available on certain client hardware.

Table 29 on page 335 shows the DOS/WINOS2 settings that you can specify.

You can find a list of standard settings for many applications in the `\IBMLAN\RPL\BB20.US\OS2\INSTALL\DATABASE.TXT` file on your WorkSpace On-Demand server.

Remember that even though a DOS-specific template appears in the **Public Application Definitions** folder, you should use the Workspace On-Demand template since this is the only template that allows you to define application parameters in this way.

---

## 7.6 Java Applications

You can run both Java applets and Java applications on your WorkSpace On-Demand 2.0 clients without using a Web browser. Applets and applications are run by the Java Virtual Machine (JVM) built into the WorkSpace On-Demand client operating system. At the time of writing, the current version of the JVM is Version 1.1.6.

The Java 1.1.6 specification requires 256-color mode (or greater). However, Java applications and applets using the Abstract Window Toolkit (AWT) on WorkSpace On-Demand clients can run in 16-color mode, but may be difficult to read. Using 256-color mode (or higher) is recommended for this version and for future versions of Java for OS/2.

### 7.6.1 Java Applets

WorkSpace On-Demand clients run Java applets on the desktop using the applet viewer `APPLET.EXE`. By making `APPLET.EXE` available as a public

application, you can make Java applications available to WorkSpace On-Demand users.

You will need to configure a separate invocation of APPLET.EXE for each Java application you intend to make available this way. In each case, you must pass the name of the HTML file defining the application as a parameter on the **Invocation** page of the **Application Definition** notebook, as shown in Figure 68.

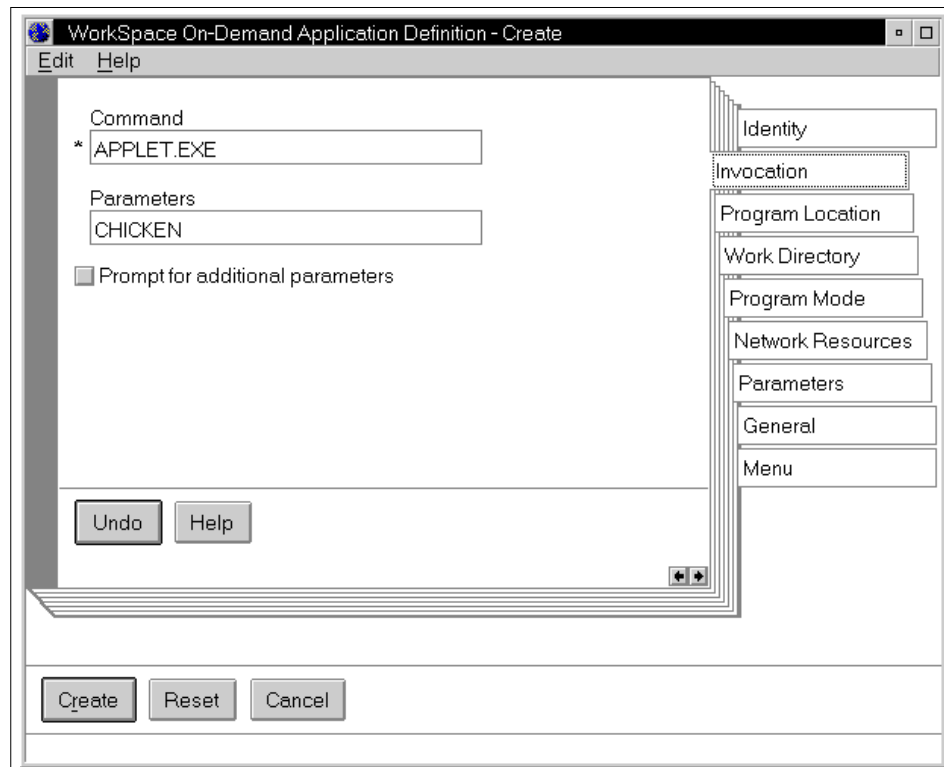


Figure 68. Defining the Java Configuration Applet

You will probably not need to give your users write access to any files in order to run Java applets. Instead, by running the CNFGAPPL utility applet once on the server, you can create a single, central copy of the PROPERTY file, located on the \JAVAx\HOTJAVA subdirectory, from which all the users can then read their Java configuration values.

However, you must redirect the file access for the PROPERTY file using a FIT entry in order for the client workstation to search the correct directory. The default FIT entry in the machine FIT file redirects all access to the



\JAVAx\HOTJAVA subdirectory to the client-specific read-write area on the server. If you wish to redirect access to this subdirectory to the generic read-only area on the server, you should override the default entry with the following:

```
Z:\JAVAl1\HOTJAVA BB20.US\JAVAl1\HOTJAVA
```

You can do this by editing the machine FIT file or by adding the above entry to the default user FIT file DD20USDU.FIT.

## 7.6.2 Java Applications

Setting up a Java application is very similar to setting up a Java applet, with only one or two differences:

- On the **Invocation** page in the **Application Definition** notebook, specify JAVA.EXE in the **Command** field.
- Type the name of the Java program you wish to run in the **Parameters** field.

Apart from the above, defining a Java application is similar to defining any other application.

---

## 7.7 Granting Access to Users

After you have installed and defined a network application on your server, you must grant access to end users so that they can use the application. OS/2 Warp Server's access control policies ensure that, by default, no user has access to any network resource, including applications, unless that access is granted by the system administrator.

### Note

This redbook does not include information on how to define a user ID under OS/2 Warp Server. This function is fundamentally unchanged and is explained in the online *Network Administration Tasks* guide. If you are unsure how to define a user ID for your OS/2 Warp Server domain, you should consult this guide for assistance.

The only significant change introduced by WorkSpace On-Demand is the way in which you assign applications to end users, in particular the provision for user-specific application parameters under WorkSpace On-Demand. This capability is described in this section.

## 7.7.1 Using the GUI

All those tasks can be performed through the OS/2 Warp Server GUI. To assign an application to an end-user:

1. Open **LAN Services** from the OS/2 Warp Server desktop.
2. Open **LAN Server Administration**.
3. Open the appropriate domain object.
4. Open **User Accounts**.
5. Open the object for an existing user ID.
6. Select the **Applications** page, as shown in Figure 69 on page 200.

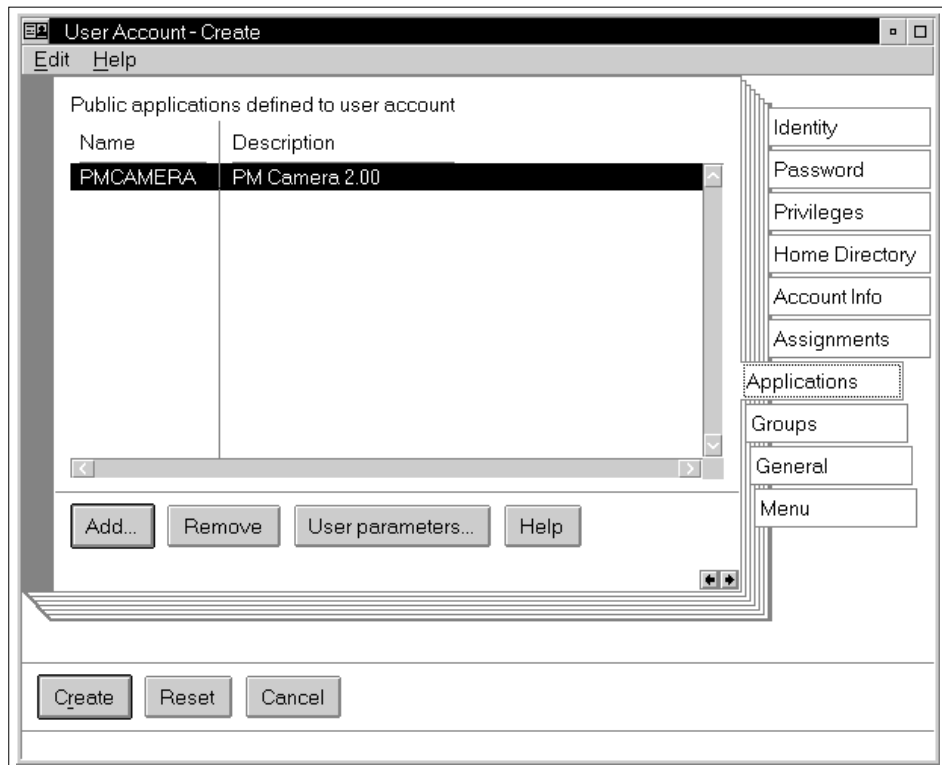


Figure 69. Granting Access to an Application

To assign or add a public application for this end-user, select the **Add** button. The Add Public Applications pop-up is displayed, as shown in Figure 70.

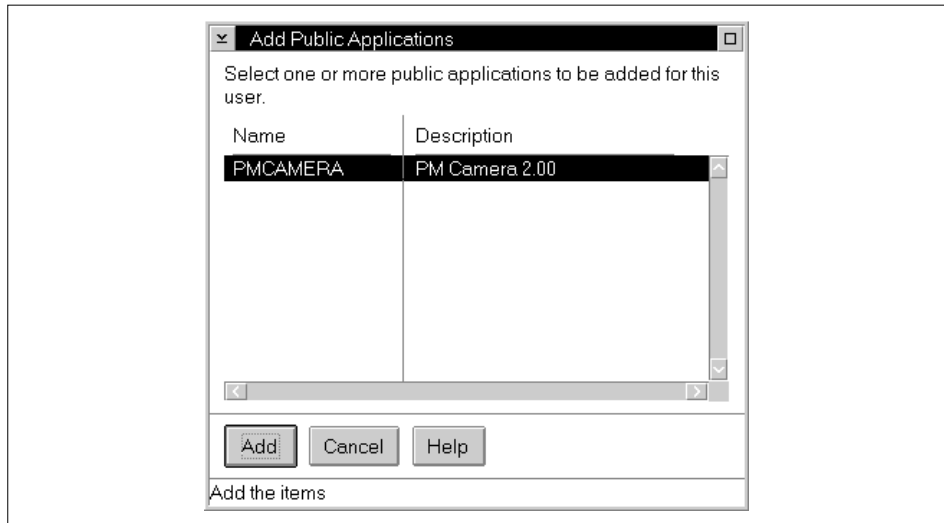


Figure 70. Add Public Applications Pop-Up

Select the application you require from the list and click on the **Add** button to add it to the list of applications on the Applications page.

You can define application access for many users at one time by dragging and dropping the application's icon on a user group icon within the Groups folder. This assigns access to the application for all users within that group. However, you must be careful when adding new users to the group after you have granted access to the application since these new users will not automatically be granted access.

### 7.7.2 Using the Command Line Interface

You can also grant access to an application for an end user through the command line interface using the following command:

```
NET USE user /ASSIGN PUBLIC:application_name
```

where *user* is the user ID to which you wish to assign the application, and *application\_name* is the name of the application as defined on the Identity page of the Application Definition notebook.

You can easily use a REXX program to extend such definitions to a large number of users; so the command line interface provides an ideal method of granting access to applications in organizations with many end users.

### 7.7.3 User-Specific Application Parameters

After you have defined a network application, you can also define certain application parameters on a per-user basis. In order to do this, you must specify the parameter as user-specific when you define the application.

To set user-specific application parameters:

1. Open **User Accounts**.
2. Open the object for an existing user ID.
3. Select the **Applications** page.
4. Select the desired application from the public application list. If the application has user-specific parameters within its definition, the **User Parameters** button will be selectable.
5. Select the **User Parameters** button. The User-Specific Application Parameters pop-up is displayed as shown in Figure 71 on page 202, showing the parameter name, where it is appended, and the value.

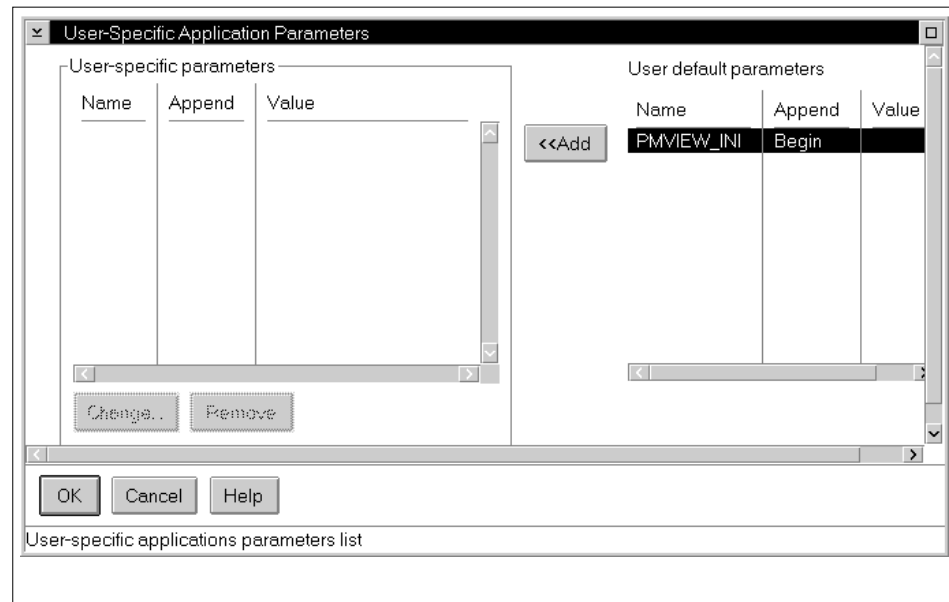


Figure 71. Adding a User-Specific Application Parameter Value

To add a user-specific application parameter setting:

1. Select an application parameter from the **User default parameters** list.
2. Select the **Add** button.

3. This adds the entry to the User-specific parameters list and displays the Set User-Specific Value pop-up.

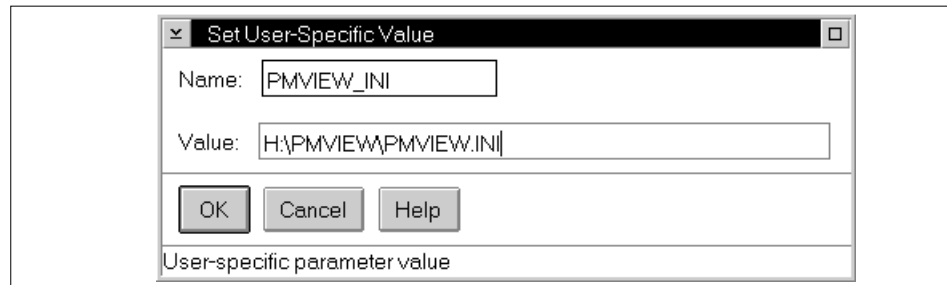


Figure 72. Set User-Specific Value Pop-Up

4. Enter the desired value for the application parameter.
5. Select **OK**.

Note that if you define a parameter as user-specific when you define the application but do not specify a value using the **Set User-Specific Value** pop-up, WorkSpace On-Demand uses the default value that you specified when defining the application.

---

## 7.8 Customizing INI Files

Certain applications may require their own entries in the client's OS2.INI or OS2SYS.INI files. When you install such applications on a traditional "fat" client, the installation program makes the necessary modifications to the INI files. In a WorkSpace On-Demand environment, however, you may need to manually add the necessary entries to these files.

When you compare the file structures on your server or reference client before and after installing an application, you may notice that one or more INI files have been changed. If this is so, the following sections will show you how to determine the specific differences between the files and how to add the necessary entries to the INI files in your WorkSpace On-Demand client's boot image.

### 7.8.1 INI File Structure

An INI file is a binary file that consists of a set of application-type groups. Within each application type is one or more entries consisting of a keyname and a value.

## 7.8.2 Comparing INI Files

You can use the INIDIFF utility included on the CD-ROM that accompanies this redbook to compare two INI files and produce a third file that contains only the differences between the two. The syntax of the `INIDIFF` command is as follows:

```
INIDIFF <source.INI> <new.INI> <delta.INI>
```

The INIDIFF.CMD utility will extract any new application types, keynames and/or values that exist in *new.INI*, but not in *source.INI*, and write them to *delta.INI*. You can then use *delta.INI* to incorporate these changes into the appropriate INI file in your client's boot image.

## 7.8.3 Modifying INI File Entries

You cannot edit an INI file using a normal text editor, but you can use an INI editor, such as UNIMAIN or FM/2, to edit the application types, keynames and values within an INI file. This is often necessary when you are creating or modifying machine classes, particularly when adding support for new video adapters. It is also necessary when you are setting up a network application that requires entries in one or more INI files.

### 7.8.3.1 Using an INI Editor

You can use an INI editor to check the *delta.INI* file created by the INIDIFF utility, and then modify the appropriate INI file in the client's boot image in one of two ways:

- Manually by using the INI editor to add the necessary application types, keynames and values
- Automatically by using the INIMERGE utility included on the CD-ROM that accompanies this redbook. The syntax of the `INIMERGE` command is:

```
INIMERGE target.INI delta.INI
```

Either method will result in a new INI file with the necessary entries to support your machine class or application.

### 7.8.3.2 Using the NET APPARM Command

WorkSpace On-Demand 2.0 allows you to add INI file entries dynamically when you launch an application. You can specify the entries to be added from

the command line by using the `NET APPPARM` command with the appropriate parameters as shown in Table 22.

Table 22. *NET APPPARM Command - INI File Parameters*

<b>NET APPPARM Parameter</b>	<b>Meaning</b>
<code>/FIELDTYPE:I</code>	Indicates that the <code>NET APPPARM</code> command is an INI file update.
<code>/INIFILE:ini_file_name</code>	Indicates the INI file to be updated.
<code>/INIAPP:ini_app_name</code>	Indicates the application type within the <i>ini_file_name</i> to be created or modified. Note that <i>ini_file_name</i> is case sensitive.
<code>/INIKEY:ini_key_name</code>	Indicates the keyname associated with the application type to be created or modified. Note that <i>ini_key_name</i> is case sensitive.
<code>/INIValue:ini_value</code>	Indicates the value associated with <i>ini_key_name</i> . This parameter is not required to create an INI file entry. If not provided, the value is assumed to be an empty string. Note that <i>ini_key_value</i> is case sensitive.
<code>/User</code>	Optional; if specified, the INI file is specific to a user.

The following rules apply to the use of these parameters:

- If the `/INIK` parameter specifies a keyname that already exists within the INI file, the `NET APPPARM` command replaces the existing value for that keyname with the value specified by the `/INIV` parameter.
- If the `/INIA` or `/INIK` parameters specify an application type or keyname that does not already exist, the `NET APPPARM` command adds that application type or keyname to the file.
- If you do not include an `/INIV` parameter, the `NET APPPARM` command simply displays the current value associated with the application type and keyname specified by the `/INIA` and `/INIK` parameters.

For example:

```
NET APPPARM application_name /AD /F:I /INIF:\app_dir\app.ini /INIA:tags
/INIK:heading /INIV:bold /UO
```

You can also use the `NET APPPARM` command to delete an application type or keyname from an application definition. To do this, you must omit the `/INIV`

parameter from the command and include the `/DELETE` parameter as shown below:

```
NET APPARM application_name /AD /F:I /INIF:\app_dir\app.ini /INIA:tags  
/INIK:heading /DELETE
```

---

## 7.9 Application Roaming

WorkSpace On-Demand introduces the concept of *application roaming*, which is the ability for users to logon to any client and immediately access their own applications and data files. Application roaming provides the user access to his/her application from any WorkSpace On-Demand workstation of the same machine class. Previously, although the application was installed on the server, environment information needed to be added to the client workstation's own files. This meant that users had access to certain applications only from particular clients. The enhancements made to the public applications allow for enhanced flexibility, allowing a user to run an application from any client workstation with compatible hardware.

It is important to understand the issue of hardware compatibility. Although a user may be able to logon and start an application on any client workstation in the network, the client workstation must have the correct hardware to support the application. For example, suppose that a graphics application requires an SVGA graphics adapter in order to run correctly. A user may run this application on different clients, provided that each client has an SVGA graphics adapter. However, if the user logs on to a client with a VGA graphics adapter and attempts to run the application, the application will fail.

Application roaming is accomplished through a two-stage process.

1. The first stage provides a hardware affinity by using the WorkSpace On-Demand client's network hardware address to associate a hardware image with a specific client. This software is loaded across the network from the WorkSpace On-Demand server to the WorkSpace On-Demand client.
2. The second stage provides a user affinity by employing the user ID to specify a user's unique applications and the environment required to support them. This environment includes the `PATH`, `LIBPATH` and `DPATH` statements as well as the environment variables required to execute those network-public applications to which the end-user has access. This information is dynamically loaded from the WorkSpace On-Demand server to any WorkSpace On-Demand client from which the user logs on.



Note that the correct use of FIT files to redirect application file access at the user and application level is key to successfully implementing application roaming. Correct redirection of file access through FITs can provide both client- and user-specific configuration data for applications, thereby overcoming some of the issues of hardware compatibility and allowing users to run an application from different clients while maintaining their own personalized application configuration.



## **Part 4. Customizing WorkSpace On-Demand**

This part of the redbook discusses ways in which you can customize WorkSpace On-Demand to meet the requirements of your particular installation.

Chapter 8, “Modifying the User Interface” on page 211, describes how to modify the default client user interface with user exits, and how to replace the default interface with your own customized shell.

Chapter 9, “Supporting Network Adapters” on page 241, describes how to integrate additional types of network adapter into your WorkSpace On-Demand environment.

Chapter 10, “Supporting Additional Hardware” on page 251, describes how to provide support for additional hardware devices in WorkSpace On-Demand, by adding definitions, modifying existing machine classes, or creating new machine classes of your own.

Chapter 11, “Deploying WorkSpace On-Demand” on page 297, discusses some issues that you should consider when deploying WorkSpace On-Demand in a large-scale production environment. The chapter covers topics such as predeployment planning, network infrastructure considerations and so on.



## Chapter 8. Modifying the User Interface

WorkSpace On-Demand provides a simplified version of the OS/2 Workplace Shell, known as the PMLOGON shell. This chapter describes the PMLOGON shell and discusses ways in which you can modify the shell to better suit your requirements, or even replace it with a shell of your own.

### 8.1 PMLOGON

For many years, OS/2 has provided a user interface known as the Workplace Shell (WPS). This shell is object oriented and can be easily customized. Programmers can build common functions once and, using object-oriented programming techniques, reuse them everywhere that such functionality is required. This allows you to easily modify or enhance the shell by making a major change in one place, and that change will automatically be utilized throughout the system.

With WorkSpace On-Demand, IBM has modified the Workplace Shell to restrict users' access to its features. The new shell eliminates pop-up menus and settings notebooks and prevents access to any type of configuration or customization by the end user. The new shell eliminates any means to obtain access to the rest of the Workplace Shell, including command prompts.

The only icons that are accessible are those that are defined for the user as network applications in the WSOD server's domain. These icons are dynamically created on the desktop for the user during the logon process. End users can do nothing with these icons except double-click on them to start them up. They cannot move, delete or alter the icons in any way. The icons are destroyed by the system when the user logs off or shuts down.

The new shell is started by the `RUNWORKPLACE` statement in the client's `CONFIG.SYS` file. Figure 73 shows the default `RUNWORKPLACE` statement.

```
SET RUNWORKPLACE=Z:\OS2\PMLOGON.EXE
```

*Figure 73. PMLOGON Shell - Default RUNWORKPLACE Statement*

PMLOGON invokes the NCAPPUTL utility to manage the construction and operation of the new shell. By default, PMLOGON initializes the shell with a basic desktop, then brings up a logon panel as shown in Figure 74. Notice that prior to logon, the desktop is hidden behind a special blue screen and is not available to the user until a successful logon occurs.

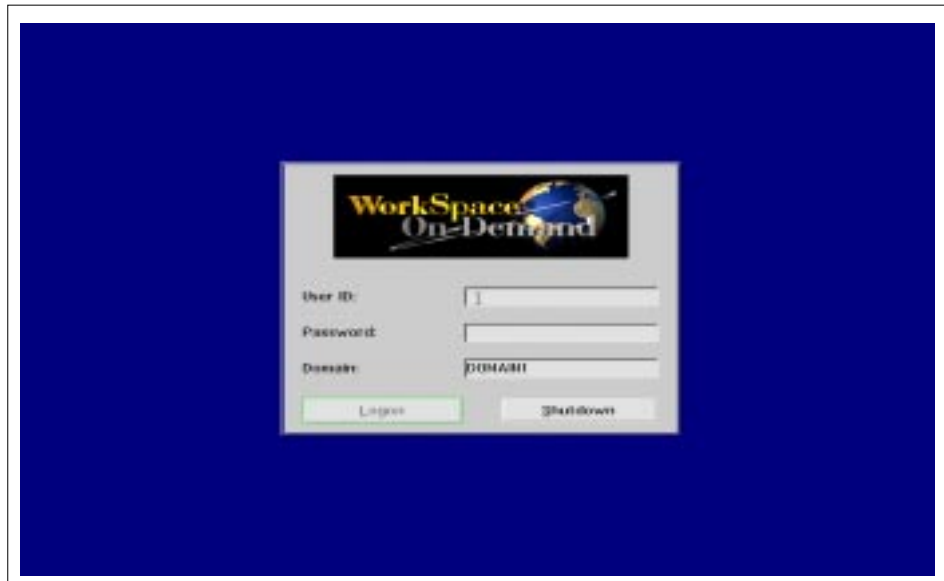


Figure 74. PMLOGON Shell - Logon Panel

The default desktop includes icons for lockup, logoff, shutdown, and refresh. These icons allow the user to lock the desktop, logoff and return to the logon panel, shut down the machine or dynamically "pull down" any changes that an administrator may have made to the user's application definitions.

When a user actually logs onto the system, PMLOGON calls the NCAPPUTL utility to query the user's assigned network applications and builds an object on the desktop for each application in the list. After building the icons, PMLOGON removes the blue screen and reveals the desktop to the user.

By default, PMLOGON arranges the icons in a horizontal row from left to right, starting at the top left-hand corner of the desktop. They are sorted in reverse alphabetical order based on the application ID (a unique alphanumeric ID for every application available to users through WorkSpace On-Demand).

---

## 8.2 Modifying the Default Shell

PMLOGON provides a number of ways that you can modify its behavior to suit your particular requirements, without the need to replace the shell with a customized shell of your own. You can modify the default shell using a variety

of parameters, user exits, setup strings and other techniques that are described on the following pages.

### 8.2.1 Logon Options

You can modify certain options that control the client logon process. These options are controlled in the CONFIG.SYS file with the `RUNWORKPLACE=` statement.

If you modify the template for the CONFIG.SYS file in the machine class directory, all subsequent clients created using that machine class type have the change. The machine class template for the CONFIG.SYS file is located in the `\IBMLAN\RPL\MACHINES\BB20.US\machine_class.MC\` directory on the server.

If you want the change to affect only a single client, modify the CONFIG.SYS in that machine's own directory. This CONFIG.SYS file is located in the `\IBMLAN\RPL\MACHINES\client_id\` directory on the server.

PMLOGON accepts a number of parameters, all of which are optional. If you specify `PMLOGON` in the `RUNWORKPLACE=` statement without any parameters, PMLOGON simply uses its defaults; it displays a logon window with no user ID or password and including the domain name obtained from the client's IBMLAN.INI file.

You can include the following parameters to specify the user ID, password and domain in the `PMLOGON` command. They are displayed in the logon window, and can be changed by the end-user:

- `/U:<UserID>` - User ID
- `/P:<Password>` - Password
- `/D:<Domain>` - Domain name

You can include the following parameters to specify the user ID, password and domain in the `PMLOGON` command. They are displayed in the logon window, but *cannot* be changed by the end-user:

- `/UF:<UserID>` - Fixed user ID
- `/PF:<Password>` - Fixed password
- `/DF:<Domain>` - Fixed domain name

The following parameters affect the behavior of the PMLOGON program during the logon process:

- `/AUTO` - Automatic logon. The logon window is not displayed, and you must specify the user ID and password with the `/U` and `/P` parameters.
- `/BMP:<filespec>` - This specifies a fully qualified path and file name for the bitmap displayed in the logon window.
- `/BFG:r,g,b` - This specifies the button foreground colors for the logon window, where r, g, and b are values between 0 and 255, representing the red, blue and green components respectively.
- `/BEG:r,g,b` - This specifies the button background colors for the logon window, where r, g, and b are values between 0 and 255, representing the red, blue and green components respectively.
- `/NOPI` turns progress indicators off. The progress indicators are the small dialogs that animate to show that PMLOGON is processing.
- `/NOSM1` removes system modality from PMLOGON.EXE (this means that PMLOGON will not switch to the foreground while it is processing) after user exit 1 is executed or at any time after a logoff.
- `/NONFLE` prevents the display of any non-fatal logon error messages.
- `/URX`: specifies the name of the user exit program, as described in Section 8.2.2, “User Exits During Boot and Logon” on page 215.

You should note the following:

- If you specify multiple instances of the same parameter (for example, `/U:usera /U:userb`), the last parameter is used, and all previous ones are ignored.
- If you specify multiple fixed and non-fixed instances of the same parameter type (for example, `/U:usera /UF:userb /U:userc`), the last fixed parameter is used (for example, `/UF:userb`), and all others are ignored.
- You can specify the `/AUTO` parameter along with `/U` or `/UF`, with `/P` or `/PF`, and with `/D` or `/DF`. If multiple instances of the same parameter type are present, the same rules apply as in the two previous notes.
- If you specify `/AUTO` and there is insufficient information to attempt a logon (for example, the user ID is not specified), PMLOGON displays the logon window and the `/AUTO` parameter is ignored.
- If you specify `/AUTO` and `/PW` and there is insufficient information to attempt a log on (for example, either the user ID or password are not specified), the Logon window is displayed, and the `/AUTO` parameter is ignored. `/PW` is processed.



You should be cautious about using the `/AUTO` parameter:

- If you specify `/AUTO` and an end user logs on successfully, then subsequently logs off, the user is automatically logged on again.
- If you specify `/AUTO` and the user's password has expired, the logon is no longer automatic. PMLOGON displays the logon window and prompts the user to change the password. Note that any user can change the password at this point by completing the Change Password window on the client. This may permit a user to access another user's account.
- If you specify `/AUTO` and the user changes the password, the `/AUTO` parameter is not honored until the next time the client machine is booted. For example, if the user logs on and the password is expired, the user changes the password when prompted. When the user logs off, PMLOGON will attempt to automatically log on again. In this case, the logon will fail because the password specified on the `RUNWORKPLACE` statement and the password at the domain controller no longer match. You must update the `RUNWORKPLACE` statement to the new password.

### 8.2.2 User Exits During Boot and Logon

A major enhancement to PMLOGON is the incorporation of user exits in the code. This feature allows PMLOGON to call a customer-written program to perform custom processing tasks at particular points in the startup and logon process.

There are five points at which PMLOGON can call the user exit program. These are:

- Immediately upon shell startup—that is, immediately after the client workstation boots.
- After starting the LAN Requester but before displaying the logon panel. This exit allows you to display your own logon panel instead of the default logon panel.
- Immediately after logon is successfully completed.
- After the desktop icons are built.
- Immediately after logon if an error has occurred (such as a bad user ID or password).

To enable the user exit code, you must modify the `RUNWORKPLACE` statement in the client's `CONFIG.SYS` file, as shown in Figure 75 on page 216.

```
SET RUNWORKPLACE=Z:\OS2\PMLOGON.EXE /URX:Z:\MYEXIT.CMD
```

Figure 75. PMLOGON Shell - Modified RUNWORKPLACE Statement

The `/URX` parameter specifies a REXX program that contains the user exit code and which will be called from PMLOGON at the specified points listed above. If `/URX` is specified with no filename, the `\OS2\PMLOGURX.CMD` command file will be executed. If `/URX` is specified with a filename with no path, the directories specified by the `PATH` environment variable will be searched.

**Note**

The user exit program *must* be a REXX program. The session environment for the REXX program is that of PMLOGON.EXE and therefore does not contain any input or output facility. For example, PULL from the keyboard and SAY to the console will not work.

PMLOGON calls the same REXX program for *all* five user exits. The REXX command file is called for all exits, even if it does not need to process all of the exits. The REXX command file must be able to return error free if it receives an exit number that it does not need to handle.

When PMLOGON exits to the REXX program, it supplies an argument specifying which exit it actually requires. This is known as the *mode* and is an integer indicating the number of the exit (0 through 4). The user exit program must check the mode argument, then call the appropriate routine corresponding to that exit.

Figure 76 on page 217 shows the flow of control when PMLOGON calls a user exit program named MYEXIT.CMD.

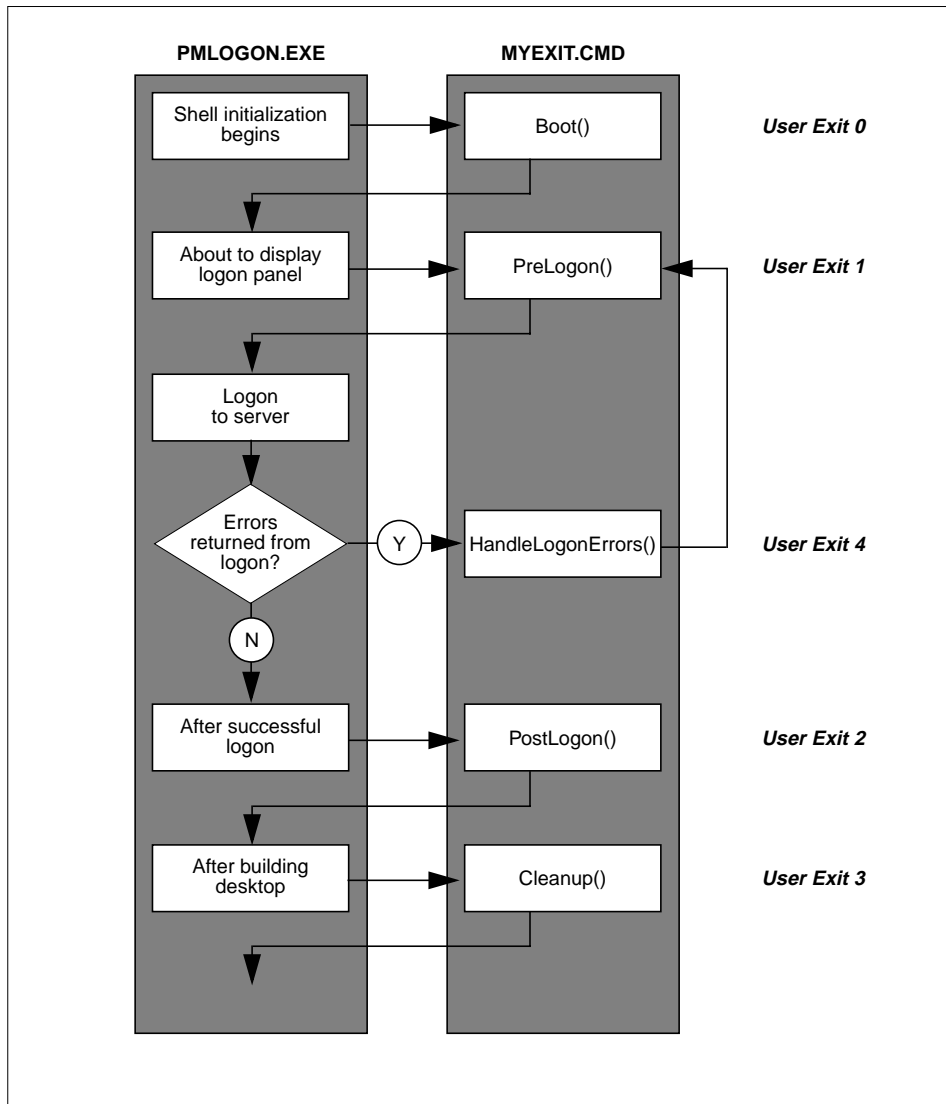


Figure 76. PMLOGON Shell - Flow of Control for User Exits

The following sections will explain the processing that can be performed by each of the user exits. Sample REXX code for the user exit program MYEXIT.CMD is included on the CD-ROM that accompanies this redbook, and it may be useful as you read the narrative.

### **8.2.2.1 Exit 0 - Boot Time**

PMLOGON.EXE calls user exit 0 immediately after it starts. In the sample MYEXIT program, this is the Boot() routine. Exit 0 allows you to start programs and handle issues before the shell is fully operational. Some examples of tasks that you can perform at this time are:

1. Checking IP connectivity
2. Initializing queues for REXX interprocess communications from MYEXIT.CMD
3. Starting programs that must run on all machines (for example, AntiVirus, Netfinity and so on)

After completing these tasks, MYEXIT returns control to PMLOGON. In our sample program, MYEXIT does not return any data to PMLOGON.

### **8.2.2.2 Exit 1 - Prior to Logon**

After the user exit program returns control, PMLOGON starts the LAN Requester task. Before it displays the logon panel, PMLOGON calls exit 1. This is the Prelogon() routine in our sample MYEXIT.CMD.

This exit allows you to substitute your own logon panel for the default logon panel normally displayed by PMLOGON and to perform any custom processing prior to logon, or based on the user's input at logon time. This type of customized logon procedure is shown in the control flow illustrated in Figure 76 on page 217.

While our sample program, MYEXIT, simply returns a default user ID and password from exit 1, the user exit program might:

1. Display a custom logon panel (such as that shown in Figure 77 on page 219).
2. Wait for input from this custom logon panel
3. Make changes to a user's LAN server profile or save key information based on the user's input
4. Return logon information to PMLOGON so that it can log on on the user's behalf

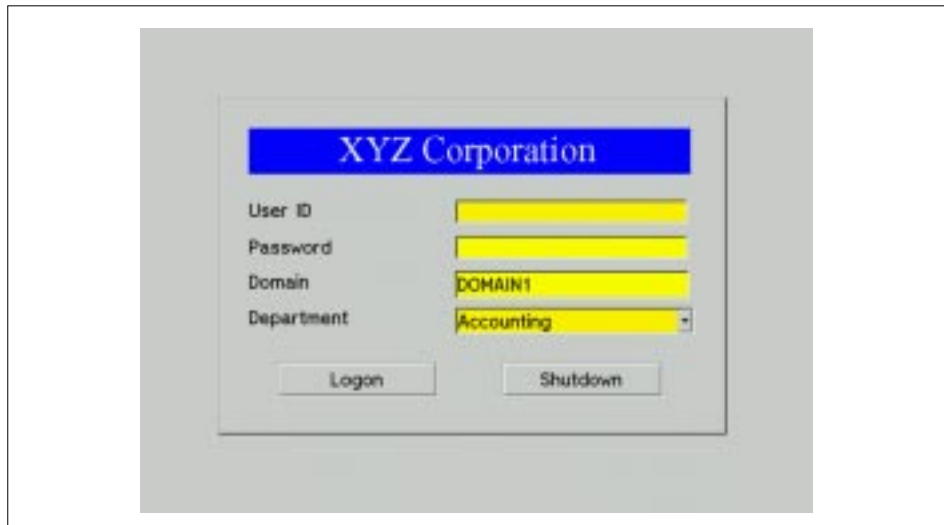


Figure 77. PMLOGON Shell - Custom Logon Panel

When the user exit program gathers logon data as part of its processing for exit 1, it can return that data to PMLOGON, which will then perform the logon for the user. Figure 78 shows the required format of a returned logon string.

```
logonstring = "/AUTO /U:userid /P:password"
```

Figure 78. PMLOGON Shell - Logon String Format

PMLOGON parses the logon string to obtain the user ID and password, and attempts to log on on behalf of the user ID specified. If the logon is successful, PMLOGON then calls exit 2. If the logon fails, PMLOGON calls exit 4 with the UPM return code as the second parameter.

### **8.2.2.3 Exit 2 - After Successful Logon**

Immediately after a successful logon, PMLOGON calls exit 2. In our sample program, MYEXIT, this is the PostLogon() routine. You can use this exit to perform any required post-logon activities such as:

1. Running `NET USE` commands to attach to network resources such as shared drives or printers
2. Run any custom utilities
3. Delete temporary files created as part of a custom logon procedure
4. Back up the user's home directory if this is the user's first logon for the day

When the user exit program completes its processing, it returns control to PMLOGON. No return values are processed from exit 2.

### **8.2.2.4 Exit 3 - After Building the Desktop**

When the user exit program returns control after processing exit 2, PMLOGON builds the desktop based on the user's application definitions on the OS/2 Warp Server domain. The network application definition contains everything necessary to create an icon on the desktop, including the name of the executable, any necessary parameters, the icon file, position, and title.

After creating the objects on the desktop but before they are displayed, PMLOGON calls exit 3. In our sample, MYEXIT, this is the Cleanup() routine.

You can use exit 3 to perform tasks such as:

- Backing up log files
- Autostarting applications such as Personal Communications/3270
- Closing down any custom logon procedures that might be in use

After the user exit completes its processing for exit 3, it returns control to PMLOGON with no return data. If the user exit program provided its own custom logon panel, then PMLOGON has nothing more to do. However, if the default logon panel is used, PMLOGON clears it from the screen.

PMLOGON then goes into a dormant state. It is still running, but the logon panel is only reactivated when the user logs off (see Section 8.2.2.6, "Logoff and Shutdown" on page 221).

### 8.2.2.5 Exit 4 - After Unsuccessful Logon

PMLOGON only calls exit 4 if an error has occurred during logon. In our sample program MYEXIT, this is the HandleLogonErrors() routine. Exit 4 allows the user exit program to notify the user or take other corrective action.

The user exit program can return a logon string from this exit, and must return a logon string if a custom logon panel is in use. The listing for MYEXIT.CMD, which is on the CD-ROM that accompanies this redbook, shows how the code for exit 2 is "reused" by exit 4. Note that the UPM error messages are defined in \MUGLIB\UPM.H.

### 8.2.2.6 Logoff and Shutdown

When the user chooses **Logoff**, the shell performs the following operations:

1. Displays a dialog box warning the user that all programs will be closed and that the user will be logged off.
2. If the user clicks **OK**, all running programs are closed, the user is logged off, and the shell is closed.
3. When PMLOGON restarts (it always will restart because it is the shell specified in the `RUNWORKPLACE` statement in `CONFIG.SYS`), it will call exit 1.
4. If the user selects **Cancel**, they will be brought back to the desktop with no changes.

Note that exit 0 is not called when PMLOGON restarts. This exit is called only once, when the client workstation first boots.

PMLOGON calls the user exit program and invokes exit 1 for the first time when the client boots, and every time the user logs off.

Exits 2, 3, 4 always act the same way. They are called during and after any logon by the user.

The "Shutdown" icon prompts the user to confirm that they want to shut down. If the user selects **OK**, all programs will be closed, the user is logged off, and an OS/2 shutdown will occur.

### 8.2.2.7 Example of Where User Exits Can Be Used

An IBM Services group in Barcelona has taken advantage of the user exits in PMLOGON to implement a host-centric user and application management system. A brief description of the implementation is given as an example of the potential enhancements that can be made to WorkSpace On-Demand using the PMLOGON user exits.

The system essentially utilizes RACF and a DB2 database on a central host for logon verification and to dynamically "serve" applications to each user. All applications are preinstalled on each WSoD server. The host database stores records for each application along with the application assignments for all the users and groups.

When a client is booted, a custom logon screen (instead of the default PMLOGON panel) is presented to the user through the user exits. The logon attempt is verified first at the server and then in RACF. If the user ID is not already defined in the LAN Server domain, it will be created and verified with RACF. If the logon attempt is verified, the user ID will be looked up in the host database to determine which applications they can access. This information is passed down to the WSoD server, and the appropriate applications are assigned to the user (using an Admin logon and the `NET APP` command line interface). Control is then passed back to PMLOGON, and the desktop is built with the network applications assigned to the user.

This system provides the capability for users to roam from site to site and still receive the applications assigned to them. It also gives a central group of administrators the ability to add, delete and manage access to applications for all users in the enterprise.

### 8.2.3 Replacing the Logon Bitmap

You can set the client logon window to display any bitmap that you wish. For example, you may wish to use your corporate logo here. You can set all client workstations in a specific machine class to use the same bitmap, or you can set any individual client to display its own bitmap image.

The logon window bitmap is controlled by the `/BMP` parameter in the `RUNWORKPLACE=PMLOGON` statement in the client `CONFIG.SYS` file. If you include the `/BMP` parameter to specify a bitmap, that bitmap is scaled and displayed in the logon window. If you do not specify a bitmap, PMLOGON uses the `PMLOGON.BMP` file in the `\OS2` directory.

To change the client logon bitmap:

1. Edit the `CONFIG.SYS` file using an ASCII editor.
2. Modify the `RUNWORKPLACE=Z:\OS2\PMLOGON.EXE` statement to include the `/BMP:file specification` parameter. The `file specification` parameter is the fully qualified path and file name of the bitmap to be displayed on the Logon window. For example:

```
RUNWORKPLACE=Z:\OS2\PMLOGON.EXE /BMP:Z:\BITMAPS\USER1.BMP
```

3. Save the `CONFIG.SYS` file.



4. Place the desired bitmap in the directory that is specified in the `RUNWORKPLACE=` statement.

**Note**

If you specify a file name only and not a fully qualified path, the `DPATH` is used to locate the bitmap. If the specified bitmap file is invalid or cannot be found, PMLOGON uses the `PMLOGON.BMP` file in the `\OS2` directory.

### 8.2.4 Replacing the Background Bitmap

The background bitmap for the client's desktop is specified in the client's `OS2.INI` file, which resides in the `\IBMLAN\RPLUSER\client_id\OS2` directory. When the client is initially defined, this file is copied from a template that resides in the `\IBMLAN\RPL\BB20.US\machine_class.MC\OS2` directory. You can edit this template file to change the background bitmap prior to defining your clients, or you can modify individual clients' background bitmaps after client definition by editing each client's `OS2.INI` file.

To change the background bitmap, you will need an INI file editor. As always, when manipulating INI files, it is best to make a backup copy of the file first. The name of the bitmap file is stored in the `PM_SystemBackground` section under the `DefaultDesktopBackground` keyname. However you can also cheat by simply replacing the file containing the default background bitmap with a file containing your required bitmap.

---

## 8.3 Customizing Program Objects

With a traditional OS/2 client workstation, it is possible for an end user to customize the objects on the Workplace Shell desktop in a wide variety of ways. Most desktop objects have a settings notebook which allows the end user to change the properties for the object. WPS setup strings can also be used to set the initial properties during the creation of an object or to change the properties of an existing object.

Manual methods of modifying the desktop and its objects do not work under WorkSpace On-Demand since the end user does not have access to system settings, settings notebooks and so on. Indeed, one of the benefits of WorkSpace On-Demand is that an end user cannot modify, and potentially corrupt, their desktop environment.

All objects are dynamically created on the desktop for the user during the logon process and are destroyed when the user logs off or shuts down. This

effectively prevents any objects on the desktop from being altered in any way by the user.

WorkSpace On-Demand 1.0 also restricted the ability of the administrator to customize objects. The public application definition, which contains all of the information necessary to create a program object, did not allow you to specify a WPS setup string. Therefore, there was no way to change any of the default property values used when the application was defined.

As a network administrator, however, you may wish to modify the layout of the desktop or the appearance of the program objects on the desktop. WorkSpace On-Demand 2.0 allows you to do this by specifying a setup string as part of the public application definition.

**Note**

The function is also available in WorkSpace On-Demand 1.0 with Fixpak 9 applied to the client image.

A new application parameter was introduced to provide this functionality. The variable name is `NCC_SETUP_POST` and its value can consist of any valid WPProgram setup string.

When the PMLOGON shell calls the NCAPPUTL utility to create its program objects at logon time, NCAPPUTL searches the public application definition for the `NCC_SETUP_POST` application parameter. If the parameter is found, its value is appended to the normal setup string that is used to create the object.

A setup string consists of a *keyname* followed by an equal (=) sign followed by a *value*. It must end with a semicolon (;). Multiple setup strings must be separated with semicolons. Below is an example of a setup string which would set the icon and icon position of a public application.

```
NCC_SETUP_POST = ICONFILE=Z:\WORDPRO.ICO;ICONPOS=20,70;
```

Figure 79. Example of `NCC_SETUP_POST` variable

Note that some setup strings currently used in the Workplace Shell, such as `OBJECTID` and `TITLE`, may not work using this method. This is due to the fact that NCAPPUTL uses information from other sources to set these values for the object. For example, the `OBJECTID` is created as `<NCID_ plus the unique application ID>`. An application with an ID of `WORDPRO` would have its `OBJECTID` set to `<NCID_WORDPRO>`.

**Hint**

The icon's title is taken from the REMARK field in the network application definition, and is therefore irrespective of the object's setup string. Note that if a carriage return is necessary in your title, specify the REMARK with the caret character (^) as in "WordPro^Startup".

While you can use any WProgram setup string in the `NCC_SETUP_POST` parameter, Table 23 shows some of the most useful WProgram setup strings.

Table 23. WProgram Setup Strings

KEYNAME	VALUE	DESCRIPTION
CCVIEW	DEFAULT NO YES	Specifies the default value for concurrent views Creates new views Displays the open view
ICONFILE	file_name	Sets the object's icon to file_name
ICONPOS	x,y	Sets the initial position in a folder. The "x" and "y" values represent the position in the object's folder in percentage coordinates. The left bottom corner of the desktop is the point of origin (0,0).
ICONRESOURCE	id,module	Sets the object's icon. The "id" is the icon resource ID in the dynamic link library "module".

See the *Workplace Shell Programming Reference* for a more complete description of the above setup strings.

You can specify setup strings with the `NCC_SETUP_POST` parameter using either the Application Definition notebook or using the command line interface. Note that you can specify `NCC_SETUP_POST` as an application-specific or user-specific parameter. If both an application-specific and a user-specific parameter exist, only the user specific parameter will be used.

**Note**

A user-specific `NCC_SETUP_POST` parameter is a useful way to customize a public application on a specific user's desktop without affecting any other user's desktop.

### 8.3.1 Positioning Icons on the Desktop

The ability to position icons on the desktop is a good example of the flexibility that setup strings provide. By default, application objects on the WorkSpace On-Demand client desktop are arranged from left to right across the top of the desktop. The icons are arranged in the reverse alphabetic order of their public application IDs.

By using a setup string, you can set the exact icon position of a program object within the folder (normally the desktop itself) in which it resides. You can use the `NCC_SETUP_POST` parameter to specify an `ICONPOS=` setup string, which is used when the program object is created at logon time. As with all setup strings, you can specify the icon position using the Application Definition notebook or the command line interface.

#### 8.3.1.1 Using the GUI

To specify an icon position of 90,90 for a public application using the LAN Server Administration GUI, complete the following steps:

1. Select the desired application from the **Public Application Definitions** folder.
2. Open the **Application Definition** notebook.
3. Click on the **Parameters** page.
4. Select the **Add** pushbutton. The Add Application Parameters pop-up is displayed.
5. Choose **Application specific** and enter `NCC_SETUP_POST` in the entry field.
6. Enter `ICONPOS=90,90;` in the **Value** field.
7. Select **OK**.
8. Select **Set** or **Apply**.

Figure 80 on page 227 shows an example of the **Add Application Parameters** with the `NCC_SETUP_POST` parameter specified. The value in the **Value** field stipulates that the application's icon should be positioned 90 percent of the way across its parent folder (starting from the left-hand side) and 90 percent of the way up the folder (starting from the bottom).

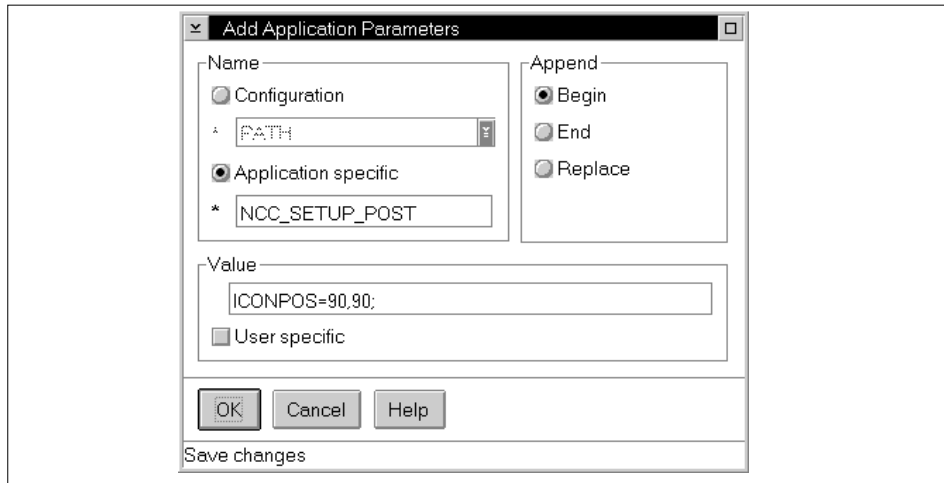


Figure 80. Icon Positioning - Using the GUI

### 8.3.1.2 Using the Command Line Interface

You can use the `NET APPARM` command to set the `NCC_SETUP_POST` parameter for an existing public application. The following `NET APPARM` command will set an icon position of 90,90:

```
NET APPARM [applid] NCC_SETUP_POST=ICONPOS=90,90; /P:R /ADD
```

This command will result in an identical application icon to the one specified using the OS/2 Warp Server GUI in Figure 80.

You must specify the placement parameter (`/P`) when creating or modifying an application parameter that is a run-time environment variable. Use the `R` placement setting to replace the existing value of `NCC_SETUP_POST` with `ICONPOS=90,90`.

The public application will be positioned in the upper-right corner of the desktop as illustrated in Figure 81 on page 228.

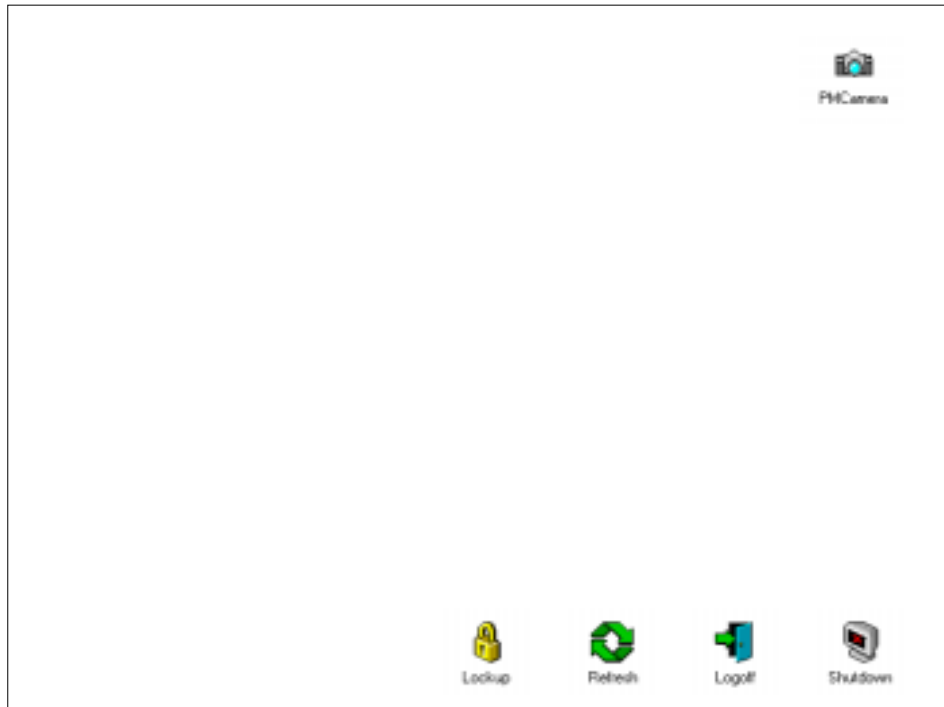


Figure 81. Icon Positioning Example

### 8.3.2 Creating Folders

The ability to place public applications in folders on the desktop is important for several reasons:

- Having many applications on the desktop causes additional time to load the desktop.
- Being able to group applications into folders on the desktop is a significant usability enhancement.

In WorkSpace On-Demand 1.0, there was no restriction for creating client-specific folders, but there was no easy way for an administrator to create them, and once created, they remained on the client's desktop regardless of which user logged on to the client. In addition, a folder created from the WPFolder class on a WorkSpace On-Demand 1.0 client had many functions, such as a context menu, that were not available on other objects. This meant that creating and using folders on a WorkSpace On-Demand 1.0 client was not practical.

WorkSpace On-Demand 2.0 introduces a new object class named ThinFolder. The installation program registers the ThinFolder class and replaces the existing folder class (WPFolder) in the client operating system image with ThinFolder. All the clients that you define using WorkSpace On-Demand 2.0 will use the ThinFolder class, which means that they cannot be dragged on the desktop.

**Note**

Any WorkSpace On-Demand 1.0 clients already defined on your server when you install WorkSpace On-Demand 2.0 will not be updated with the ThinFolder class replacement of WPFolder. If you create public application definitions within folders, allocate those applications to an end user, and the end user logs on to a WorkSpace On-Demand 1.0 client, the folders will appear, but they will have the full WPFolder capabilities including full context menus, drag capabilities and so on.

WorkSpace On-Demand introduces a new application parameter named `NCC_FOLDER`. This parameter can be added to a public application definition to designate the folder in which the application should be placed. The value of `NCC_FOLDER` defines the folder and must be a valid WPFolder (and therefore WPFileSystem and WPObject) setup string.

**Hint**

An open view of a folder will always be able to be moved and sized. The location and size of the open folder view are saved within one logon session, but are not preserved between two logon sessions (since all objects are destroyed and re-created between each logon).

If you wish to permanently set the location and size of a folder on the desktop, use the `ICONVIEWPOS` setup string as a parameter to the `NCC_SETUP_POST` environment variable.

You can use any WPFolder or WPObject setup string within the `NCC_FOLDER` parameter. Table 24 on page 230 and Table 25 on page 231 show some of the most useful WPFolder and WPObject setup strings.

Table 24. WPFolder Setup Strings

KEYNAME	VALUE	DESCRIPTION
ALWAYSSORT	YES NO	Sort order is always maintained. Sort order is not maintained.
BACKGROUND	N M S B C	Sets the folder background: N = image file name M = image mode S = scaling factor B = background type C = background color
DEFAULTVIEW	ICON TREE DETAILS	Default folder view is ICON. Default folder view is TREE. Default folder view is DETAILS.
DETAILSFONT	size.facename	Size and font name for details view
DETAILSVIEW	MINI NORMAL	Small icons in Details view. Normal icons in Details view.
ICONFONT	size.facename	Size and font name for icon view.
ICONNFILE	1,filename	Open folder icon file is filename.
ICONNRESOURCE	1,id,module	Sets the resource of the animation (open) folder icon to resource number "id" in the DLL "module".
ICONVIEW	FLOWED NOGRID NONFLOWED INVISIBLE MINI NORMAL	Sets the folder icon view to the specified style.
ICONVIEWPOS	x,y,cx,cy	Initial folder size in icon view. These values are percentages of the screen size.
TREEFONT	size.fontname	Size and font name for the Tree view
TREEVIEW	NORMAL MINI INVISIBLE LINES NOLINES	Sets the folder tree view to the specified style.
WORKAREA	YES NO	Sets the folder to a work area. Sets the folder to a non-work area.



Table 25. WPObject Setup Strings

KEYNAME	VALUE	DESCRIPTION
CCVIEW	DEFAULT YES NO	Specifies the default value for concurrent views. Creates new views. Displays the open view.
HIDEBUTTON	YES NO	Views have a hide button instead of a minimize button. Views have a minimize button instead of a hide button.
ICONFILE	filename	Sets object's icon to icon in filename.
ICONPOS	x,y	Sets the initial icon position in a folder. "x" and "y" values represent the position in the object's folder in percentage coordinates.
MINWIN	HIDE VIEWER DESKTOP	Hide when minimize button selected. Minimize object to Minimized Window Viewer. Minimize object to desktop.
OBJECTID	<name>	Sets a persistent ID for the object.
TITLE	title	Sets the object's title.

See the *Workplace Shell Programmers Reference* for a more detailed explanation of the above setup strings.

The `TITLE=` setup string within the `NCC_FOLDER` parameter determines the name of the folder in which the network application is placed. If you do not specify an `NCC_FOLDER` parameter for an application object, or if you specify `NCC_FOLDER`, but omit the `TITLE=` setup string, the object will be created directly on the end user's desktop, as in WorkSpace On-Demand 1.0.

### 8.3.2.1 Multiple Applications in the Same Folder

If two or more network applications are created in the same folder, each application must have similar `NCC_FOLDER` setup strings. Each `NCC_FOLDER` setup string is passed to the `NCAPPUTL` utility and is used to create the folder. `NCAPPUTL` processes values in setup strings in different ways:

- In most cases, the last value set for a particular setup string is the value that will take effect when the end user's desktop is populated. For example, if two applications have an `NCC_FOLDER` parameter and one has

DEFAULTVIEW=TREE and the other has DEFAULTVIEW=ICON, then the last program object processed by NCAPPUTL.EXE would take effect.

Other setup strings are only processed the first time an object is created. For example, if two applications have an NCC\_FOLDER parameter and one has ICONPOS=50,50 and the other one has ICONPOS=10,10, the first object created will have its ICONPOS= position used.

Since you cannot set the order in which applications are created on the desktop, and therefore the order in which applications NCC\_FOLDER environment variable are parsed and used, we strongly recommend that you use the same setup string values for NCC\_FOLDER for all applications that reside in the same folder to avoid unpredictable results.

To illustrate the creation of multiple applications in the same folder, suppose that two public applications, Command Reference and REXX Information, are created in a folder named Information. The same NCC\_FOLDER environment variable is used for both applications and consists of the following value:

```
NCC_FOLDER = Title=Information;
```

Figure 82. NCC\_FOLDER Environment Variable

The result is an Information folder on the client desktop, which contains the Command Reference and REXX Information program objects as shown in Figure 83 on page 232.

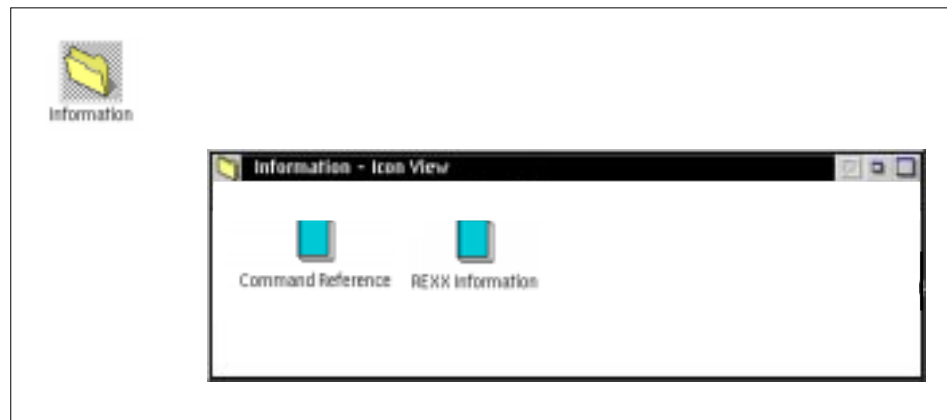


Figure 83. Multiple Applications in the Same Folder

### 8.3.2.2 Multilevel Folders

WorkSpace On-Demand 2.0 provides the ability to create multilevel folders—that is, to nest a folder within another folder. A special setup string named `NCC_CHILD_SETUP` can be added to the value of the `NCC_FOLDER` parameter. The value of `NCC_CHILD_SETUP` is the name of another parameter containing a WPFolder setup string for the child folder in which the program icon should be created.

For example, suppose a public application definition includes the parameters shown in Figure 84.

```
NCC_FOLDER=Title=Secretaries;NCC_CHILD_SETUP=BOOKKEEP;  
BOOKKEEP = Title=Bookkeeping;
```

Figure 84. `NCC_FOLDER` Environment Variable - Multilevel Folders (1)

The `NCAPPUTL` utility will create the program object in the Bookkeeping folder, which itself resides within the Secretaries folder on the desktop.

You can repeat this methodology can be to create as many levels of folders as you wish. To use another example, suppose that you wish to create a folder named OS/2 System, with a subfolder named Command Prompts. The Command Prompts folder contains the OS/2 Window and OS/2 Full Screen applications. The application definitions for OS/2 Window and OS/2 Full Screen both include the parameters shown in Figure 85 on page 233.

```
NCC_FOLDER = Title=OS/2 System;NCC_CHILD_SETUP=PROMPTS;  
PROMPTS = Title=Command Prompts;
```

Figure 85. `NCC_FOLDER` Environment Variable - Multilevel Folders (2)

The result is an OS/2 System folder on the desktop with a Command Prompts subfolder containing the OS/2 Window and OS/2 Full Screen program objects as seen in Figure 86 on page 234.

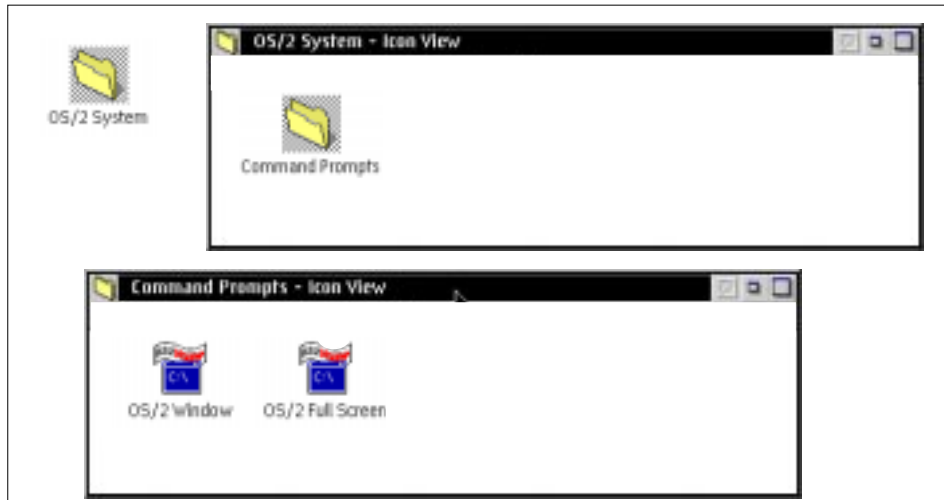


Figure 86. Multilevel Folders

### 8.3.2.3 Customizing Folders

You can use valid WPFolder setup strings to customize the appearance and behavior of folders on your user's desktop. For example, suppose that you create an Information folder with two applications, Command Reference and REXX Information. You can set the following properties for the Information folder:

- Title ("Information")
- Icon resource (the Warp 4 default resource was used)
- Icon position (90,90)
- Initial folder size and position
- Icon font (14 point Helvetica)
- Icon view (flowed)

The public application definitions for the Command Reference and REXX Information applications both include the environment variables shown in Figure 87.

```
NCC_FOLDER = Title=Information;ICONFONT=14.Helv.;ICONPOS=5,90;
ICONRESOURCE=60,PMWP;ICONVIEW=FLOWED;
```

Figure 87. Environment Variables

The desktop resulting from these public application definitions is shown in Figure 88.

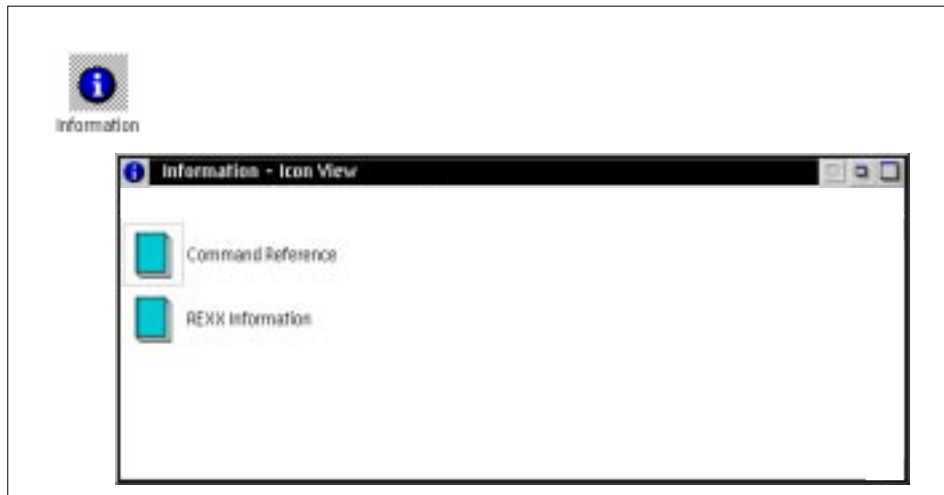


Figure 88. Customizing Folders

### 8.3.3 Modifying the Default Icons

The default WorkSpace On-Demand desktop includes icons for Lockup, Refresh, Logoff and Shutdown in the lower-right corner of the screen. These icons are not defined as public applications; so you cannot customize them in the same manner in the same manner as regular program objects. The definitions for these objects are contained in the client's OS2.INI file, which resides in the \BMLAN\RPLUSER\client\_id\OS2 directory. You can change the properties of these objects (such as the icon or icon position) or even remove them altogether by editing each client's OS2.INI file.

If you wish to change the same property for all clients, the best method is to edit the INI.RC file (in the \BMLAN\RPL\BB20.US\machine\_class\OS2 directory) and re-create the base OS2.INI for each machine class. The INI.RC file includes `PM_InstallObject` entries which define the Shutdown, Logoff, Refresh, and Lockup objects, as shown in Figure 89.

```

/* Default set of Restricted Workplace Shell objects */
"PM_InstallObject" "Shutdown;WPPProgram;<WP_DESKTOP>;UPDATE"
"EXENAME=?:\os2\tshutdwn.exe;HELPPANEL=4001;ICONPOS=88
6;OBJECTID=<THIN_SHUTDOWN>"
"PM_InstallObject" "Logoff;WPPProgram;<WP_DESKTOP>;UPDATE"
"EXENAME=?:\os2\tlogoff.exe;HELPPANEL=30018;ICONPOS=78
6;OBJECTID=<THIN_LOGOFF>"
"PM_InstallObject" "Refresh;WPPProgram;<WP_DESKTOP>;UPDATE"
"EXENAME=?:\os2\ncapputl.exe;PARAMETERS=-R;ICONPOS=68
6;OBJECTID=<THIN_REFRESH>"
"PM_InstallObject" "Lockup;WPPProgram;<WP_DESKTOP>;UPDATE"
"EXENAME=?:\os2\tlockup.exe;ICONPOS=58 6;OBJECTID=<THIN_LOCKUP>"

```

Figure 89. Default Desktop Icons - INI.RC Entries

You can modify or remove these entries; then use the MAKEINI utility to re-create the OS2.INI file from the INI.RC file by first deleting or renaming the current OS2.INI file, then issuing the following command:

```
MAKEINI OS2.INI INI.RC
```

You can also make modifications to the OS2.INI file for all clients and all machine classes, by editing the base INI.RC file in the client operating system image, and then running the NCRPLCFG utility.

Note that any clients you create before running the MAKEINI utility will not be affected since their OS2.INI files have already been created. If you wish your changes to apply to these clients, you must modify their OS2.INI files directly, or delete and re-create the client definitions.

---

## 8.4 Locking Up the Desktop

WorkSpace On-Demand 2.0 creates a Lockup program icon on the desktop of all client workstations. Launching this program locks the desktop and presents the end user with a lockup bitmap.

Users can unlock the desktop and remove the lockup bitmap by entering their LAN password and pressing the **Enter** key. If the end user has no LAN password, the lockup bitmap is removed by pressing the space bar and the **Enter** key.

The default lockup bitmap for all clients is the BIGBLU.BMP file that resides in the \BMLAN\RPL\BB20.US\OS2\BITMAP directory. The simplest way to

change the lockup bitmap file is to replace the BIGBLU.BMP file with a new bitmap.

**Note**

The user's lockup password is taken from the text entered in the password field on the logon panel after a successful logon. However, if a user does not *require* a password, but nevertheless enters text in the password, the text will be ignored for logon purposes, but will be picked up and used as the lockup password.

This may result in users being unable to unlock their desktops since they were unaware of, or unable to remember, the text that they entered in the password field on the logon panel. In such cases, the user must reboot the client.

WorkSpace On-Demand 2.0 introduces new parameters for PMLOGON.EXE that relate to the desktop lockup function. Table 26 lists these additional parameters and their meanings.

Table 26. Desktop Lockup - PMLOGON Parameters

Parameter	Description
/AL	Indicates that automatic desktop lockup is enabled. The desktop lockup screen will be enabled after 15 minutes of keyboard and mouse inactivity.
/AL: <i>minutes</i>	Indicates that automatic desktop lockup is enabled. The desktop lockup screen will be enabled after the specified number of <i>minutes</i> of keyboard and mouse inactivity. <i>Minutes</i> must be a decimal number between 1 and 99. If <i>minutes</i> is not valid, the default timeout value of 15 minutes will be used.
/LOS	Indicates that the desktop lockup screen will be displayed when the end user's desktop first appears.
/NOLOCK	Indicates that the desktop lockup object should be destroyed when the end user's desktop first appears.

---

## 8.5 Logoff and Shutdown Options

The Logoff and Shutdown icons on the user's desktop are used to invoke programs named TLOGOFF.EXE and TSHUTDOWN.EXE, respectively. Under normal circumstances, the PMLOGON shell calls these programs. However, if you are modifying or replacing the PMLOGON shell, you may wish to call these programs directly. For example, you may use the PMLOGON user exits to create a customized logon panel and to provide a Shutdown option for the user. You can call TSHUTDOWN.EXE directly from your customized logon panel.

You can modify the behavior of the Logoff and Shutdown programs with command line parameters. Both TLOGOFF.EXE and TSHUTDOWN.EXE support the following parameters:

- /Q - Prevents the LAN message box from appearing during logoff or shutdown.
- /N - Prevents the "Are you sure" dialog from appearing during logoff or shutdown.

See the *WorkSpace On-Demand Administrator's Guide* for information on how to include these parameters for your clients.

---

## 8.6 Replacing the PMLOGON Shell

The PMLOGON shell does not have to be used as the Workplace Shell for WorkSpace On-Demand. It can be replaced by modifying the `SET RUNWORKPLACE` statement in the CONFIG.SYS. This statement indicates what program to load as the Workplace Shell. For example, Netscape Navigator could be used as the shell program to provide the user with a browser interface by modifying the `SET RUNWORKPLACE` statement as shown in Figure 90.

```
SET RUNWORKPLACE=Z:\NETSCAPE\NETSCAPE.EXE -k <URL>
```

Figure 90. Modified SET RUNWORKPLACE Statement

The `-k` option starts Netscape Navigator in kiosk mode with no menu bar. When the client is started, it will boot directly to a browser desktop as shown in Figure 91.



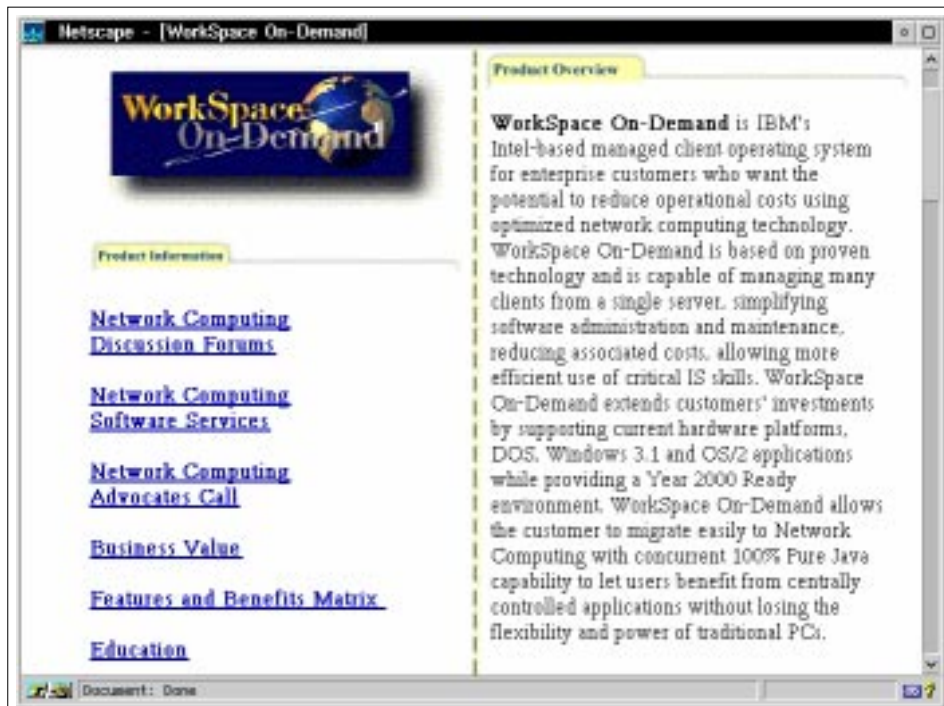


Figure 91. Netscape Navigator as the User Shell

This provides a simple way to create a secure kiosk with intranet or Internet access. Note, however, that in this environment, all features provided by PMLOGON are no longer be available. This includes user logon and verification and the ability to use user and application FIT files (since they only processed at logon and application startup by the default PMLOGON shell and its application launcher).



---

## Chapter 9. Supporting Network Adapters

Although WorkSpace On-Demand supports a large number of network adapters, only a few of them are configured and enabled at the time the product is installed. This chapter describes how to enable other network adapters as well as how to add unsupported adapters to your list.

---

### 9.1 Enabling Adapters Supported by WorkSpace On-Demand 2.0

When you install WorkSpace On-Demand 2.0 on your server, the installation procedure edits the RPL.MAP file and enables any server records for which the required support files (such as the drivers, CNF file and so on) exist on the server.

**Note**

The syntax of the RPL.MAP file is quite complex, and it is easy to inadvertently add or remove a character or space from a record in the file. If you do this, you can corrupt the RPL.MAP file, and the RPL service will fail to start.

For this reason, we *strongly* recommend that you make a backup copy of RPL.MAP before editing the file.

If you edit the RPL.MAP file, you will see a large number of server records, many of which have a semicolon character as the first character in the record. This semicolon disables the server record by making the RPL service treat it as a comment. In order to enable a server record, you must delete the semicolon, save the file, and restart the RPL service.

---

### 9.2 Enabling Adapters Supported by OS/2 Warp Server

IBM officially supports a large number of NDIS network adapters with OS/2 Warp Server. For a list of these adapters, see the OS/2 Warp Server home page at:

<http://www.software.ibm.com/os/warp/pspinfo/nic.htm>

or the WorkSpace On-Demand home page at:

<http://www.software.ibm.com/os/warp/workspace/nics/index.htm>

Although these adapters are supported by OS/2 Warp Server, you may need to take a number of steps to support them under WorkSpace On-Demand. OS/2 Warp Server only delivers the OS/2 device drivers, but a WorkSpace On-Demand client that uses the IEEE 802.2 boot mechanism requires both DOS and OS/2 device drivers during the boot process.

You must therefore manually install the DOS driver, the NIF file and its appropriate MSG file from a device driver diskette, option diskette or diagnostics diskette. These diskettes are typically delivered with the network adapter. See Section 9.3, "Enabling Unsupported Adapters" on page 242, for instructions on how to install the device drivers on your WorkSpace On-Demand server.

---

### 9.3 Enabling Unsupported Adapters

You can use any token-ring or Ethernet network adapter with WorkSpace On-Demand, provided that you can obtain functional DOS and OS/2 device drivers for the adapter.

**Note**

1. Enabling an unsupported adapter does not change the support statement for that particular adapter from IBM.
2. This procedure will enable an adapter for WorkSpace On-Demand only. For information on how to enable an adapter for DOS or OS/2, refer to the OS/2 Warp Server online documentation.

To enable an unsupported network adapter, you must complete the following steps:

- Obtain the most recent DOS and OS/2 NDIS drivers.
- Create a directory structure for the new drivers.
- Copy the device drivers and the related files into the directory structure.
- Create a PROTOCOL.INI file for the DOS environment.
- Create a boot block definition (CNF) file for the new adapter.
- Modify the NDISDD.PRO file.
- Create and enable the associated server record in the RPL.MAP file.

This procedure will work for most NDIS-compliant network adapters that support Remote IPL. The following pages describe each step in more detail.

As an example, we add the Madge Smart 16/4 AT Plus Ringnode Adapter. For the remainder of this chapter, we refer to this adapter as the Madge adapter.

**Note**

The Madge adapter is supported by WorkSpace On-Demand Manager 2.0, and you do not need to perform these steps to utilize the Madge adapter in your environment. However, the Madge adapter illustrates how to configure support for an unsupported network adapter and provides a guide to help you configure support for other adapters that you may require.

### 9.3.1 Obtain the DOS and OS/2 NDIS Drivers

Locate the most current DOS and OS/2 device drivers and other files for the network adapter you wish to add. You can usually obtain these files from the hardware vendor's Web site.

### 9.3.2 Install the DOS Driver

You must create new subdirectories within the \IBMLAN\RPL\DOS directory. For the Madge adapter, you should create the following directories:

```
\IBMLAN\RPL\DOS\MADGE
\IBMLAN\RPL\DOS\MADGE\OS2
```

**Driver Directories**

DOS drivers and their associated NIF files are usually placed in the \IBMLAN\DOSLAN\LSP\DOS directory.

You might wish to place the DOS device driver into its own subdirectory so that the directories can be easily accessed and restored from a backup. This might simplify re-creating support for the adapter in the event that migrating to a newer version of WorkSpace On-Demand re-creates the original directory structure.

Your choice of directory location *must* match the entries in the CNF file, which you will create and modify in a later step.

Copy the DOS driver and its associated NIF file into the directory you have selected. For the Madge adapter, we copied SMARTND.DOS to the \IBMLAN\RPL\DOS\MADGE directory. If there are DOS message files for your adapter, copy them into this directory as well.

### 9.3.3 Create a PROTOCOL.INI File for DOS

You must create a PROTOCOL.INI file for the DOS phase of the boot process. The easiest way to do this is to take an existing PROTOCOL.INI file from another adapter, and use it as a template. For example, when creating the PROTOCOL.INI file for the Madge adapter, we chose to copy the existing PROTOCOL.INI file for the IBM 16/4 Token-Ring Network Adapter and modify it. The PROTOCOL.INI file we copied is named IBMTOK.NIF and resides in the \BMLAN\RPL\DOS\TOKENRNG directory.

Figure 92 shows the completed DOS PROTOCOL.INI file for the Madge adapter.

```
;DOS PROTOCOL.INI used to boot OS/2 clients
;
;This PROTOCOL.INI is for the Madge Smart 16/4 AT Plus Ringnode Adapter

[PROTMAN_MOD]
  DriverName = PROTMAN$
[DXMJ0_MOD]
  DriverName = NETBEUI$
  Bindings = MADGE_MOD
[DXME0_MOD]
  DriverName = DXME0$
  Bindings = MADGE_MOD
[MADGE_MOD]
  DriverName = smartnd$
  PIO = YES
```

Figure 92. DOS PROTOCOL.INI for Madge Adapter

The bindings= entry in each section must be:

```
bindings=MADGE_MOD
```

This indicates that the physical DOS driver is identified in the smartnd section of the file.

The final section must be:

```
[MADGE_MOD]
  DriverName=smartnd$
```

This is the DOS driver section. It specifies that the name of the DOS driver is SMARTND.DOS (the DOS extension is assumed). This section indicates that all protocols should bind with the SMARTND.DOS driver.

Note the commented parameters at the end of the file. These were in the original file for the IBM Token-Ring adapter, but are not valid for the Madge adapter.

### 9.3.4 Install the OS/2 Driver

You must install the OS/2 driver and associated files into the correct place on your WorkSpace On-Demand server. Copy the OS/2 driver and the NIF file to the \BMLAN\RPL\BB20.US\IBMCOM\MACS directory. For the Madge adapter, these files are:

```
MDGND.OS2
MDGND.NIF
```

The NIF file contains important information that is used in the Network Adapter drop-down list on the Hardware page of the Client Definition notebook.

The example below shows a part of the MDGND.NIF file shipped with the Madge adapter:

```
[MDGND]
Type = NDIS_SINGL
Title = "Madge Fastmac Plus OS/2 NDIS MAC driver for
Smart 16/4 Ringnodes"
```

The text in the Title field is used to populate the Network Adapter drop-down list in the Client Definition notebook. You can modify the text between the double quotation marks if you wish to provide a more meaningful description.

#### Note

The \BMLAN\RPL\BB20.US\IBMCOM\MACS directory must contain a NIF file for your adapter, and the NIF file must contain a Title field. If the adapter you are configuring does not provide this information, you must provide it by creating your own NIF file.

If a message file exists for the network adapter, copy it into the \BMLAN\RPL\BB20.US\IBMCOM directory. For the Madge adapter, there was no message file.

### 9.3.5 Create a PROTOCOL.INI File for OS/2

You must create a PROTOCOL.INI file for the OS/2 phase of the boot process. In our example, we copied the PROTOCOL.INI file for the IBM Token-Ring adapter from the \IBMLAN\RPL\DOS\TOKENRNG\OS2 to the \IBMLAN\RPL\DOS\MADGE\OS2 directory and edited the file to include the correct driver information. Figure 93 shows the OS/2 PROTOCOL.INI file.

```
;OS/2 PROTOCOL.INI used to boot OS/2 clients
;This PROTOCOL.INI is for the Madge Smart 16/4 AT Plus Ringnode Adapter
;
[PROTMAN_MOD]
  DriverName = PROTMAN$
[DXNJ0_MOD]
  DriverName = NETBEUI$
  Bindings = MADGE_MOD
  PiggybackAcks = 0
  Sessions = 2
  NCBS = 6
  Names = 2
  Stacksize = 512
  Maxdatarcv = 2048
[DXNE0_MOD]
  DriverName = DXME0$
  Bindings = MADGE_MOD
[MADGE_MOD]
  DriverName = SMARTND$
  PIO = YES
```

Figure 93. OS/2 PROTOCOL.INI for Madge Adapter

The `bindings=` entry in each section must be:

```
bindings=MADGE_MOD
```

This indicates that the physical OS/2 driver that loads is defined in the `MADGE_MOD` section of the PROTOCOL.INI file.

The final section must be:

```
[MADGE_MOD]
  drivername=smartnd$
```

This is the OS/2 driver section. It specifies that the name of the OS/2 driver is `SMARTND.OS2`. Again, note the parameters commented out at the end of the file. These are from the original PROTOCOL.INI for the IBM Token-Ring adapter and are not valid for the Madge adapter.



### 9.3.6 Create a Boot Block Definition File

You must create a boot block definition (CNF) file for your adapter. The CNF file defines the contents of the boot block that is downloaded to the client by the server's RPL service.

Again, it is best to copy and modify an existing CNF file. For our example, we copied the CNF file for the IBM 16/4 Token-Ring adapter. This file is named BB2USNTR.CNF and resides in the \IBMLAN\RPL directory. We copied this file to create a new file named BB2USMDG.CNF, which is also in the \IBMLAN\RPL directory.

You must now modify the CNF file for the new adapter. As shown in Figure 94, we load everything exactly as the IBM Token-Ring adapter does. The only change is to load the correct adapter driver (SMARTND.DOS) and the correct PROTOCOL.INI file (the same PROTOCOL.INI file that we modified earlier to support the Madge adapter). Figure 94 on page 247 shows the completed CNF file for the Madge adapter, with the changes highlighted.

```
; OS/2 Boot Block Configuration
; Madge Smart 16/4 AT Plus Ringnode Adapter
;
RPL DOS\RPLBOOT.SYS
DAT DOS\MFSD30.SYS
ORG 1000H
LDR BB20.US\OS2LDR ~ OS2LDR UFSD.SYS MFSD30.SYS
DAT DOS\UFSD.SYS
DAT DOS\MADGE\OS2\PROTOCOL.INI
DAT C:\IBMLAN\DOSLAN\LSP\DXM.MSG
EXE C:\IBMLAN\DOSLAN\LSP\NETBIND.COM ~ ~ ~
; **NETBIOS and IEEE 802.2*****
; DRV C:\IBMLAN\DOSLAN\LSP\DXMT0MOD.SYS PBA=0~S=12~ST=12~C=14~O=N ~
; DRV C:\IBMLAN\DOSLAN\LSP\DXME0MOD.SYS ~ 6 ~
; **NETBIOS and IEEE 802.2*****
;
; ** NETBIOS*****
DRV C:\IBMLAN\DOSLAN\LSP\DXMJ0MOD.SYS ~ 6 ~
; ** NETBIOS*****
DRV C:\IBMLAN\DOSLAN\LSP\DXMA0MOD.SYS 001 ~ ~
DRV C:\IBMLAN\DOSLAN\LSP\DOS\SMARTND.DOS ~ 10 ~
DRV C:\IBMLAN\DOSLAN\LSP\PROTMAN.DOS /I: ~ ~
```

Figure 94. BB2USMDG.CNF

### 9.3.7 Edit the NDISDD.PRO File

Among other things, the NDISDD.PRO file determines the contents of the Network Adapter drop-down list on the Hardware page of the Client Definition notebook. In order for your new adapter to be selectable from the drop-down list, the file must contain a record for your adapter.

Figure 95 on page 248 shows an extract from the NDISDD.PRO file after adding a record for the Madge adapter. The new record is highlighted for clarity.

```
;NDIS device driver profile records are composed of 5 fields.
;They are described below.
;Field 1   This field specifies the DOS NDIS device driver name.
;Field 2   This field specifies the subdirectory name that
;           contains the configuration file(s) need to remote
;           boot the RIPL client.
;           d:\IBMLAN\RPL\IBMCOM\xxx\ contains the configuration
;           files needed to RIPL OS/2 clients
;           d:\IBMLAN\RPL\DOS\xxx\ contains the configuration file(s)
;           needed to RIPL both DOS and OS/2 clients.
;           d - the drive on which the RIPL component has been installed
;           xxx - the text located in the second field of a given line
;Field 3   NIF file name for OS/2 device driver
;Field 4   CNF file name id reference for server record id
;Field 5   Specifies the boot methods supported by the adapter
.....
ENDS2ISA.DOS  IBM_CLA      ENDS2ISA.NIF   OETCLA      802.2
IBMTRP.DOS   TOKENTRP    IBMTRP.NIF    OTKTRP      802.2,DHCP
SMARIND.DOS  MADGE       SMARIND.NIF   OTKMDGSMT   802.2
EL90X.DOS    3C90XPCI   EL90XIO2.NIF  OET3FE      802.2,DHCP
.....
```

Figure 95. NDISDD.PRO file

The server uses the NIF file specified in NDISDD.PRO to build the CONFIG.SYS and PROTOCOL.INI files for client workstations that you define as having this adapter installed.

Field 2 in the NDISDD.PRO record must match the name of the subdirectory that you created within the \IBMLAN\RPL\DOS directory. In our example, this is the MADGE subdirectory.

### 9.3.8 Edit the RPL.MAP File

You must edit the RPL.MAP file to create a server record for your new adapter. Figure 96 shows the server record for the Madge adapter.

```
; server records
yyyyyyyyyyy BB20USMDG.CNF 3 10 N - Workspace-On-Demand-2.0-Madge-Token-Ring - - , - R_BB20_US_OTKMDGSMT - -
```

Figure 96. RPL.MAP file - Server Record Entry

The record ID in the server record must match the key specified in field four of the NDISDD.PRO record. In our example, this is OTKMDGSMT.

### 9.3.9 Test Your Definition

The new adapter should now be selectable from the Hardware page in the Client Definition notebook. Test your adapter definition by performing the following steps:

- Stop and restart the RPL service.

If the RPL service fails to restart, it may be due to two problems:

- There may be an error in the server record you entered in RPL.MAP.
- There may be an error in the CNF file. Workspace On-Demand reads all CNF files when it starts the RPL service and checks that the drivers and other files listed in the CNF files exist and are valid. Check the paths and file names listed in your CNF file to ensure all the files are correctly named and reside in the directories indicated.
- Use the remote IPL requester configuration GUI to define a client using the new adapter. If the adapter is not visible in the Network Adapter drop-down list on the Hardware page in the remote IPL requester configuration notebook, it may be due to a number of problems:
  - Check the OS/2 NIF file, and ensure that it has a Title field. If the Title field is absent or contains only spaces, the adapter will not have an entry in the Override Network Adapter drop-down list.
  - Check the RPL.MAP file and ensure that the server record for the adapter is enabled.
  - Check the NDISDD.PRO file to ensure that field three matches the name of the OS/2 NIF file and field four matches the record ID of the server record in RPL.MAP.

After you successfully define a client with the new adapter, you should boot the client to ensure that all the necessary drivers and control files are valid and functioning correctly.

## Chapter 10. Supporting Additional Hardware

WorkSpace On-Demand 2.0 includes support for a number of different types of client workstations and devices. However, it is likely that you will need to provide support for clients and devices that are not supported by the standard WorkSpace On-Demand product. This chapter describes how you can add support for different network adapters and generate machine classes to support new types of client workstation under WorkSpace On-Demand.

---

### 10.1 Machine Classes

Simply stated, a machine class is the *structured* collection of device drivers and other support files that defines a WorkSpace On-Demand client's hardware environment. Note that in this definition, we use the word *structured* because the location of these files on the WorkSpace On-Demand server is important to successfully implementing a machine class.

The RIPL support provided by OS/2 Warp Server supported only a single, generic machine type. With the availability of WorkSpace On-Demand, however, you now have the ability to include multiple machine classes to support a broad range of different client workstations.

#### 10.1.1 When Do You Need a Machine Class?

A change in one component of a workstation configuration will require a different machine class in your WorkSpace On-Demand environment. Some of the actions that might require a new machine class are:

- Changing video adapters
- Adding clients with different bus types (ISA, MCA, PCI, EISA)
- Adding clients with different hard drive protocols (SCSI, IDE)
- Changing a client's keyboard to a different type
- Changing a client's mouse to a different type

In summary, it is probable that, whenever you change one or more device drivers or settings within a client's hardware configuration, you will need to create a new machine class. However, there are several exceptions to this rule:

- Changing network adapters
- Changing monitors (provided the video adapter does not change)

You can make these changes without changing a client's machine class since these items are defined separately from the machine class in the Client Definition notebook.

You must give careful thought to the types of hardware that you will support with WorkSpace On-Demand since different combinations of hardware devices can result in a proliferation of machine classes. For example, suppose that you decide to support hardware with five different types of video adapter and two different types of network adapter. This creates a total of ten possible combinations, which may result in ten different machine classes that you must create and maintain, as shown in Table 27.

Table 27. Machine Class Permutations for a Given Hardware Combination

	Video Adapter A	Video Adapter B	Video Adapter C	Video Adapter D	Video Adapter E
Network Adapter 1	MC1A	MC1B	MC1C	MC1D	MC1E
Network Adapter 2	MC2A	MC2B	MC2C	MC2D	MC2E

Note that the table above does not take into account any other differences between client workstations. For example, if you have some machines with SCSI drives and others with IDE drives, you may need to double the number of machine classes, resulting in twenty machine classes that you must create and maintain.

As you can see from the example shown in Table 27, it is important to minimize the number of different hardware combinations on your client workstations. This will significantly reduce the effort and cost associated with creating and maintaining machine classes in your WorkSpace On-Demand environment.

### 10.1.2 What is in a Machine Class?

As already mentioned, a machine class is a structured collection of device drivers and other hardware-specific support files that define a client workstation's hardware configuration. These files are located in specific locations in a directory structure known as the *machine class file structure* or *machine class directory* on the WorkSpace On-Demand server.

Note that there are rigid rules that define the machine class file structure and the location of files within it. When creating or modifying machine classes, you must follow these rules in order for your machine class to be usable by WorkSpace On-Demand.

The structured collection of device drivers and other support files that comprise a machine class include:

- Operating system configuration files
  - CONFIG.SYS
  - AUTOEXEC.BAT
  - OS2.INI
  - OS2SYS.INI
- WIN-OS/2 configuration files
  - WIN.INI
  - SYSTEM.INI
- A machine class FIT file
- Device drivers and control files such as:
  - Video adapter
  - Sound card
  - DASD type
  - CD-ROM
  - Keyboard
  - Mouse

Depending on a client's hardware configuration, the content of these files will differ among WorkSpace On-Demand clients. It is these differences that must be *documented* and then incorporated into a new machine class in order to support a new WorkSpace On-Demand client hardware platform.

### 10.1.3 What is Outside a Machine Class?

There are two hardware configuration items that are not part of the client's machine class. These are:

- Network adapter
- Monitor

You can define and change the network adapter for a client in the Client Definition notebook separately from the machine class. Therefore, you can have a number of clients with the same machine class but with different network adapters.

Similarly, you can specify the monitor type and video resolution in the Client Definition notebook separately from the machine class. Note, however, that the list of entries available in the Client Definition notebook depends upon the video adapter in the client, which *does* form part of the machine class. For

any machine class, you will therefore have a finite range of monitors and video resolutions from which to choose.

#### 10.1.4 Machine Classes Shipped with WorkSpace On-Demand

WorkSpace On-Demand provides several different machine classes with the product. However, it should be apparent that the machine classes shipped with WorkSpace On-Demand may not address all of the WorkSpace On-Demand client hardware configurations needed to support WorkSpace On-Demand in a particular user's environment.

For example, suppose that two IBM PC 730 machines are identical in every aspect except that each machine has a different video adapter chip set. Consider the fact that each machine requires its own separate machine class, and you will appreciate the importance of machine classes in successfully implementing WorkSpace On-Demand. Table 28 shows the machine classes that are shipped with the WorkSpace On-Demand 2.0 product.

Table 28. Predefined Machine Classes in WorkSpace On-Demand 2.0

Machine Class	Model / Bus	Video Type
GENGRADD	Generic ISA/PCI GRADD. Most PCs with an ISA / PCI bus and SVGA display adapter that is VESA 1.2 Compliant.	VESA 1.2
IBM300GL	IBM Personal Computer 300GL model 6282-38U	Cirrus Logic 5436/5446
IBM300G2	IBM Personal Computer 300GL model 6272-880	Cirrus Logic 5436/5446
IBM350	BM Personal Computer 350 model 6586-57H	S3 TRIO 64/64V+
IBM730	IBM Personal Computer 730 model 6877-KAZ	S3 TRIO 64/64V+
IBM750	IBM Personal Computer 750 model 6887-86B	S3 TRIO 64/64V+
ISAVGA	Any PC with an ISA bus and VGA display adapter that is supported by OS/2 Warp Version 4	Generic VGA
MCAVGA	IBM PCs (486+) with Micro Channel Bus and standard VGA with SCSI/ESDI support	VGA
MCAXGA	BM PCs (486+) with Micro Channel Bus and XGA video with SCSI/ESDI support	XGA



You can use the OS/2 Warp Server GUI to create additional machine classes to accommodate variations in the type of network adapter, monitor, video resolution, and other configuration points such as keyboard, mouse, hard disk adapter and remote boot method needed by a client. However, this flexibility may not be sufficient to adequately define all the types of hardware that you wish to use as WorkSpace On-Demand clients.

There are ways to expand the types of client machines supported for use as WorkSpace On-Demand clients. One way is to engage an IBM Services Provider or IBM Business Partner to develop and test machine classes to meet your specific needs. The other way, which we will explore in the rest of this chapter, is to develop your own WorkSpace On-Demand machine classes.

### 10.1.5 Additional Machine Classes

The CD-ROM that accompanies this redbook contains fourteen machine classes that provide support for video adapters not supported in the basic WorkSpace On-Demand 2.0 product. These additional machine classes are offered strictly on an as-is basis, with no support nor warranty whatsoever, either expressed or implied.

These new machine classes include a read-only video subdirectory and a hardware-specific video subdirectory for each machine class. The CD-ROM also includes a modified version of the MCLASS.RSP file that you can use to integrate support for these new machine classes into your WorkSpace On-Demand 2.0 environment. If you do wish to enable support for only some of these machine classes, you must edit the MCLASS.RSP file to remove the video adapters that you do not require.

#### Note

These machine class files are packaged on the CD-ROM in the same directory structure as they appear on a WorkSpace On-Demand server. However, there is no direct one-to-one correspondence between the hardware-specific video subdirectories and the read-only video subdirectories because several of the machine classes differ only in the amount of video RAM that they require. In this case, the read-only drivers used by both machine classes are identical and are packaged in a single subdirectory.

After reading and reviewing Section 10.5, “Creating the New Machine Class” on page 285, you should understand how to implement these new machine classes.

### 10.1.6 Planning Your Machine Classes

You should give careful consideration to the machine architectures and hardware components that you plan to support with WorkSpace On-Demand. The more different machine types you use, the more machine classes you will need to define. With every machine class that you define, you will increase the effort to implement WorkSpace On-Demand since you must test each machine class with all the applications you want to run in this environment.

You can reduce the work of implementing, testing and supporting your WorkSpace On-Demand environment by limiting the combinations of critical hardware components, and therefore the number of machine classes, to a minimum.

When planning to deploy a large number of WorkSpace On-Demand clients, you should consider the following:

- Try to avoid a wide range of hardware configurations in order to minimize the implementation, testing and administration effort and the associated costs.
- Try to select a single hardware platform to support. After you have made your hardware decision, buy the number of machines needed for the planned installation cycle and store these machines during the roll-out process. If cost or other considerations prohibit a single purchase, try to buy machines in a small number of large lots rather than buying them in small groups over time. This will reduce the likelihood of small but significant differences in hardware configuration.
- For extremely large projects, deploying WorkSpace On-Demand across the entire enterprise may take several years and may make it impossible to base the entire roll-out on a single hardware platform. In such cases, you should plan the deployment in a series of cycles, year by year, and expect to use a new hardware platform in each cycle since hardware manufacturers are likely to update their product lines during the period of the roll-out. You will need to implement and test WorkSpace On-Demand on each new hardware configuration and its resulting machine class.
- There will probably be new system software releases during this time, and implementing WorkSpace On-Demand will therefore require a combination of hardware and software testing. Not only must you take care of the installation of new machines but also plan for migration of existing machines, already deployed, to the new software release.
- Install a test environment at your central site that reflects your enterprise topology. In this test environment, attempt to have examples of every

supported hardware configuration so you can test your current WorkSpace On-Demand machine classes on all supported hardware.

- Be aware that you will need to support a machine class once it has been rolled out into the enterprise. You will need to support the hardware with every new release of the WorkSpace On-Demand product and system software components running under WorkSpace On-Demand.

---

## 10.2 The Machine Class Create Utility

If the predefined machine classes shipped as part of WorkSpace On-Demand 2.0 do not support your particular hardware environment, you can use the Machine Class Create utility to create a new machine class that meets your particular requirements.

Complete instructions for using the Machine Class Create utility are located in the *WorkSpace On-Demand Release 2.0 Administrator's Reference* that is shipped as part of WorkSpace On-Demand 2.0's online documentation.

The Machine Class Create utility greatly reduces the effort required to build a new machine class, but it is initially limited to the predefined hardware configurations and their associated device drivers and other configuration files packaged with WorkSpace On-Demand 2.0. However, you can extend the hardware support provided by the Machine Class Create utility to incorporate new hardware devices, as described in Section 10.3, "Creating Your Own Machine Class" below.

---

## 10.3 Creating Your Own Machine Class

By using one of the machine classes supplied with WorkSpace On-Demand as a prototype, you can create your own machine class to accommodate the specific hardware requirements of your WorkSpace On-Demand clients. This process is known as *machine classing*.

The steps necessary to configure hardware support will differ depending on the type of hardware that is to be supported. For example, configuring support for a specific video subsystem requires the most extensive modifications to device driver directories, system files such as OS2.INI and video configuration files such as the video FIT extension file. Other supported hardware, such as a SCSI controller, may only require changes to the WorkSpace On-Demand client's CONFIG.SYS file.

### 10.3.1 Hardware Requirements

In order to create your own machine class, you will need to provide a test hardware environment as shown in Figure 97.

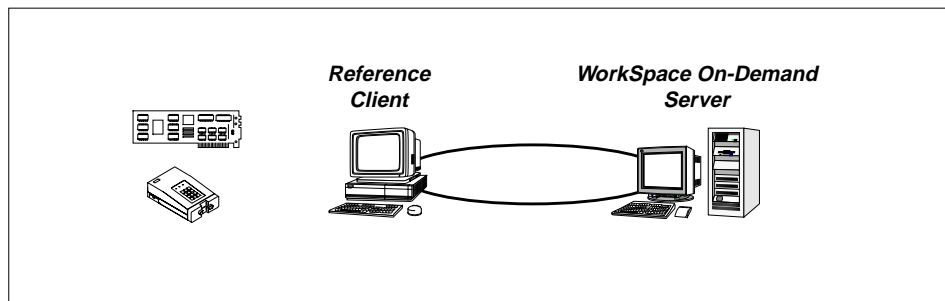


Figure 97. Creating a Machine Class - Test Hardware Environment

You will need a client machine of the type that you wish to support, including all of the hardware adapters and peripheral devices for which you wish to incorporate support in your new machine class. This machine is known as a *reference client*. The reference client should include the following:

- A BIOS that allows it to boot from the network as well as from local storage devices.
- A network adapter that is supported by WorkSpace On-Demand 2.0 or for which you have already generated the appropriate definitions as described in Chapter 9, "Supporting Network Adapters" on page 241.
- Its own local disk drive, with sufficient disk space to install OS/2 Warp 4.

When creating your new machine class, you will use the reference client as both a WorkSpace On-Demand client and a traditional "fat" client in standalone mode.

You will also need a WorkSpace On-Demand server with a network adapter that allows it to connect to the reference client.

### 10.3.2 Feasibility Tests

Use the following two tests to confirm if you can *easily* create a machine class to support the type of machine you wish to use for a WorkSpace On-Demand client.

1. Can you remotely boot the target client using the generic ISAVGA or MCAVGA machine classes?

This is a *low-level* machine class providing the most basic hardware support functions. If your client can pass this test, it means that the system unit itself is capable of supporting a basic WorkSpace On-Demand configuration and that your network adapter is supported.

**Note**

You may need to configure support for your client's network adapter before proceeding with this test. See Chapter 9, "Supporting Network Adapters" on page 241, for a complete discussion of configuring network adapters.

2. Can you successfully install and configure OS/2 Warp Version 4, including the devices and drivers that you require, on the target client?

Successfully executing this test verifies that you can isolate and extract the required device driver, INI and other support files from a working system that are necessary to create a viable machine class that supports the hardware devices that you wish to use.

**Note**

When testing, it may be possible to develop a machine class for hardware that fails either of these tests. However, the effort required will be much greater than for those machines which pass these tests. There is also the issue that machines with devices and drivers that are not supported by OS/2 Warp Version 4 (See test #2) may not be supported.

### 10.3.3 Useful Tools for Machine Classing

There are a number of different types of tools and utilities that you will find useful when creating and testing a machine class. These categories are described in the following sections along with some specific examples.

#### 10.3.3.1 File Management Utility

A file management utility such as File Commander is very useful for machine classing and installing new applications. We have included a shareware copy of File Commander on the CD-ROM that accompanies this redbook.

#### 10.3.3.2 File Comparison Utility

The methodology described in the remainder of this chapter requires you to compare the contents of a number of files and directories. A file comparison utility is therefore very useful. WorkSpace On-Demand 2.0 provide utilities

named MKTMPENV and DIFFTREE that allow you to copy and compare directory trees, but we have chosen to use two other shareware utilities:

- Graphical File Compare compares the contents of two files.
- PMDMatch compares the contents of two directory trees.

Each of these utilities is included on the CD-ROM that accompanies this redbook.

#### **10.3.3.3 INI File Editor**

When configuring support for certain types of hardware devices such as video adapters, you must be able to examine and modify the contents of the OS2.INI file. Since INI files contain binary data and therefore cannot be edited using a regular text editor, you must use a specialized INI editor such as Unimaint or FM/2. These utilities are included on the CD-ROM that accompanies this redbook.

We have also included REXX command files named INICREAT and COPYAPP on the CD-ROM. These utilities are used to create an INI.RC file for a new machine class, as described later in this chapter.

#### **10.3.3.4 File Monitoring Utility**

In situations where the number of files required to support a machine class is very large, testing a machine class can become a complicated exercise. It is often useful to monitor the files that a client attempts to open on the WorkSpace On-Demand server to determine whether particular files reside in the correct locations. FWatch is a tool that enables you to monitor file activity on the client by logging file I/O activity to the client's serial port.

FWatch is included on the CD-ROM that accompanies this redbook. In addition, you will need to provide a null-modem cable and a workstation with an ASCII terminal capability on which to monitor the file I/O activity. For this task, we have also included a shareware copy of Logicomm on the CD-ROM that accompanies this redbook.

When setting up FWatch, you must add a `DEVICE=` statement for `FWATCH.SYS` to the client's `CONFIG.SYS` file. You must add this statement after the `COM.SYS` device driver, and be sure to comment out the `DEVICE=` statement for `PROTDISK.SYS`.

**Note**

At the time this redbook was written, using early drivers of the WorkSpace On-Demand Manager 2.0 code, we encountered a problem when using the diskette drive on a client in conjunction with MKTEMPENV, which manifested itself as a failure to capture a file copied in from the floppy disk drive to the WorkSpace On-Demand 2.0 server.

---

## 10.4 Machine Classing Methodology

Under WorkSpace On-Demand 1.0, machine class creation was a purely manual process and consisted of two distinct phases:

- Determining the files required to support a particular hardware configuration
- Placing those files in the correct locations on the WorkSpace On-Demand server

WorkSpace On-Demand simplifies these tasks somewhat by introducing a number of utilities, such as the Machine Class Create utility, which allows you to create a machine class file structure by specifying combinations of known configuration items. You can therefore "mix and match" configuration items from the machine classes shipped with WorkSpace On-Demand 2.0 in order to create a new machine class that supports your client hardware.

If you wish to provide support for a new device, however, you must carry out the two phases listed above, and provide the necessary information to enable the Machine Class Create utility to include your device's configuration files when it creates a machine class file structure. The remainder of this chapter describes a methodology that will help you to identify the necessary files and place them in the correct locations so that they can be incorporated into the machine classes created by the Machine Class Create utility.

The methodology consists of the following major tasks:

1. Install OS/2 Warp 4 on a reference client using a basic hardware configuration and only VGA graphics support.
2. Define the client on your WorkSpace On-Demand server using the ISAVGA machine class. Modify the client's CONFIG.SYS to include any tools that you will use (such as FWatch), and comment out the PROTDISK.SYS driver to allow access to the client's local drive.

3. Create a user ID on the server and grant access for that user ID to any tools that you will use on the client (such as FWatch).
4. Create temporary directories on the server to hold configuration files.
5. Reconfigure the reference client to boot from the network, logon and capture the system state by copying the system configuration files from the client's local drive to a temporary directory on the server.
6. Reconfigure the reference client to boot from its local drive, and use the OS/2 Warp 4 **Selective Install** utility to configure all of the required hardware support for your client.
7. Reconfigure the reference client to boot from the network, and capture the new system state by copying the system configuration files from the client's local drive to a different temporary directory on the server.
8. Create subdirectories on the server to hold the drivers and other configuration support files.
9. Create the necessary configuration template files to support your new hardware configuration.
10. Copy drivers, configuration templates and other configuration support files to their correct locations on the server.
11. Modify the MCLASS.RSP file on your server so that your new configuration files can be detected by the Machine Class Create utility.
12. Build and test your new machine class using the Machine Class Create utility.

We will use the same physical machine as both a traditional "fat" client with OS/2 Warp 4 and a WorkSpace On-Demand client. This dual role can be achieved simply by reconfiguring the machine's BIOS to boot either from local storage devices or from the network.

The remainder of this chapter covers the step-by-step procedures needed to enable support for the Cirrus Logic CL-GD546X video graphics adapter. We chose this adapter because it is used with the newer Pentium II-equipped IBM Personal Computer 300GL machines, such as the 6561-42U. The IBM300GL machine class shipped with WorkSpace On-Demand 2.0 supports the Cirrus Logic GD-5436/46 adapter chip set.



We believe that documenting the specific steps needed to install a "real" hardware device is of more value than discussing the methodology in general terms. However, you should be aware that different vendors implement different installation strategies. We have made an effort to point out the most common discrepancies and to suggest alternative procedures where appropriate. We believe that this methodology can be used, with some modifications, for most types of hardware that you will encounter.

#### **10.4.1 Install OS/2 Warp 4 on the Reference Client**

You must first install OS/2 Warp 4 on the reference client, specifying only basic configuration details and not adding support for any additional hardware devices. We will use this hardware configuration as a "baseline" in order to determine the specific files that are added and modified when we configure the system to support additional devices in a later step.

In our Cirrus Logic CL-GD546X example, the only difference between the basic configuration and our desired configuration is the video adapter. We were therefore able to accept the OS/2 installation programs auto-detect choices, except that we overrode the video adapter support to specify VGA as the primary video adapter.

#### **10.4.2 Define the Reference Client to WorkSpace On-Demand**

You must now define the reference client on your WorkSpace On-Demand server, as described in Chapter 6, "Working with Client Workstations" on page 137. Specify the ISAVGA or MCAVGA machine class depending on the client's bus type in the machine class drop-down list on the Hardware page of the Client Definition notebook. When creating the machine class for the Cirrus Logic CL-GD546X, we created a client named NEWTEST.

After defining the client, edit the client's CONFIG.SYS file to comment out the PROTDISK.SYS driver. In our test example, the client's CONFIG.SYS file is located in the \IBMLAN\RPL\MACHINES\NEWTTEST directory. Locate the PROTDISK.SYS driver and comment out its entry as shown below:

```
REM DEVICE=Z:\OS2\BOOT\PROTDISK.SYS
```

Commenting out this driver enables read/write access to files on the reference client's local hard disk. This will enable you to copy configuration files to the server in a later step.

#### **10.4.3 Create a User ID**

You must now define a user ID on the server and grant access to the tools and utilities that you will use when creating your new machine class. At the

very least, you should grant access to a command prompt, but you may wish to include other tools such as GFC.

**Hint**

We recommend that you create the user ID with administrator authority to eliminate potential problems with file access permissions.

In our machine classing example, we created a user ID named LOTTEST, with administrator authority.

#### 10.4.4 Create Temporary Directories on the Server

WorkSpace On-Demand will not allow you to copy configuration files directly from the client to read-only areas in the client's boot image. You must therefore create a temporary holding space for such files.

You must create two separate directories on your server. One will contain the configuration files that define the system state of your client in its basic configuration. The other will contain the files that define the system state of the client in its enhanced configuration, with all required hardware device support included. You will then use these two directories to compare the system states and determine the files that were added or modified when configuring support for your hardware devices.

In our machine classing example, we booted the reference client from the server, logged on and issued the following commands:

```
MD Z:\VIDVGA
MD Z:\VIDSVGA
```

These commands created two directories within the client's boot image on the server. The client's redirector maps these files to subdirectories within the \BMLAN\RPLUSER\*client\_name* directory, where you have read/write access to the files from the client.

#### 10.4.5 Capture the Basic System State

Copy the following files from the reference client's hard disk to a directory on the server.

```
C:\CONFIG.SYS
C:\OS2\OS2.INI
C:\OS2\OS2SYS.INI
C:\OS2\MDOS\WINOS2\SYSTEM.INI
C:\OS2\MDOS\WINOS2\WIN.INI
```

In our machine classing example, we copied these files to the Z:\VIDVGA directory.

Now that these files have been copied, you should log off, shut down the reference client, and disconnect its session to the server with the following command:

```
NET SESS \\NEWTEST /D /Y
```

Disconnecting the session will close any server files that remain open and avoid the possibility of corrupting any files.

#### 10.4.6 Configure Hardware Support for OS/2 Warp 4

Turn off your reference client and install the required hardware devices. In our machine classing example, the Cirrus Logic CL-GD546X adapter is imbedded in the machine's motherboard and required no hardware installation.

Reconfigure the reference client's BIOS to boot from local storage, and reboot the client. Run the OS/2 Warp 4 Selective Install utility or other appropriate software as instructed in the documentation supplied with your hardware devices. Reboot the client after installation, and ensure that the hardware devices are functioning correctly.

In our machine classing example, we verified correct installation by observing the contents of the screen resolution, screen refresh rate, and monitor type dialogs in the System object. Remember that the VGA mode provides only a single resolution, refresh rate and a default monitor type, whereas the Cirrus Logic CL-GD546X driver provides multiple choices.

##### Note

When configuring a video graphics adapter on the reference client using OS/2 Warp 4, *do not* select or configure a specific resolution, refresh rate or monitor choice. Selecting any of these parameters will modify the reference client's OS2.INI file with an application key that updates and sets new, fixed default values.

In a later step, you will use the reference client's OS2.INI file to create a new machine class. Applying fixed default values at this time eliminates the flexibility to specify a new client's video resolution, refresh rate or monitor type when creating a new client with the new machine class.

If you intend to provide WIN-OS2 support for your Workspace On-Demand clients, you should ensure that you test the new video graphics adapter with WINOS2 windows and full-screen modes.

## 10.4.7 Capture the Enhanced System State

Reconfigure the reference client's BIOS to boot from the server, reboot the client and log on. You must now capture the system state by copying the system configuration files to the server.

### 10.4.7.1 General Hardware Device Support

When creating a machine class to support any hardware device, you will need to copy the following OS/2 system and other configuration files to the server:

```
C:\CONFIG.SYS
C:\OS2\INI.RC
C:\OS2\OS2.INI
C:\OS2\OS2SYS.INI
C:\OS2\SVGADATA
C:\OS2\VIDEO.CFG
C:\OS2\MDOS\WINOS2\SYSTEM.INI
C:\OS2\MDOS\WINOS2\WIN.INI
```

#### Note

IBM Thinkpads, or other laptop computers routinely support additional or multiple monitors. For example, the IBM Thinkpad 760XD requires eight separate SVGADATA files. For this reason, there may be additional files, other than the SVGADATA.PMI file, that you must copy.

In our machine classing example, we created a SYSFILES subdirectory within the Z:\VIDSVGA directory to hold these files.

### 10.4.7.2 Video Adapter Support

When creating a machine class to support a video adapter, there are a number of additional files that you must copy to the server. In our machine classing example, we created a DSPFILES subdirectory within the Z:\VIDSVGA directory to hold these files.

In many cases, the files that support a particular video adapter are unique to that adapter. Fortunately, the DSPINSTL utility that installs OS/2 Warp display drivers creates a log file that lists all the files copied to the reference client's hard disk during the installation process. This file is named DSPINSTL.LOG,

and it resides in the \OS2\INSTALL directory on the reference client's boot drive.

**Note**

If you are installing a Matrox video graphics adapter, the installation log file will be named DISPLAY.LOG. However, it resides in the same directory and contains the same information as the DSPINSTL.LOG file.

The DSPINSTL.LOG file is cumulative. If you perform multiple video driver installations, each subsequent installation's log entries are appended to a new section at the end of the file. For this reason, you should ensure that you use only the last section of the DSPINSTL.LOG as a guide when copying files to the appropriate directories on the server.

Edit the DSPINSTL.LOG file on the reference client, using a normal text editor, and move to the last section in the file, which documents the last display driver installation. In our machine classing example, this section of the file appears as shown in Figure 98 on page 268. Note that Figure 98 includes a number of annotations, which we will use when discussing which files should be copied to the server.

```

E:\EXE - DSPINSTL.LOG
File Edit Options Help
-----
Installing "CIRRUS, GD546X" on 08/14/1998 10:38:44
C:\OS2\NDOS\WINOS2\SYSTEM\DC1546X.DRV
C:\OS2\NDOS\WINOS2\SYSTEM\546XSL.DRV
C:\OS2\NDOS\WINOS2\SYSTEM\546XFS.DRV
C:\OS2\NDOS\WINOS2\SYSTEM\CGA40MOA.FON
C:\OS2\NDOS\WINOS2\SYSTEM\CGA80MOA.FON
C:\OS2\NDOS\WINOS2\SYSTEM\CGURE.FON
C:\OS2\NDOS\WINOS2\SYSTEM\EGA40MOA.FON
C:\OS2\NDOS\WINOS2\SYSTEM\EGA80MOA.FON
C:\OS2\NDOS\WINOS2\SYSTEM\SERIFE.FON
C:\OS2\NDOS\WINOS2\SYSTEM\SSERIFE.FON
C:\OS2\NDOS\WINOS2\SYSTEM\SYMOOLE.FON
C:\OS2\MONITOR.DIF <==NOTE==
C:\OS2\INSTALL\VCFGCID.CMD <==REMOVE=
C:\OS2\INSTALL\GETMONID.CMD <==REMOVE=
C:\OS2\DLL\BVHSVG.A.DLL
C:\OS2\DLL\IBMGPMI.DLL
C:\OS2\DLL\VCFGMRI.DLL
C:\OS2\DLL\VIDE0CFG.DLL <==REMOVE=
C:\OS2\DLL\VIDE0CFG.DLL
C:\OS2\DLL\VIDE0PMI.DLL
C:\OS2\DLL\OENPMI.DLL
C:\OS2\DLL\CIRRUS.DLL
C:\OS2\NDOS\YSVG.A.SYS
C:\OS2\NDOS\VPRPMI.SYS
C:\OS2\SCREEN01.SYS
C:\OS2\SVG.A.EXE <==REMOVE=
C:\OS2\VIDEO.CFG <==NOTE==
C:\OS2\SVGADATA.PMI <==NOTE==
C:\OS2\DLL\VCFGMRI.DLL
C:\OS2\DLL\MPV10SYS.DLL

```

Figure 98. DSPINSTL.LOG

You can use the entries in the DSPINSTL.LOG file to create a command file that will copy the required files to the temporary directory on the server. In our machine classing example, this is the Z:\VIDSVGA\DSPFILES directory.

However, there are a number of files that must not be copied to the server, and several others that are harmless but will simply waste space if they are copied to the server. These files are discussed below.

### **MONITOR.DIF**

Do not copy this file to the server.

The monitor definitions file, MONITOR.DIF, is used in conjunction with the SVGADATA.PMI file to create an enumerated list of the video modes supported by the combination of the video graphics adapter and the video monitor selected for this client. This enumerated list is stored in the VIDEO.CFG file.

When you boot a new client, WorkSpace On-Demand uses its own copy of MONITOR.DIF, which resides in the \IBMLAN\RPL\BB20.US\OS2\ directory on the server, to validate the integrity of the client's VIDEO.CFG file. If you map a vendor-unique copy of the MONITOR.DIF file into a new machine class, the new client's VIDEO.CFG file is created using the mapped, vendor-specific MONITOR.DIF file.

When the server performs an integrity check at boot-time, the integrity check will fail because the client's initial VIDEO.CFG file was created using a different version of the MONITOR.DIF file. The result is that the client uses the lowest resolution supported by its video adapter.

For this reason, you *should not* copy MONITOR.DIF from the reference client to the server. Instead, you should allow WorkSpace On-Demand to use its default MONITOR.DIF file.

**Hint**

It is important to understand the relationship between the video configuration files VIDEO.DIF, VIDEODO.CFG and SVGADATA.PMI, and the system file, OS2.INI, when determining the video mode in which a remote IPL requester will boot. Appendix D, "Video Handling" on page 339, describes how these files interact.

**SVGADATA.PMI**

The SVGADATA.PMI file defines the video modes that a particular video graphics adapter's chip set can support. It is also used by WorkSpace On-Demand 2.0 to populate the Video resolution drop-down list on the Hardware page in the Client Definition notebook.

You can create SVGADATA.PMI dynamically by running SVGA ON from a DOS command prompt, or it may be supplied by the adapter manufacturer, as is the case with the file for the CL-GD546X chip set that we use in our machine classing example.

**Note**

The fact that the SVGADATA.PMI is provided by the manufacturer rather than being created dynamically is documented in the README.1st file that accompanies the other software supplied with the adapter. Also, you can infer this fact by noting that the file name appears in the DSPINSTL.LOG file.

See Appendix D, “Video Handling” on page 339, for more details on how SVAGADATA.PMI is used in conjunction with the VIDEO.DIF and VIDEO.CFG files.

In our machine classing example, the SVGADATA.PMI file is copied to the server although it will later be copied into a hardware-specific video subdirectory rather than into the read-only video directory with the other video device drivers.

### **VIDEO.CFG**

The VIDEO.CFG file contains the video modes that are supported by a specific combination of video adapter and monitor. This information is created dynamically by the interaction of SVGADATA.PMI and VIDEO.DIF.

WorkSpace On-Demand uses this file to initially create a machine class, but the file will reside in a hardware-specific video subdirectory rather than the read-only video directory with the other video device drivers.

### **Vendor-Specific and Other Files**

The software that accompanies some video adapters may include vendor-specific utilities and other files that are not discussed here. You should consult the documentation that accompanies your video adapter and its software. As a general rule, files with the following extensions *should not* be copied from the reference client:

- \*.EXE
- \*.SYM
- \*.CMD
- \*.DIF

The documentation that accompanies the CL-GD546X supporting software used in our machine classing example indicates that the files VCFGID.CMD, GETMONID.CMD and RXVIDCFG.DLL are supplied by the vendor to support CID-based installations. We do not copy these files to the server. SVGA.EXE is not used because the vendor provides an SVGADATA.PMI;, so we did not copy SVGA.EXE to the server.

### **Copying Files to the Server**

Using DSPINSTL.LOG as a template and following the guidelines we have just discussed, it is possible to build a command file that copies these files from the reference client to the temporary subdirectory on the server. Figure 99 on page 271 shows an example of a command file created from the DSPINSTL.LOG file. We used this command file in our machine classing example to copy the necessary files from the reference client to the server.



```

COPY C:\OS2\INDOS\WINOS2\SYSTEM\DCI546X.DRV Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\INDOS\WINOS2\SYSTEM\546XSL.DRV Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\INDOS\WINOS2\SYSTEM\546XFS.DRV Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\INDOS\WINOS2\SYSTEM\CGA40N0A.FON Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\INDOS\WINOS2\SYSTEM\CGA80N0A.FON Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\INDOS\WINOS2\SYSTEM\COURE.FON Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\INDOS\WINOS2\SYSTEM\EGA40N0A.FON Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\INDOS\WINOS2\SYSTEM\EGA80N0A.FON Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\INDOS\WINOS2\SYSTEM\SSERIFE.FON Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\INDOS\WINOS2\SYSTEM\SSERIFE.FON Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\INDOS\WINOS2\SYSTEM\SYMBOL.E.FON Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\DLL\IBMHVGA.DLL Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\DLL\IBMGPMI.DLL Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\DLL\VCFGMR1.DLL Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\DLL\VIDEOCFG.DLL Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\DLL\VIDEOPMI.DLL Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\DLL\VOEMPMI.DLL Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\DLL\CIRBUS.DLL Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\INDOS\VSVGA.SYS Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\INDOS\VPSPMI.SYS Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\SCREEN01.SYS Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\VIDEO.CFG Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\DLL\VCFGMR1.DLL Z:\VIDSVGA\DSPPFILES
COPY C:\OS2\DLL\NPVIDSYS.DLL Z:\VIDSVGA\DSPPFILES

```

Figure 99. Batch File Copy Command

After you have copied these files to the temporary directories on the server, you can copy the files to the correct locations on the server from where they can be used by the Machine Class Create utility.

## 10.4.8 Create New Subdirectories

Depending on the type of hardware device you are supporting with your new machine class, you may need to create one or more new subdirectories on the server to accommodate the device drivers and support files.

### 10.4.8.1 General Hardware Device Support

In many cases, you can copy device drivers into existing subdirectories within the client's boot image, such as the \BMLAN\RPL\BB20.US\OS2 directory. Whether you create new subdirectories or use existing subdirectories depends on the way you wish to set up your client's boot image and the requirements of the hardware devices you wish to support.

### 10.4.8.2 Video Adapter Support

With WorkSpace On-Demand 2.0, you must create two new video subdirectories to support a new video adapter. These subdirectories are:

- Read-Only Video Subdirectory

This directory contains read-only device drivers and associated files. In our machine classing example, we created a CL5465 subdirectory within the \BMLAN\RPL\BB20.US\OS2\VIDEO directory on the server.

- **Hardware-Specific Video Subdirectory**

This directory contains OS/2 system files and video configuration files that will be copied into the new machine class file structure by the Machine Class Create utility. In our machine classing example, we created a CL5465 subdirectory within the \BMLAN\RPL\MACHINES\VIDEO directory on the server.

We suggest that you use another hardware-specific video subdirectory from an existing machine class to provide a "skeleton" for this new subdirectory. These files will provide a useful guide to the files that you require in your new subdirectory and will help to ensure that you have all the files that you need to support your video adapter.

For example, the \BMLAN\RPL\MACHINES\BB20.US\VIDEO\CL5436 directory contains files that are almost identical to those needed for the CL5465 video graphics adapter. As a configuration aid, we copied all the files from the existing \BMLAN\RPL\MACHINES\BB20.US\CL5436 directory, including the OS2 subdirectory and its contents, into the newly created \BMLAN\RPL\MACHINES\BB20.US\CL5465 directory. Remember to change the name of the FIT file from CL5436.FIT to CL5465.FIT.

With these two new subdirectories in place, we are ready to capture the new video graphics adapters configuration information.

## **10.4.9 Create Configuration Template Files**

Some of the files copied from the reference client will serve as a basis for creating the system files and configuration files necessary to support the new hardware configuration. Other files, such as VIDEO.CFG, can be copied directly to the newly created subdirectories in the client's boot image.

### **10.4.9.1 CONFIG.HW**

CONFIG.HW is one of the files used to build a WorkSpace On-Demand 2.0 client's CONFIG.SYS file. It contains the statements that support a particular hardware device. You must create the CONFIG.HW file by determining the statements in the CONFIG.SYS file that relate specifically to the hardware you wish to support with your new machine class.

Figure 100 shows the Graphical File Comparison utility (GFC) being used to compare the two CONFIG.SYS files in the VIDVGA and VIDSVGA directories

from our machine classing example. This utility is included on the CD-ROM that accompanies this redbook.

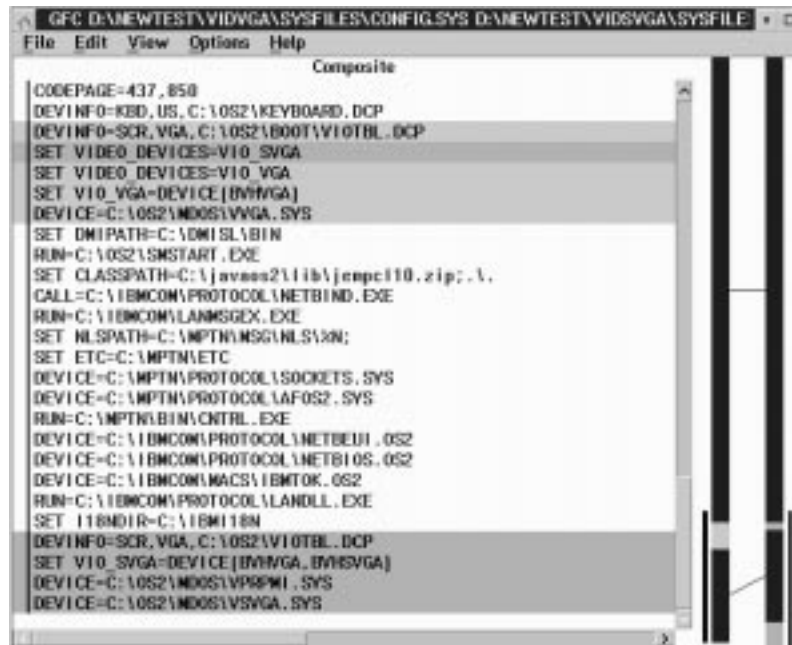


Figure 100. Comparing CONFIG.SYS Files

After you have identified the differences between the two CONFIG.SYS files, use a text editor to create a CONFIG.HW file containing only the statements that differ between the two files.

After you've identified the differences, create a CONFIG.HW file containing only the differing statements. Figure 101 shows the completed CONFIG.HW file for our machine classing example.

```
REM **** NCVIDEO BEGIN ****
DEVINFO=SCR,VGA,Z:\OS2\BOOT\VIOTBL.DCP
SET VIDEO_DEVICES=VIO_SVGA
SET VIO_SVGA=DEVICE (BVHVGA,BVHSVGA)
DEVICE=Z:\OS2\MDOS\VSVG.A.SYS
DEVICE=Z:\OS2\MDOS\VPRPMI.SYS
REM **** NCVIDEO END ****
```

Figure 101. CONFIG.HW

In the case of a video adapter, be sure to annotate the file's contents with the `NC Video Begin` and `NC Video End` comments as shown in Figure 101.

#### 10.4.9.2 Machine Class INF File

If you wish to use your machine class to create clients that will use the DHCP PXE boot mechanism, you must create a file named *machine\_class*.INF in the machine class directory. This file contains a list of the video driver files required to be downloaded during the first phase of the DHCP PXE boot process.

Figure 102 shows an example from the predefined IBM300GL machine class.

```
\BB20.US\OS2\BOOT\SCREEN01.SYS
\BB20.US\OS2\BOOT\VIOTBL.DCP
\BB20.US\OS2\DLL\BVSCALLS.DLL
\BB20.US\OS2\DLL\VIOCALLS.DLL
\BB20.US\OS2\DLL\BVHSVGA.DLL
\BB20.US\OS2\DLL\BVHVGA.DLL
```

Figure 102. Machine Class INF File

The files listed in the *machine\_class*.INF file are common for many types of video adapter. It is very likely that you can simply copy the *machine\_class*.INF file from an existing machine class into the machine class directory for your new machine class.

Note that you only require a *machine\_class*.INF file if you wish to use the DHCP PXE boot mechanism with your machine class. If you use only IEEE 802.2 RIPL clients, this file is not required.

#### 10.4.9.3 OS2.INI File

When adding support for certain types of hardware, particularly video adapters, you will need to make changes to a client's OS2.INI file. Under WorkSpace On-Demand 1.0, you needed to determine the differences between the OS2.INI files from your basic and enhanced configurations, and manually create a new OS2.INI file. With the introduction of the Machine Class Create utility under WorkSpace On-Demand 2.0, this process becomes somewhat easier since you can now use ASCII files known as resource compiler (RC) files, containing only the information needed to support the devices that you wish to support with your new machine class. These files are used by the Machine Class Create utility to build an OS2.INI file. This is similar to the MAKEINI utility, with which you may already be familiar, which uses a file named INI.RC to create or rebuild the OS2.INI file.

#### Note

Unlike the OS2.INI file, INI.RC files, including OS2.RCH, are flat ASCII text files and cannot contain binary values. The devices with which we have worked, including the examples used in this redbook, do not require binary data values within the OS2.INI file. However, we cannot guarantee that this will always be the case.

#### **Comparing OS2.INI Files**

In order to determine the information within the OS2.INI file, you must compare the two files from your basic and enhanced configurations. You can do this in a number of ways. For example, you can use an INI editor such as UNIMAIN or FM2 to examine the reference client's OS2.INI files and determine which application-types and their associated keys are required to support the hardware devices in question. After you have determined the application-types and keys that are required, you can use the same INI editor to create a new, or *delta*, INI file that contains only the information required to support the new devices.

However, we recommend a simpler alternative, using the INIDIFF.CMD utility included on the CD-ROM that accompanies this CD-ROM. This utility automatically compares two INI files and creates a third INI file that contains only the differences between the first two files. The syntax of the INIDIFF command is:

```
INIDIFF <old.INI> <new.INI> <delta.INI>
```

For example, when we used the INIDIFF.CMD utility to compare the two OS2.INI files in our machine classing example, we invoked the utility with the following command:

```
INIDIFF \VIDVGA\OS2.INI \VIDSVGA\OS2.INI DELTA.INI
```

The INIDIFF.CMD utility calls the COPYAPP.CMD utility, which is also included on the CD-ROM. You must place the COPYAPP.CMD utility in the same directory as the INIDIFF.CMD utility or in a directory referenced by the system's PATH statement.

Note that INIDIFF creates an INI file that includes *all* changes between the two OS2.INI files being compared. This means that the *delta*.INI file will often contain application-types that, while new or modified in the INI files being compared, are not directly related to the hardware devices you wish to support. In our experience, it is useful to edit the *delta*.INI file and check the

entries placed there by the INIDIFF.CMD utility. In some cases, you can remove unnecessary entries and reduce the size of the file.

In some instances, the entries included in the *delta*.INI file may contain binary values that cannot be incorporated into an RC file. In such cases, you cannot use the Machine Class Create utility to build the OS2.INI file for your new machine class, and you will therefore need to build the OS2.INI file manually.

### **Required Application-Types**

The application-types required within the *delta*.INI file will differ depending on the hardware devices that you are supporting with your new machine class. The most common type of hardware that modifies the OS2.INI file, however, is the video adapter, and video adapter support typically requires the following two application types:

- PM\_DISPLAYDRIVERS will always be required. If you see a keyname DEFAULTSYSTEMRESOLUTION, we strongly recommend that you delete it. This keyname is the result of configuring the reference client's screen for a specific resolution and/or display type. If you leave this key in the *delta*.INI file, you will be unable to select a specific video resolution when defining a client with your new machine class. If you delete this keyname, you must manually add the RESOLUTION\_CHANGED keyname with a data value of 1.
- A number of WIN\_RES\_... application-types are required to support multiple video resolutions under WIN-OS2. In most cases, you will want to keep these application-types in the DELTA.INI file.

### **Additional Application-Types**

Depending on the type of device you are supporting, you may require additional application-types and keynames. For example, the Trident 3-D 9397 chip set used in the IBM Thinkpad 770 requires the TRIDENT9680 and TRIDENT0S2 application-types in its OS2.INI file. You must carefully examine the reference client's OS2.INI file to determine which application-types and keynames are necessary for proper support. Unfortunately, there may be some trial and error in this process.

Figure 103 displays a portion of the *delta*.INI file required to support the CL-GD546X video graphics adapter in our example, as viewed using the UNIMAIN INI file editor.

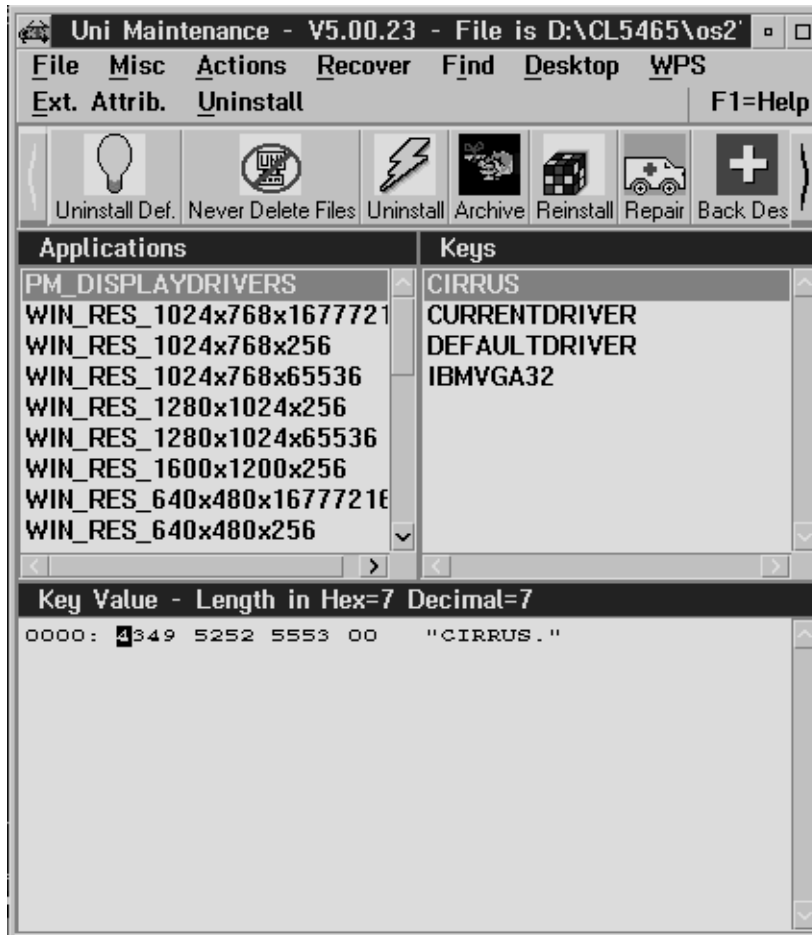


Figure 103. DELTA.INI File

### Creating the INI.RCH File

Once the *delta*.INI file is complete and correct, the next step is to use the MAKERC utility (supplied on the CD-ROM that accompanies this redbook) to convert the *delta*.INI file to an ASCII text file named INI.RC. The syntax of this command is:

```
MAKERC <delta.INI> INI.RC
```

**Note**

After creating the INI.RC file, use a text editor to ensure that each line conforms to the format:

"VALUE" "KEY" "DATA"

Our testing has shown that extraneous double quotation marks occasionally occur in INI files, especially in lengthy data descriptions, and MAKERC copies these into the INI.RC file, causing it to be corrupted. An example of such a corrupted statement is shown below:

```
"WIN_RES_CL_8" "3" "win.ini fonts "Courier 10,12,15 (8514/a res)"      courf.fon"
```

In this example, the double quotation marks immediately after the font resolutions and before the font file name must be removed. An example of a corrected statement is shown below:

```
"WIN_RES_CL_8" "3" "win.ini fonts Courier 10,12,15 (8514/a res)      courf.fon"
```

Figure 104 shows a correctly edited INI.RC file. Note that double quotation marks are used to delineate exactly three fields in each record.

```
STRINGTABLE REPLACEMODE
BEGIN
.....
"PM_DISPLAYDRIVERS" "IBMVG32" "IBMVG32"
"PM_DISPLAYDRIVERS" "CIRRUS" "CIRRUS"
"PM_DISPLAYDRIVERS" "CURRENTDRIVER" "CIRRUS"
"PM_DISPLAYDRIVERS" "DEFAULTDRIVER" "IBMVG32"
"WIN_RES_1024x768x256" "WIN_RES_SET" "WIN_RES_CL_7"
"WIN_RES_1280x1024x256" "WIN_RES_SET" "WIN_RES_CL_A"
.....
.....
"WIN_RES_CL_2" "1" "system.ini boot.description aspect 100,96,96
.....
"WIN_RES_CL_8" "3" "win.ini fonts Courier 10,12,15 (8514/a res)      courf.fon"
.....
END
```

Figure 104. INI.RCH File

You should ensure that the records in the file are preceded with the STRINGTABLE REPLACEMODE and BEGIN strings and that the last record is followed by the END string as shown in Figure 104.

When the file is correct and complete, you should rename it to INI.RCH. This is the file name that is expected by the Machine Class Create utility.



#### 10.4.9.4 OS2SYS.RCH File

Adding support for certain hardware devices makes changes to the OS2SYS.INI file in a similar way to that already discussed for the OS2.INI file. As with OS2.INI, the Machine Class Create utility allows you to create an RC file containing the necessary changes, which is used by the Machine Class Create utility to create the OS2SYS.INI file.

The steps necessary to create an OS2SYS.RCH file are similar to those already discussed for the OS2.RCH file. You can use the same utilities to compare two files and extract the differences, then convert the binary INI file to an ASCII RC file. You can then check this RC file with a text editor before renaming it to create the RCH file.

In our example, installing support for the CL GD-646X video chip set did not alter the OS2SYS.INI file. However, you should always check the two versions copied from the reference client to determine whether changes have been made.

#### 10.4.9.5 FIT Extension

Every Workspace On-Demand client uses a file index table (FIT) to redirect file I/O requests from the client to the correct locations on the server. FIT entries for the operating system and its associated hardware support are held in the machine FIT file, as described in Chapter 3, "Understanding File Index Tables" on page 85. Your new machine class must include a FIT extension that will update the machine FIT file with the entries necessary to redirect requests for the drivers and other configuration files that support your hardware devices.

The FIT extension has two major sections: a section for writable files and a section for read-only files.

- The *writable section* contains redirection requests for files that will be accessed in read/write mode. In many cases, you can use an existing FIT extension from another machine class to help identify these files. For example, we used the existing CL5436.FIT file as the basis for our new FIT extension CL5465.FIT. However, there will be instances when you do not have a similar FIT extension FIT to use as a template. In such cases, you will find tools such as FWatch to be extremely useful
- The *read-only* section contains redirection requests for file that will be accessed in read-only mode, such as device drivers. In our example, you will recall that we used the DSPINSTL.LOG file to create a command file that we then used to copy files from the reference client to the server. We took this command file and modified its contents to serve as the basis for the read-only section of the FIT extension.

Figure 105 shows the FIT extension file that we created for our machine classing example.

```
; video support for the Cirrus Logic GD546X Chip Set

; writable files

Z:\OS2\PRIVATE.*           \\RPLSERVER\WRKFILES\DEFAULT\OS2
Z:\OS2\SVGADATA.*         \\RPLSERVER\WRKFILES\DEFAULT\OS2
Z:\OS2\VIDEO.*            \\RPLSERVER\WRKFILES\DEFAULT\OS2

; read-only files

Z:\OS2\MDOS\WINOS2\SYSTEM\DCI54X.DRV      BB20.US\OS2\VIDEO\CL5465\DCI546X.DRV
Z:\OS2\MDOS\WINOS2\SYSTEM\546X.DRV       BB20.US\OS2\VIDEO\CL5465
Z:\OS2\MDOS\WINOS2\SYSTEM\CGA*.FON       BB20.US\OS2\VIDEO\CL5465
Z:\OS2\MDOS\WINOS2\SYSTEM\COURE.FON      BB20.US\OS2\VIDEO\CL5465\COURE.FON
Z:\OS2\MDOS\WINOS2\SYSTEM\EGA*.FON       BB20.US\OS2\VIDEO\CL5465
Z:\OS2\MDOS\WINOS2\SYSTEM\SERIFE.FON     BB20.US\OS2\VIDEO\CL5465\SERIFE.FON
Z:\OS2\MDOS\WINOS2\SYSTEM\SERIFE.FON     BB20.US\OS2\VIDEO\CL5465\SERIFE.FON
Z:\OS2\MDOS\WINOS2\SYSTEM\SSERIFE.FON    BB20.US\OS2\VIDEO\CL5465\SERIFE.FON
Z:\OS2\MDOS\WINOS2\SYSTEM\SYMBOLE.FON    BB20.US\OS2\VIDEO\CL5465\SYMBOLE.FON
Z:\OS2\DLL\BVHSVGA.DLL                  BB20.US\OS2\VIDEO\CL5465\BVHSVGA.DLL
Z:\OS2\DLL\IBMGPMI.DLL                  BB20.US\OS2\VIDEO\CL5465\IBMGPMI.DLL
Z:\OS2\DLL\VIDEO*.DLL                  BB20.US\OS2\VIDEO\CL5465
Z:\OS2\DLL\OEMPMI.DLL                  BB20.US\OS2\VIDEO\CL5465\OEMPMI.DLL
Z:\OS2\DLL\CIRRUS.DLL                  BB20.US\OS2\VIDEO\CL5465\CIRRUS.DLL
Z:\OS2\MDOS\VS VGA.SYS                  BB20.US\OS2\VIDEO\CL5465\VS VGA.SYS
Z:\OS2\MDOS\VPRMI.SYS                  BB20.US\OS2\VIDEO\CL5465\VPRMI.SYS
Z:\OS2\SCREEN01.SYS                    BB20.US\OS2\VIDEO\CL5465\SCREEN01.SYS
Z:\OS2\DLL\VCFGMRI.DLL                  BB20.US\OS2\VIDEO\CL5465\VCFGMRI.DLL
Z:\OS2\DLL\WPVIDSYS.DLL                 BB20.US\OS2\VIDEO\CL5465\WPVIDSYS.DLL
```

Figure 105. FIT Extension CL5465.FIT

Note that in this case, we have used wildcards to reduce the number of explicit mappings in the FIT file. Remember that since the in-memory copy of the FIT is restricted to 64 KB in size, wildcards can help to reduce the number of entries in the FIT. See Section 3.1.1, “Using Wildcard Characters in FIT Files” on page 86, for more information on how to use wildcards in FIT files.

#### 10.4.9.6 SYSTEM.INH

The SYSTEM.INH file is used by the Machine Class Create utility to create an updated SYSTEM.INI file for WINOS2 support. The steps necessary to create this file are similar to those used to create the CONFIG.HW file. You must compare the two SYSTEM.INI files from your basic and enhanced configurations, and create a SYSTEM.INH file containing only the differences. You can use a file comparison utility such as GFC to compare the two files; then use a text editor to create the SYSTEM.INH file.

Figure 106 on page 281 shows the SYSTEM.INH file that we created for the CL GD-646X video chip set in our machine classing example.

```
[boot]
display.driv=546xfs.drv
sdisplay.driv=546xsl.drv
fdisplay.driv=546xfs.drv
386grabber=avga.3gr
286grabber=vgacolor.2gr

[boot.description]
display.driv=CL-GD546x
fdisplay.driv=CL-GD546x
sdisplay.driv=CL-GD546x

[386Enh]
display=vdd546x.386

[CL_WinAccel]
fontcaching=0
linearaddr=U
signnmsg=off
changeres=off
fontsize=small
colordepth=8
resolution=640x480

[CLVGA]
logo=U
```

Figure 106. SYSTEM.INH

#### 10.4.9.7 WIN.INH

The WIN.INH file is used by the Machine Class Create utility to create an updated WIN.INI file for WINOS2 support. The steps necessary to create this file are similar to those used to create the CONFIG.HW and SYSTEM.INI files. You must compare the two WIN.INI files from your basic and enhanced configurations, and create a WIN.INH file containing only the differences. You can use a file comparison utility such as GFC to compare the two files, then use a text editor to create the WIN.INH file.

Figure 107 shows the WIN.INH file that we created for the CL GD-646X video chip set in our machine classing example.

```
[Desktop]
IconSpacing=100
```

Figure 107. WIN.INH

**Note**

You may find that there are no changes between the WIN.INI files from your basic and enhanced configurations. In such cases, you do **not** need to create a WIN.INH file.

### 10.4.10 Copy Files to New Subdirectories

You have now created or copied all the drivers and configuration files that you need to provide support for your hardware devices. Where necessary, you have also created new subdirectories to hold these files. You must now copy each file to the required subdirectory from which it can then be accessed by the Machine Class Create utility.

#### 10.4.10.1 General Hardware Device Support

In many cases, providing support for a new hardware device simply means adding a driver to the CONFIG.SYS file. You can copy the driver itself to the \BMLAN\RPL\BB20.US\OS2 directory, and refer to it in your FIT extension.

#### 10.4.10.2 Video Adapter Support

Providing support for video adapters is somewhat more complex due to the larger number of files involved. WorkSpace On-Demand 2.0 provides a standard directory structure for video adapter support that is utilized by the Machine Class Create utility to build machine class file structures.

There are two directories required to support a video adapter on a WorkSpace On-Demand server. These directories are described in Section 10.4.8, "Create New Subdirectories" on page 271. You must now copy the necessary files into each directory.

You should first copy the read-only device drivers. You can do this by modifying the command file that you used to copy these files from the reference client to the server so that it copies the same files from the temporary holding area to the correct subdirectory. In our example, we copied the read-only drivers to the \BMLAN\RPL\BB20.US\OS2\VIDEO\CL5465 directory.

Next, you should copy the hardware-specific, read/write configuration files from the temporary holding area to the correct directory on the server. In our example, we used the \BMLAN\RPL\MACHINES\BB20.US\VIDEO\CL5465 directory that we created in Section 10.4.8, "Create New Subdirectories" on page 271, and replaced the existing template files with the corresponding files from the temporary holding area.

The resulting directories and their contents from our machine classing example are shown in Figure 108.

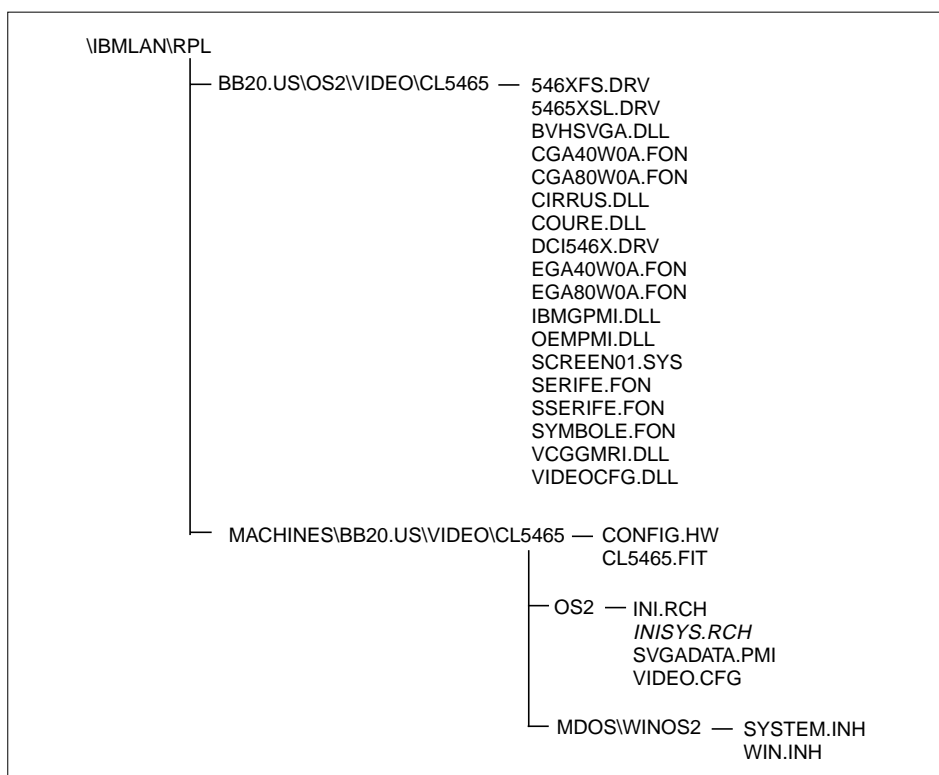


Figure 108. Video Subdirectories and Files for the CL-GD546X Chip Set

Note that INISYS.RCH is not required for our machine classing example, but is shown in Figure 108 for completeness.

#### 10.4.11 Modify the MCLASS.RSP File

At this point, all of your drivers and support files are in the correct positions to enable the Machine Class Create utility to build your machine class file structure. However, you must make the Machine Class Create utility aware of your new files and directories. You do this by editing the MCLASS.RSP file that the Machine Class Create utility uses to find the hardware devices and their supporting files.

MCLASS.RSP is an ASCII file that you can edit using a normal text editor. For our machine classing example, we needed to add a subfeature to the MC\_VIDEO\_TYPE structure, as shown in Figure 109.

```

MC_VIDEO_TYPE=(
    SubfeatureID=MC_VIDEO_TYPE_01
    SubfeatureID=MC_VIDEO_TYPE_02
    SubfeatureID=MC_VIDEO_TYPE_03
    SubfeatureID=MC_VIDEO_TYPE_04
    SubfeatureID=MC_VIDEO_TYPE_05
    SubfeatureID=MC_VIDEO_TYPE_06
    SubfeatureID=MC_VIDEO_TYPE_07
    SubfeatureID=MC_VIDEO_TYPE_08
    ObjectTitle[0]=1_VIDEO_TYPE
    Mode=User
    PackageTitle=MC_VIDEO_TYPE Install

```

Figure 109. Adding a New Video\_Type to MCLASS.RSP

In our example, we added the highlighted MC\_VIDEO\_TYPE\_08 record to the seven subfeatures already present in the MC\_VIDEO\_TYPE structure in the default MCLASS.RSP file.

We then copied the existing video type structure MC\_VIDEO\_TYPE\_01 to create the new MC\_VIDEO\_TYPE\_08 structure. Figure 110 shows the edited MC\_VIDEO\_TYPE\_08 structure.

```

MC_VIDEO_TYPE_08=(
    ObjectTitle[0]=Cirrus Logic 5465 (2M RAM)
    Mode=User
    PackageTitle=MC_VIDEO_TYPE Install
    Variable=(
        Name=VideoDir
        Description=Name of directory under VIDEO
        Value=CL5465

```

Figure 110. Editing the New Video\_Type in MCLASS.RSP

The steps required to modify the video type structure are:

- Change the MC\_VIDEO\_TYPE from 01 to 08.
- Change the ObjectTitle [0] field to Cirrus Logic 5465 (2M RAM).
- Change the VALUE field within the VARIABLE section to VALUE=CL5465.

You have now completed all the steps necessary to integrate your new hardware device into the Machine Class Create utility.

---

## 10.5 Creating the New Machine Class

You can use the Machine Class Create utility to create a new machine class that incorporates support for your new hardware device. Begin by opening the **BB20.US Machine Classes** folder, which resides in the **Machine Class Operating Systems** folder, as shown in Figure 111.

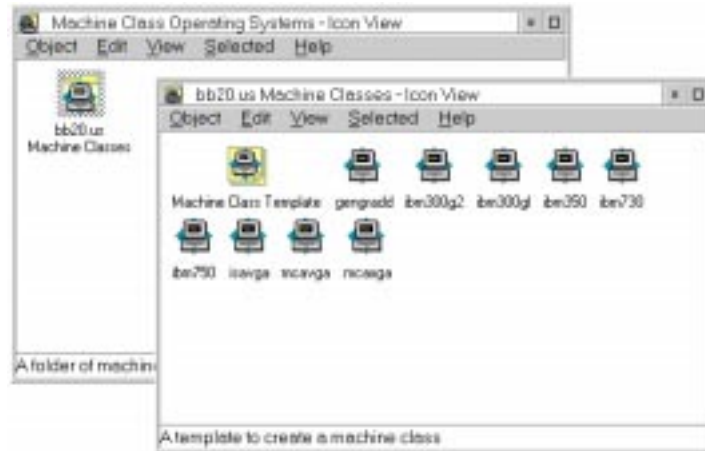


Figure 111. Creating a New Machine Class

Drag the **Machine Class Template** object to a blank area in the **BB20.US Machine Classes** folder, and drop the object. The OS/2 Warp Server GUI will open the Machine Class - Create notebook, which is described in detail in the following sections.

### 10.5.1 Identify the New Machine Class

When the Machine Class - Create notebook first opens, it displays the Identity page as shown in Figure 112.

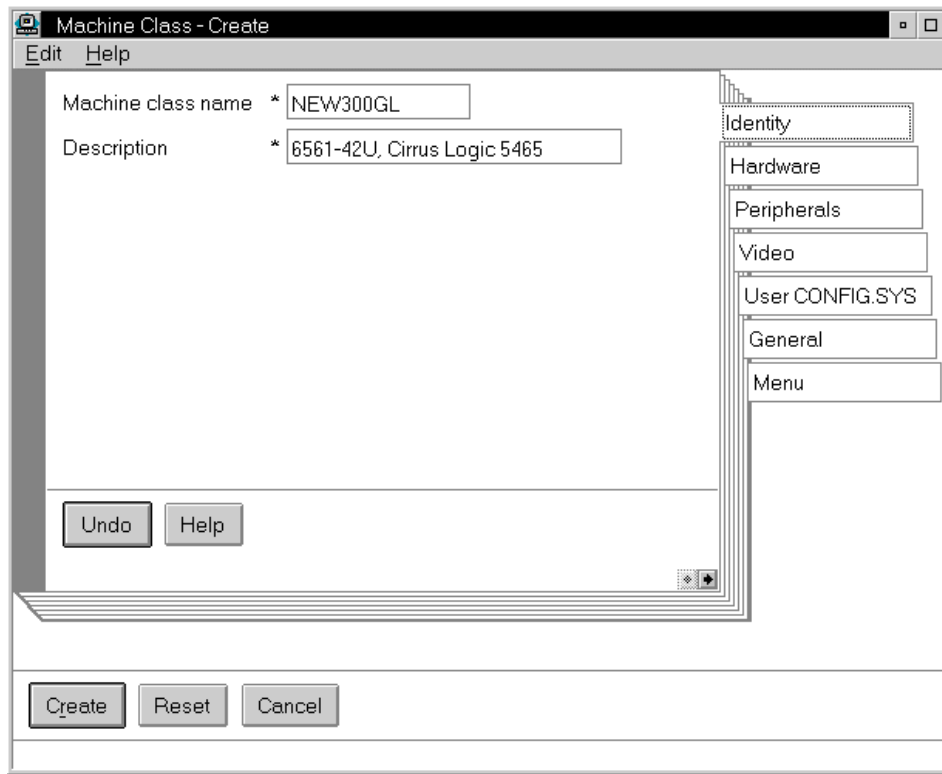


Figure 112. Machine Class Create Notebook - Identity Page

Enter a name for your new machine class in the Machine class name field and an appropriate description in the Description field. In our example, we created a machine class named NEW300GL to support the IBM Personal Computer 6561-42U, which includes the Cirrus Logic GD-546X chip set on the motherboard.



## 10.5.2 Configure Hardware Support

The next step is to specify the hardware configuration for the new machine class, using the Hardware page in the Machine Class - Create notebook as shown in Figure 113 on page 287.

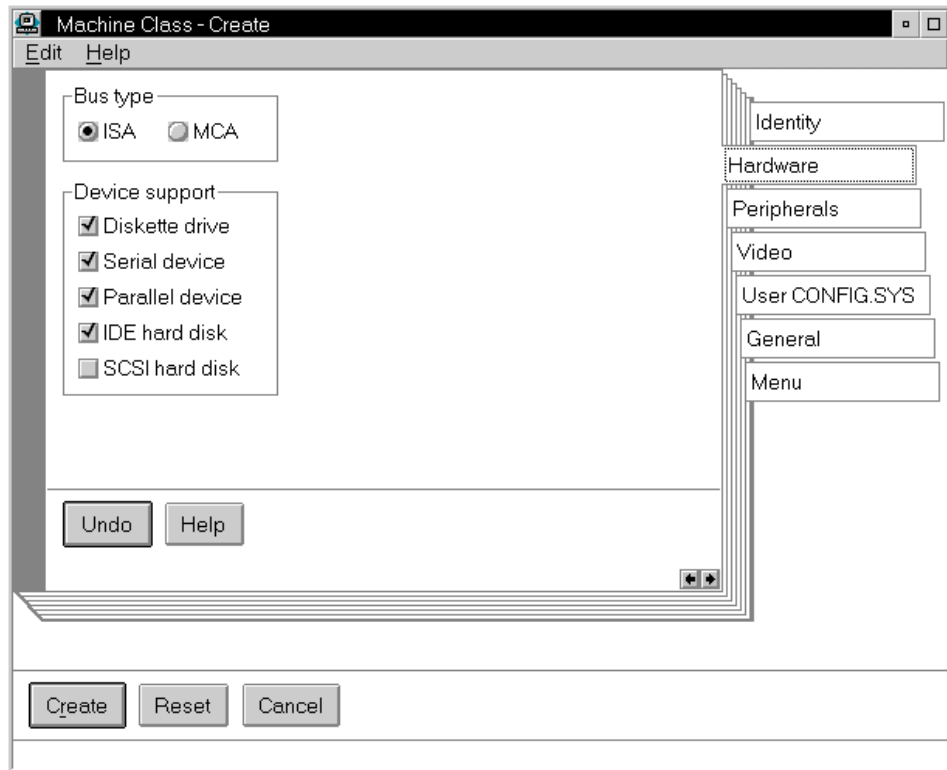


Figure 113. Machine Class Create Notebook - Hardware Page

Select the correct bus type for your machine class from the **Bus type** group of radio buttons. For our example, we selected the **ISA** radio button for the IBM PC 300GL Model 6561-42U.

Select the other types of devices for which you require support in your new machine class by selecting the appropriate check boxes in the **Device support** group. In our example, we accepted the default selections.

### 10.5.3 Configure the Peripheral Components

The next step is to configure the types of peripheral devices to be supported by your new machine class, using the Peripherals page in the Machine Class - Create notebook, as shown in Figure 114 on page 288.

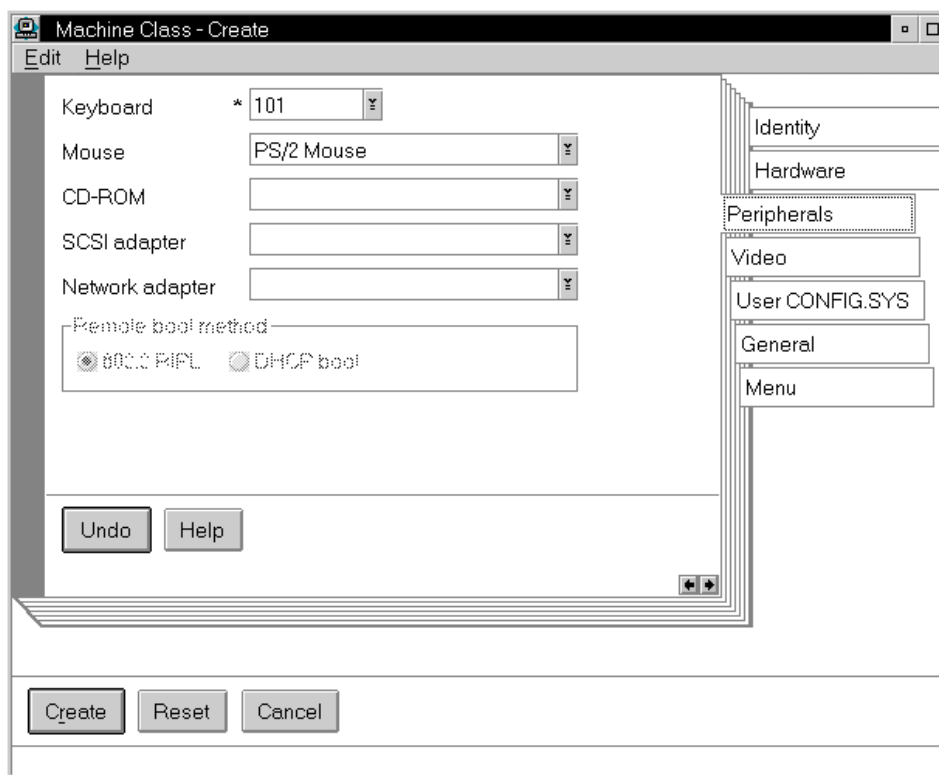


Figure 114. Machine Class Create Notebook - Peripherals Page

In our example, we have chosen a 101-key keyboard from the Keyboard drop-down list and a PS/2-style mouse from the Mouse drop-down list. We do not require support in the client for the CD-ROM drive (since we will want the drive to be disabled), nor do we require support for SCSI devices.

Note that you can specify a network adapter within the machine class by selecting an adapter from the Network adapter drop-down list. However, we recommend that you *do not* select a network adapter at this point. You can specify the network adapter for each client when you define the client, independently of the machine class definition. This provides additional flexibility for your client definitions.

## 10.5.4 Configure Video Support

The next step is to configure video support for your new machine class using the Video page in the Machine Class - Create notebook, as shown in Figure 115 on page 289.

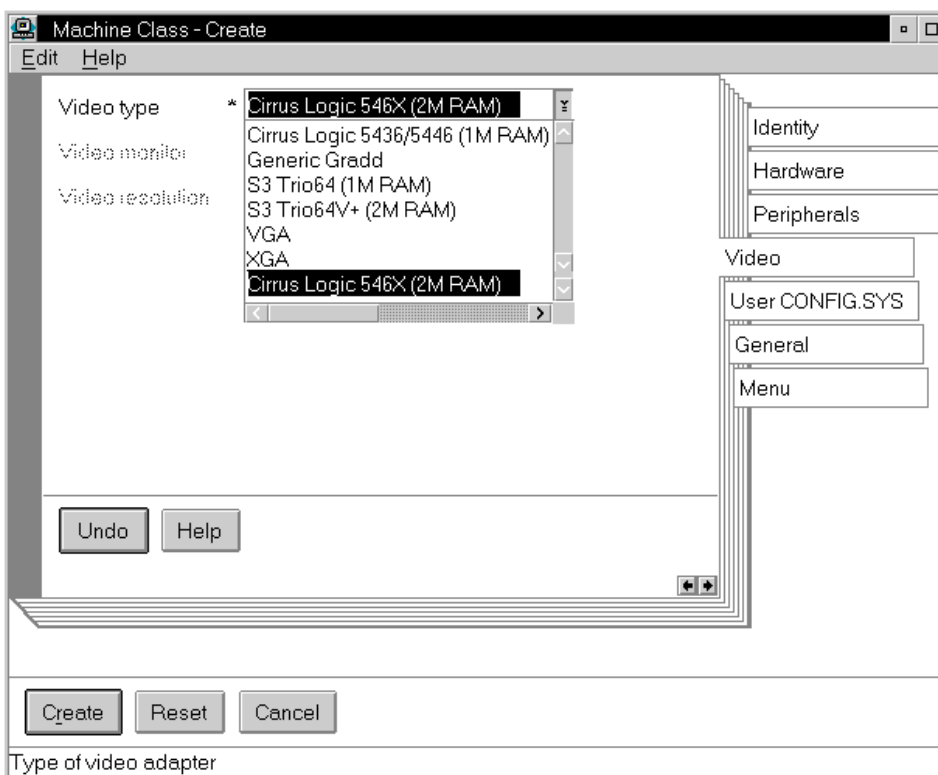


Figure 115. Machine Class Create Notebook - Video Page

Select the video adapter that you require from the **Video type** drop-down list. In our example, this list has been expanded from the default six entries since we have added support for the Cirrus Logic 546X video chip set. The Machine Class - Create notebook uses the MCLASS.RSP file, which we edited in Section 10.4.11, "Modify the MCLASS.RSP File" on page 283, to populate the Video type drop-down list.

Note that the Video Resolution drop-down list is "grayed out" in our example. This is because we have elected not to configure a specific video monitor at this point. Instead, we will specify the monitor type, resolution and color depth when we build individual clients using this machine class. However, it is

possible to preconfigure clients with this level of detail by building it into the machine class itself rather than specifying it on a client-by-client basis.

### 10.5.5 Configure Individual CONFIG.SYS Entries

For hardware devices that simply require one or more entries in the client's CONFIG.SYS file, you can add these entries directly using the User CONFIG.SYS page in the Machine Class - Create notebook, as shown in Figure 116 on page 290.

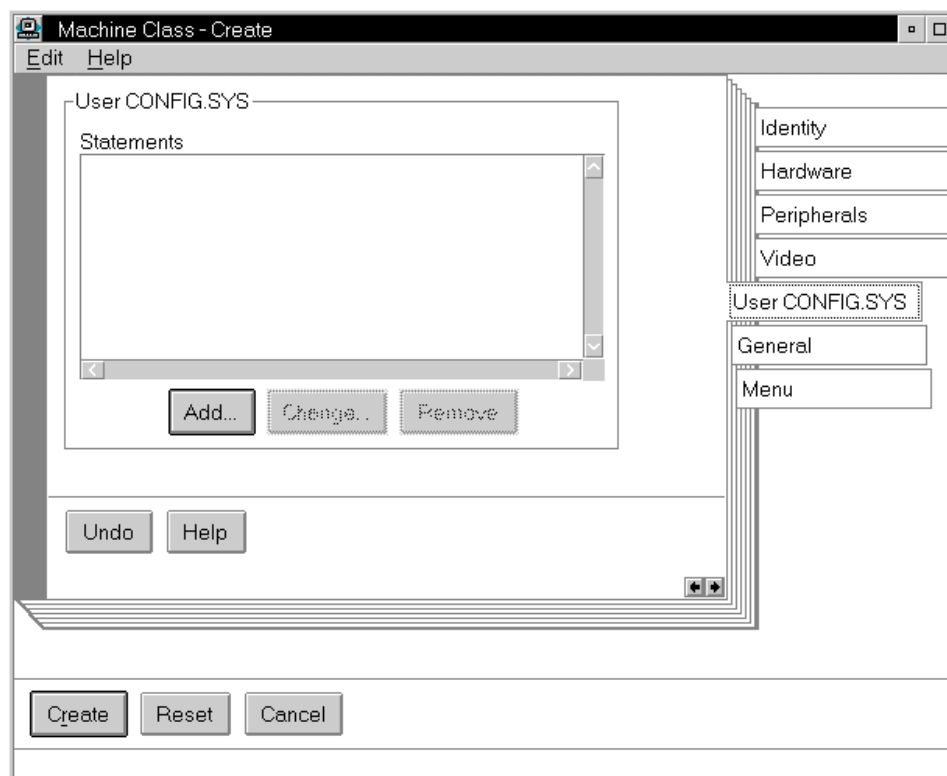


Figure 116. Machine Class Create Notebook - User CONFIG.SYS Page

In our example, we do not need to add any CONFIG.SYS statements on this page since we have already created a CONFIG.HW file to add the required video device drivers to all clients built using this machine class.

## 10.5.6 Create the New Machine Class

You are now ready to create the machine class file structure by selecting the **Set** button at the bottom left-hand corner of the Machine Class - Create notebook.

The Machine Class Create utility uses the definitions you specified in the Machine Class - Create notebook to build the new machine class file structure. After it has completed this activity, your new machine class will appear as an object in the BB20.US Machine Classes folder.

Figure 117 shows the standard machine class file structure created whenever you use the Machine Class Create utility to create a new machine class. As already mentioned, it is imperative that you follow this structure if you create a new machine class or modify an existing machine class without using the Machine Class Create utility.

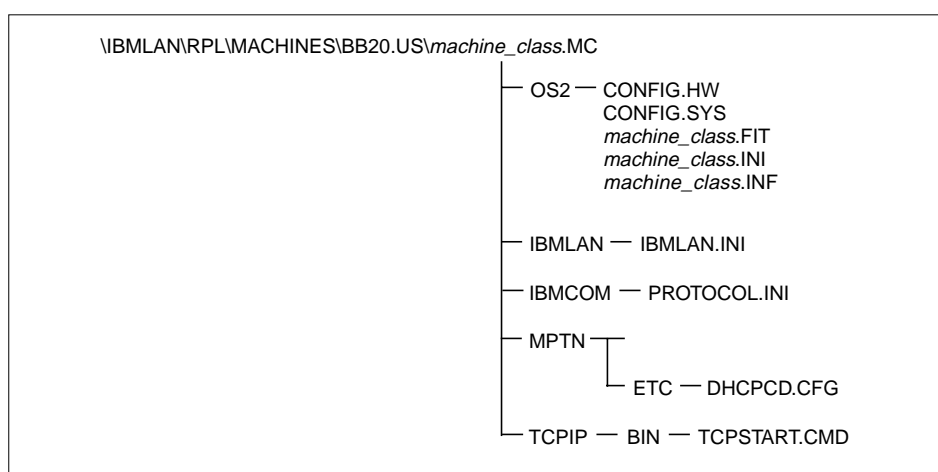


Figure 117. Machine Class File Structure

Now that you have created your new machine class, you must test it by defining a client using the new machine class.

## 10.5.7 Test the New Machine Class

You must now test your machine class by creating a new client definition for your reference client using the new machine class. See Section 6.1, “Defining a Client Workstation” on page 137, for more information on defining a client.

After you have defined your client, you should boot the client workstation using the new client definition, and ensure that all hardware devices function correctly using the support provided by the new machine class.

### 10.5.8 Points to Consider

If you now check the read-only portion of the BB20.US RPL tree, you'll see that there is a new machine class entry for NEW300GL as well as a new client entry for the new client, NEW300.

In the read/write portion of the BB20.US RPL tree, you see a new subdirectory for the new remote IPL requester as well. In it, among other files, are the new machine's OS2.INI, VIDEO.CFG and SVGADATA.PMI files. Based on the interrelationships of these files, as discussed in Section 10.4.10, "Copy Files to New Subdirectories" on page 282, you can see that these files must be open to update. This is because selecting a specific monitor type, resolution, color depth, or vertical refresh rate results in an updated version of the SVGADATA.PMI, VIDEO.CFG and OS2.INI files.

---

## 10.6 Common Problems

There are a number of problems that you may occur when creating and testing a new machine class. Some of the most common errors, and some ways to overcome them, are described in the following sections.

### 10.6.1 Corrupted NETGUI.INI

During the creation of this redbook, we found that it was possible to corrupt the NETGUI.INI file that controls the OS/2 Warp Server GUI. A corrupted NETGUI.INI file usually manifests itself by displaying an empty Machine Class Operating Systems folder along with a message to the effect that the INI file was corrupted. Other symptoms may include the inability to create a new machine class accompanied by a corrupted INI message.

This problem is likely to occur if you delete client-specific files and subdirectories without terminating the network sessions of the clients that they support or if you leave supporting files open and attempt to delete a client or machine class that uses those files. If you encounter this problem, you can correct it by following these steps:

In most cases, you can remedy this situation by following these steps:

1. Close the OS/2 Warp Server GUI.
2. Change directories to \IBMLAN\NETPROG.
3. Erase NETUGUI.PDB.

4. Copy NETGUI.BAK to NETGUI.INI.
5. Restart the OS/2 Warp Server GUI.

### 10.6.2 Unable to Create a New Remote IPL Requester

If you delete a client definition and then attempt to define another client with the same name, it is possible that you will receive a message to the effect that a client already exists with that name. This problem typically occurs when you manually delete the client-specific files and directories that support a client, before deleting the client definition from the Remote IPL Requesters folder.

This problem can usually be resolved by checking the **User Accounts** folder for your OS/2 Warp Server domain. If a user ID exists for the client, delete the user ID from the User Accounts folder, and then attempt to define the client again using the Client Definition notebook.

### 10.6.3 No Entries in the Machine Class Dropdown List

When you test your machine class by defining a client, you may find that the Machine Class drop-down list on the Hardware page in the Client Definition notebook is blank. This problem typically occurs when you create a machine class manually by copying an existing machine class file structure and modifying files, rather than using the Machine Class Create utility.

The problem is the result of an error in a machine class INI file. Each machine class has its own INI file that resides within the machine class file structure. If you copy a machine class and do not change the name or contents of the INI file, the resulting problem causes no machine classes to appear in the Machine Class drop-down list when you define a client.

Check your machine class INI files and ensure that their names match the machine class to which they belong and that the NAME field within each file also matches the name of the machine class.

### 10.6.4 No Entries in the Video Monitor Drop-Down List

When you test your machine class by defining a client, you may find that the **Video Monitor** and/or **Video Resolution** drop-down lists on the Hardware page in the Client Definition notebook are blank. This problem typically occurs when you are defining a machine class to support a new video adapter.

The problem usually stems from an error in the SVGADATA.PMI or VIDEO.CFG files. For example, you may not have copied the correct files to

the hardware-specific video subdirectories on your server, or you may have copied the files to the wrong location.

Ensure that the SVGADATA.PMI and VIDEO.CFG files reside in the hardware-specific video subdirectory on your server, as described in Section 10.4.10, “Copy Files to New Subdirectories” on page 282.

### **10.6.5 Exception in Device Driver SINGLEQ\$**

When booting a client using your new machine class definition, you may receive a message indicating an error in the SNGLQUE\$ driver, and the client will fail to boot. This problem typically occurs when you are defining a machine class to support a new video adapter.

The problem usually stems from an incorrect driver or support file being loaded at boot time. If your video adapter requires a number of support files and the client loads one or more files that do not match the remaining files, the boot process will fail.

The most common cause of this problem is an incorrect redirection statement in the client's machine FIT file. You should ensure that the FIT extension for the machine class redirects all file I/O requests for video-related files to the correct locations on the server.

### **10.6.6 Black Screen Instead of a Client Desktop**

When booting a client using your new machine class definition, you may see the client load its device drivers, then fail when it attempts to switch video modes in order to display the desktop, showing only a blank screen with a blinking cursor in the top left-hand corner.

This problem is similar to that described in Section 10.6.5, “Exception in Device Driver SINGLEQ\$” on page 294, and typically stems from an incorrect video driver being loaded at boot time. This, in turn, may be caused by the driver itself being incorrect or by the client's machine FIT file redirecting an I/O request to the wrong location on the server.

You should ensure that you have copied the correct files to the read-only video subdirectory when creating your machine class, and that the FIT extension for your machine class redirects the client's I/O requests to that subdirectory.



## 10.7 Querying Machine Class Information

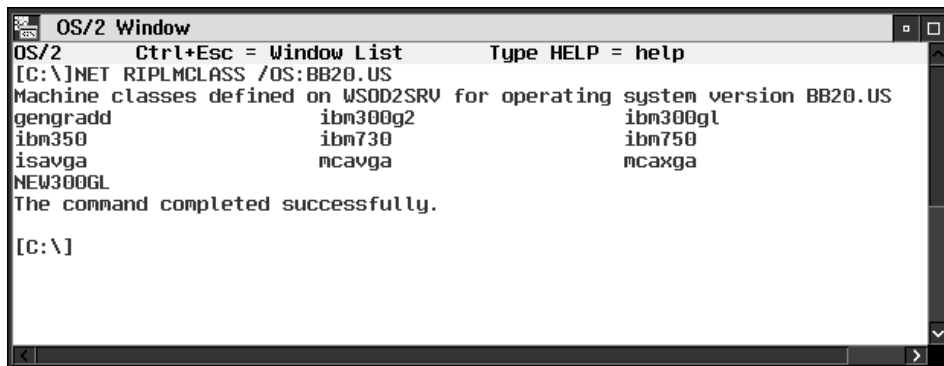
In an enterprise environment, it may be useful to check the WorkSpace On-Demand Manager 2.0 servers from time to time in order to determine the machine classes defined at each server. This may become necessary to ensure that your server's definitions are still consistent with the actual supported hardware. If any machine class that is defined at one location is not defined on another server or, even worse, exists on multiple servers but differs in content, you can use the `NET RIPLMCLAS` command to gather the relevant information that you will need to document and correct the situation.

The `NET RIPLMCLAS` command provides both a summary of the machine classes supported by a specific WorkSpace On-Demand Manager 2.0 server and a detailed view of the configuration supported by a specific machine class. For the correct syntax of the command, see the WorkSpace On-Demand 2.0 Administrator's Guide that is packaged as part of the online documentation.

To obtain a list of the machine classes installed on your server, use the following command:

```
NET RIPLMCLAS /OS:BB20.US
```

This command will display a list of installed machine classes on the server from which it was run as shown in Figure 118.



```
OS/2 Window
OS/2 Ctrl+Esc = Window List Type HELP = help
[C:\]NET RIPLMCLASS /OS:BB20.US
Machine classes defined on WSOD2SRV for operating system version BB20.US
genradd          ibm300g2          ibm300gl
ibm350           ibm730            ibm750
isavga          ncavga            ncaxga
NEW300GL
The command completed successfully.

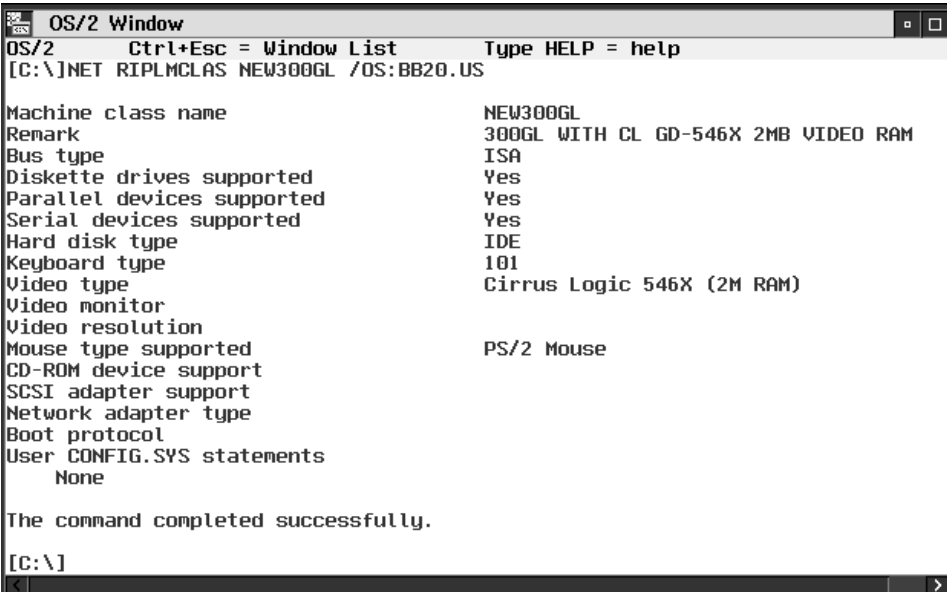
[C:\]
```

Figure 118. `NET RIPLMCLAS /OS:BB20.US`

If you need to obtain the same information for another server in your network, you must specify the `SE` option specifying the server's name as shown in the following example:

```
NET RIPLMCLAS /OS:BB20.US /SE:WSOD2SRVR
```

If you specify the machine class and the operating system, the command's output displays detailed information about the machine class as shown in Figure 119.



```
OS/2 Window
OS/2      Ctrl+Esc = Window List      Type HELP = help
[C:\]NET RPLMCLAS NEW300GL /OS:BB20.US

Machine class name          NEW300GL
Remark                      300GL WITH CL GD-546X 2MB VIDEO RAM
Bus type                    ISA
Diskette drives supported   Yes
Parallel devices supported  Yes
Serial devices supported    Yes
Hard disk type              IDE
Keyboard type               101
Video type                  Cirrus Logic 546X (2M RAM)
Video monitor
Video resolution
Mouse type supported        PS/2 Mouse
CD-ROM device support
SCSI adapter support
Network adapter type
Boot protocol
User CONFIG.SYS statements
    None

The command completed successfully.

[C:\]
```

Figure 119. NET RPLMCLAS NEW300GL /OS:BB20.US

As with other commands, you can redirect the output to an ASCII file in order to collect and integrate information concerning the machine classes in your environment. This type of a procedure insures that all machine class definitions are correct and consistent.

Note, however, that you cannot use the `RPLMCLAS` command to modify a machine class. If you need to modify your machine classes remotely, use the Machine Class Create utility to modify the machine class at a central site, and then deliver the corrected files to remote locations either across the corporate network or on removable media such as CD-ROM or diskette.

---

## Chapter 11. Deploying WorkSpace On-Demand

This chapter discusses the issues and requirements involved in the deployment of WorkSpace On-Demand in a large enterprise network. The stages that should be carried out prior to a deployment to ensure a successful rollout are discussed. The final part of the chapter discusses ways to manage a large-scale deployment of WSoD servers and provides examples of some automation techniques.

---

### 11.1 Deployment Planning

Proper planning, analysis and testing *prior* to a WorkSpace On-Demand deployment are all critical factors that contribute to the overall success of a rollout. If these phases are skipped or not given adequate time and attention, the success of the rollout will be jeopardized.

#### 11.1.1 Points to Consider

Planning is one of the most crucial parts of any deployment. Good planning helps avoid unexpected problems and minimizes cost, time and effort. A number of suggestions and techniques are given here to aid in the deployment planning phase. Review Chapter 4, "WorkSpace On-Demand Planning" on page 103, for an in-depth description of hardware, software and network requirements.

##### 11.1.1.1 Gather Information

Many factors must be taken into consideration during the planning phase of a WorkSpace On-Demand deployment. Some questions which should be addressed during the planning phase are listed below. This list is intended to stimulate the planning thought process. Not all questions may be relevant to your particular situation.

##### ***Brainstorming Questions***

- Will you install entirely new servers and clients, or will you incorporate WorkSpace On-Demand into an existing install base?
- What is the current hardware configuration of the target install base?
  - How many types of servers are present, and what are the hardware configurations of each one?
  - Do all of the existing servers have adequate processors, memory and hard disk space, or are upgrades necessary?

- How many types of clients comprise the existing install base and what are the hardware configurations of each one?
- Do all of the existing clients have adequate processors and memory, or are upgrades necessary?
- Will you need to develop new machine classes for any of the existing clients?
- Can you standardize the server and client hardware?
- Will the same hardware be available throughout the entire rollout?
- Will "diskless" clients be used?
- What video resolution is required?
- What peripheral devices (printers, scanners, and so on) are attached to the clients?
- What types of network adapters are installed in the clients, and are they supported by WorkSpace On-Demand?
- Are any upgrades (such as Boot PROMs) required for the network adapters?
- Do any of your network adapters require flash updates to their firmware?
- Will *Wake On LAN* network adapters be used?
- What is the current software configuration of the target install base?
  - Are all of the existing servers at the prerequisite software levels (OS/2 Warp Server, OS/2 RIPL support, FixPaks and so on), or are upgrades necessary?
  - Do you need to support both boot mechanisms?
  - If you require DHCP PXE support, can you use existing DHCP servers?
  - Can you add the PXEPROXY service to your existing DHCP servers?
  - Is it feasible to run all the necessary DHCP services (DHCP, PXEPROXY, BINL and TFTP) on the WorkSpace On-Demand Servers?
  - Do your clients' network adapters provide the necessary support for the DHCP PXE boot mechanism?
  - What applications and versions are used in the existing install base?
  - What applications and versions will be used in the WorkSpace On-Demand deployment?

- What are the prerequisites and configuration issues for each application?
- What type of sessions do the applications require (PM, VDM, Java and so on)?
- What future applications may be required in the deployment, and what are the anticipated prerequisites?
- What is the network infrastructure of the target install base?
  - How many clients are attached to each server in the existing sites?
  - What is the maximum number of clients attached to a single server in the existing sites?
  - What network protocols are used in the existing install base?
  - What network protocols must be supported in the WorkSpace On-Demand deployment?
  - What is the network topology and bandwidth?
  - Are the networks divided into segments, and what segmentation devices (bridges, routers, remote bridges) are used?
  - Are any changes or upgrades to the existing network infrastructure required?

#### **11.1.1.2 Determine Scope and Design Criteria**

After gathering all of the pertinent information, the next phase is to determine the design requirements for the WorkSpace On-Demand deployment. Factors which should be considered include performance, security, availability, and usability features. The following questions should help determine the scope and design criteria for the deployment.

#### ***Brainstorming Questions***

- What are the expected performance characteristics of the WorkSpace On-Demand deployment?
  - What are the performance benchmarks (timings) for the existing install base?
  - What are acceptable benchmarks for a WorkSpace On-Demand install base?
  - What is the maximum load (number of clients) that will be placed on a single server?
  - What is the maximum acceptable boot time in a "boot storm"?
- What features need to be built into the system?

- What applications and versions will be used in the deployment?
- What programs must be autostarted (such as IBM AntiVirus, Netfinity Network Interface) on the client?
- Will users "roam" between clients, or will they be assigned to specific clients?
- How should icons and folders be arranged on the client desktop?
- Are multiple desktop configurations required for different users?
- Is a customized logon program (using the PMLOGON user exits) required?
- What level of security is required?
- Which files in the client image should have write access and which should only have read-only access?
- Is access to the local drive of the client required (for local swapping)?
- What are the availability requirements?
- Is a local bootable partition on the client required if the WSoD server goes off-line?
- What backup or failover mechanisms need to be in place?
- Which files (such as RPL.MAP) on each server should be backed up and restored if a server replacement is necessary?
- What processes need to be designed or automated?
  - Will local administrators be assigned to any of the WorkSpace On-Demand sites?
  - Is an unattended installation process of WorkSpace On-Demand necessary?
  - How will client definitions be created on each server?
  - How automated does the client definition process need to be?
  - What configuration information (LAA, LU addresses and so on) is needed to create each client?
  - How will client definitions be deleted on each server?
  - How will applications be defined on each server?
  - How will applications be assigned to users?
  - What processes are in place for updates or change control?
  - How often will updates be necessary?

- What days and times will client workstations be active and powered on?

### **11.1.2 Analyze Deployment Techniques**

The deployment techniques and implementations of other WorkSpace On-Demand customers may provide suggestions of things to consider. We recommend that you read the redbook entitled *WorkSpace On-Demand Early Customer Experiences* (SG26-5107) for an in-depth look at the experiences of a number of enterprises that have deployed WorkSpace On-Demand. For your convenience, we have summarized some commonly used deployment techniques in this section.

#### **11.1.2.1 Standardize Client Hardware**

A large number of client hardware configurations adds complexity to a WorkSpace On-Demand deployment since each configuration will require a separate machine class to be developed, tested and supported. Make an effort to standardize on one or as few as possible client hardware configurations. Over the course of time, this will save considerable time and effort.

#### **11.1.2.2 Categorize Sites for Easier Deployment**

When deploying WorkSpace On-Demand into a geographically dispersed environment, the task of generating server configurations can be simplified by classifying each location into a set of categories and assigning a standard network and server configuration to each category.

For example, all locations could be classified into the following three categories:

- Small configuration - less than  $x$  clients
- Medium configuration - between  $x$  and  $y$  clients
- Large configuration - between  $y$  and  $z$  clients

After conducting performance testing for each category, a standard network configuration and a standard set of server tuning parameters can be used to provide the optimum performance for each category. This can significantly reduce the amount of work required to deploy WorkSpace On-Demand since a standard server image can be created for each category.

#### **11.1.2.3 Define Network Applications Prior to Deployment**

If an OS/2 Warp Server domain has already been deployed and you intend to add WorkSpace On-Demand clients, the deployment of WorkSpace

On-Demand can be accelerated by installing application code on the existing OS/2 Warp Servers and defining these applications as network applications in the existing domain prior to deploying WorkSpace On-Demand.

In this way, the network applications can be installed and tested before the new boot servers and client operating systems are deployed. This phased approach reduces the logistical complexity of the deployment and lessens the likelihood of disruption to existing LAN services.

If applications are set up as network applications on the existing OS/2 Warp Server servers, they can be accessed by users on existing "fat" clients. When WSoD servers and clients are deployed, users on the WSoD clients can log on to the existing domain and will immediately have access to their network applications.

#### **11.1.2.4 Segment the Network**

When designing the physical network topology, it makes sense to divide the local area network into multiple LAN segments. This segmentation improves network stability and redundancy while offering additional network management opportunities not available in passive single-segment environments. In high-traffic scenarios, segmenting the network can help to improve network performance by spreading the workload across several different physical network segments.

#### **11.1.2.5 Configure WSOD Servers as Additional Servers**

When deploying a WorkSpace On-Demand server in an existing OS/2 Warp Server domain, you can minimize disruption to the existing LAN services and avoid modifying existing server configurations by using a separate machine as a WSOD server and configuring this machine as an additional server in the existing domain.

Users on WSoD clients that boot from this server will, by default, log on to the existing domain. Any resources and network applications already defined for those users will be available to them on the WorkSpace On-Demand clients.

#### **11.1.2.6 Automate the Collection of Configuration Data**

It is necessary to supply a network adapter address when defining a client workstation to the WorkSpace On-Demand server. In networks with existing clients already deployed, it is necessary to retrieve the network adapter address for each particular machine. While doing this, it is good practice to collect as much additional data about the hardware as possible.



### 11.1.2.7 Use Redundant Servers to Improve Reliability

In a multi-server environment, using more than one WorkSpace On-Demand server improves the redundancy of the network. In the event that the primary WSoD server fails, other servers can automatically assume the server's workload. This is known as *mutual failover*. The only likely disruption is that users will need to reboot their client workstations.

WorkSpace On-Demand is designed in such a way that any WSOD server can reply to a RIPL request from a client workstation, provided that client is defined to the server. When working in a mutual failover environment, it is therefore sensible to duplicate the RPL.MAP file, which defines the clients that can boot from a particular server, to all WorkSpace On-Demand servers in the domain.

In situations where high availability is a critical requirement, consider using a large number of small servers with a reduced number of clients per server, rather than a small number of larger servers. This multi-server technique means that a server failure affects only a minimal number of clients, and the majority of end users will be able to carry out their normal work. However, remember that other single points of failure, such as file and database servers or communications gateways, must also be provided with appropriate levels of redundancy.

In environments where high availability is a significant requirement, many organizations implement a backup domain controller to provide logon verification in the event that the primary domain controller fails. Because the task of logon verification is not particularly resource intensive, the backup domain controller function can often be combined with other functions such as those of a file server or communications gateway.

Similarly, you may wish to implement a backup WorkSpace On-Demand server to service boot requests in the event that the primary WorkSpace On-Demand server is unavailable. Such a configuration has two benefits:

- It provides a backup in the event of a boot server failure, thereby eliminating a single point of failure in the network.
- It provides a second server to handle boot requests during periods of heavy workload, when the primary boot server may be slow to respond.

You can combine the backup WorkSpace On-Demand server function with that of another machine, such as a file server or communications gateway, in a similar manner to that already mentioned for a backup domain controller. Since the WorkSpace On-Demand server is most commonly required during periods when most clients are booting, there is likely to be little demand for

the machine's primary file serving or communications function at that time, and thus the two functions should not unduly affect one another.

Implementing a backup WorkSpace On-Demand server is relatively simple:

- You must provide an OS/2 Warp Server system with the server software prerequisites described in Section 4.2, "Software Prerequisites" on page 109.
- You must create identical client workstation definitions on the backup WorkSpace On-Demand server.

To ensure the fullest level of function on your backup WorkSpace On-Demand server, you should ensure that the backup domain controller (if any) contains copies of files such as the user FIT file (see Chapter 3, "Understanding File Index Tables" on page 85, for more discussion of FITs).

#### **11.1.2.8 Modify the Client's Boot Drive**

When installing applications on a reference client, it is typical to install the application's files to the client's boot drive, usually drive C. Indeed, some applications produce configuration files that are hard-coded to include drive C, and as such, the applications do not work correctly when installed on a different drive. This can create problems when integrating applications into the WSoD client image on the server.

A simple but effective solution is to define the client's logical boot drive on the server as drive C rather than the default drive Z. The entries in the machine and user FIT files are adjusted accordingly. This approach has a number of benefits:

- It simplifies the process of transferring environment settings from a reference client to the client's boot image since, by default, locally installed applications refer to drive C.
- The CONFIG.SYS file is easier to manipulate since it looks and feels exactly the same as that of a traditional OS/2 Warp client.
- Existing clients' installed applications can be copied to the WSoD client image on the boot server (\IBMLAN\RPL\BB20.US) with little or no modification.

#### **11.1.2.9 Have a Fall-Back Option**

Before migrating client workstations to WorkSpace On-Demand, it is prudent to have some way to return to the previous "fat client" environment in the event that something delays the deployment, such as an unforeseen application problem. In this way, you can revert to the previous environment

and continue normal operations until the problem can be fixed and the WorkSpace On-Demand deployment can proceed. This is known as having a *fall-back* capability.

In most personal computers, the only change necessary to make them boot from the network is a modification to the machine's BIOS in order to change the default startup sequence. Should the client need to be booted from its own hard drive in "fat client" mode, it is simple to reconfigure the BIOS back to its original state

### 11.1.3 Testing

Comprehensive testing is an important phase of any deployment. The testing activities described here are only those related directly to deployment. We assume that you have previously tested your server configurations, client images, application profiles, and machine classes.

As with all testing activities, the individuals carrying out these production tests should be different from those performing the development and integration work.

Production testing can generally be classified in the following three categories:

- *System Integration Testing (SIT)* is typically done towards the end of the development cycle. This is a good opportunity to focus on and resolve any problems with system components such as network applications, machine classes, video or device drivers, printing, and peripheral support. Automation processes should be fully tested as well. A round of stress testing with the maximum number of clients on a single server is also a good idea.
- *User Acceptance Testing (UAT)* provides valuable feedback from the end user perspective. Normally, the best candidates to perform UAT are those individuals who are very familiar with the user environment and applications. Multiple stages of UAT may be required to fully test the system
- *Pilot deployments* in a "live" end user environment are carried out by most enterprises before commencing full-scale deployment. A pilot is highly recommended since it provides an opportunity to discover and rectify any problems not previously discovered during the SIT and UAT phases, and is also likely to unearth any procedural problems in the deployment planning as well as technical problems with the system itself. If such problems arise in a pilot environment, they can be fixed without impacting a large number of end users.

If the enterprise will use multiple client and server configurations, you should try to include each configuration in the pilot deployment. Alternatively, you may wish to conduct a series of pilots to encompass all required configurations.

You should schedule a sufficient amount of time between the pilot(s) and the start of full-scale deployment to ensure that any problems found can be addressed and resolved without impacting the production deployment schedule.

---

## **11.2 Infrastructure Considerations**

When planning to install WorkSpace On-Demand in an existing network environment, you will need to understand the existing network infrastructure to ensure that your WorkSpace On-Demand installation proceeds smoothly and does not unduly disrupt the existing operation of the network. A large enterprise may have multiple physical LANs connected by devices such as bridges and routers, with many clients running numerous network protocols. Many of these factors will influence WorkSpace On-Demand's operation in such an environment.

### **11.2.1 Current Hardware Inventory**

If you want to migrate existing "fat client" users to WorkSpace On-Demand, it makes sense to utilize as much of the existing client hardware as possible. As part of your preinstallation planning, you should therefore create an inventory of existing hardware configurations so that you can determine the work needed to support these configurations in a WorkSpace On-Demand environment. See Section 10.1.6, "Planning Your Machine Classes" on page 256, for more information on planning your hardware support.

### **11.2.2 Using Multiple Boot Mechanisms on a Single Server**

We recommend that you run only a single boot mechanism (either IEEE 802.2 RPL or DHCP PXE) on a particular WorkSpace On-Demand server. Testing has shown that over time, the performance of a server booting clients using both boot mechanisms tends to degrade, resulting in longer boot times for IEEE 802.2 RPL clients. While rebooting the server rectifies the problem, it is better to run only a single boot mechanism on any one server.

### **11.2.3 Number of Clients Per Server**

The number of client workstations that can practically be serviced by a WorkSpace On-Demand server is determined by a number of factors including:

- Whether the clients use local or remote swapping
- The effective network bandwidth, which may in turn be affected by the network topology
- The processor speed of the server
- The file system and cache size on the server
- Other services (such as file or database serving) that are running on the server

The practical limit on the number of clients per server is typically determined by the users' perception of acceptable boot performance, rather than by a physical limitation. You will need to evaluate how the different combinations of determining factors affect your environment, and determine an acceptable ratio of clients to servers. In situations where the number of clients results in an unacceptable server performance, you may wish to consider using multiple servers.

#### **11.2.4 Network Topology**

If the network topology differs between LAN segments, you must ensure that you place a WorkSpace On-Demand server on each segment since the server must have the same network topology as the clients that it supports.

#### **11.2.5 Network Traffic Profiles**

When a client boots from a WorkSpace On-Demand server, it requires approximately 8 MB of data to be transferred over the network. If you add additional device drivers and middleware to the client image, the amount of data will increase accordingly. This means that when you implement WorkSpace On-Demand, the traffic on your network will increase, at least during the period in which clients are booting. If you already have a high network load, you should consider the implications of this additional traffic.

However, if much of your existing traffic is based on applications running on clients that are to be migrated to WorkSpace On-Demand, this may not be a problem. The network load imposed by WorkSpace On-Demand will decrease significantly after the client boot process is completed, and the application traffic can then continue more or less as usual. However, if you plan to bring new clients with new WorkSpace On-Demand applications into an existing network with a heavy traffic load, you may need to restructure your network to avoid congestion.

## 11.2.6 Network Protocols

WorkSpace On-Demand requires the NetBIOS or TCPBEUI protocols to be loaded into the client's memory. If you are not already using NetBIOS, this will add an additional protocol in your network. To minimize NetBIOS traffic, you can configure your WorkSpace On-Demand clients to use NetBIOS only to load WorkSpace On-Demand and to use your existing protocols for application traffic.

## 11.2.7 Routers and Bridges

Many large networks are divided into multiple physical segments, connected by devices such as routers and bridges. While this kind of segmentation can result in significant improvements in performance and manageability, it does introduce some additional considerations when deploying WorkSpace On-Demand.

### 11.2.7.1 Routers

WorkSpace On-Demand uses the NetBIOS or TCPBEUI protocols to boot its clients. The NetBIOS (BETBEUI) protocol is not routable. If you use this protocol as your boot protocol and you have LAN segments connected by routers, you must have a WorkSpace On-Demand server in every LAN segment, regardless of other factors.

For this reason, you may wish to use TCPBEUI as your boot protocol in preference to NetBIOS, since you can then boot a client from a WorkSpace On-Demand server located on a different LAN segment, connected to the client through a router. If you do this however, you should note that the same considerations described below for bridges apply equally to routers.

Note also that in order to support the DHCP PXE boot mechanism across a router, the router must have boot relay agents installed and operational.

### 11.2.7.2 Bridge Congestion

Figure 120 on page 309 shows an example of a logical LAN with multiple segments connected by bridges.

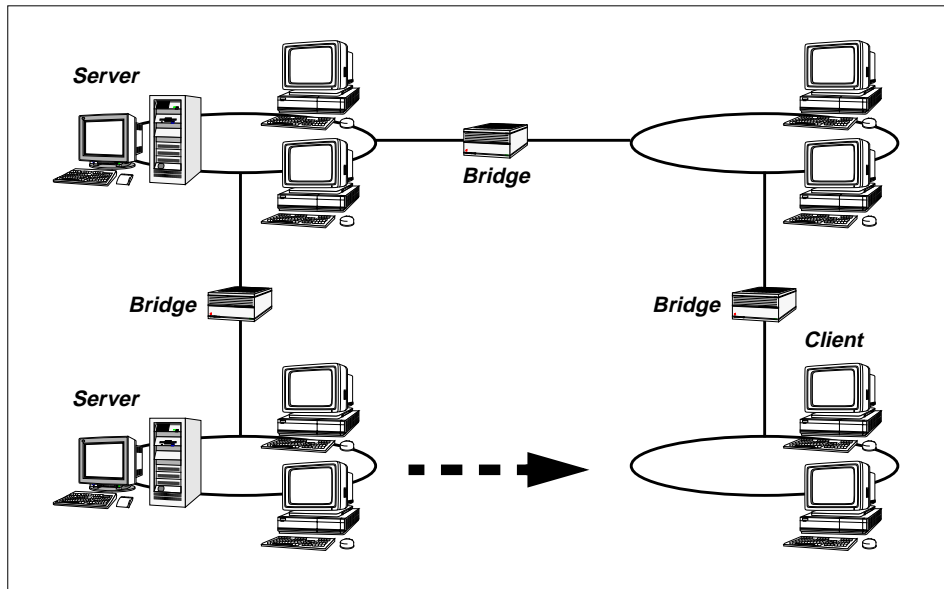


Figure 120. LAN Segments Connected by Bridges

Note that the WorkSpace On-Demand servers for this network are located on one segment, while many of the clients reside on other segments. Such a network can cause problems when there is heavy network traffic and the bridges become congested, since a congested bridge may discard a frame instead of retransmitting it. The WorkSpace On-Demand client can recover from a lost frame in some instances, but not in all cases, and lost frames can cause a client's boot process to fail.

Whenever possible, avoid having bridges between your WorkSpace On-Demand server and its clients. If this is impossible, try to minimize the number of bridges (hop count) between the server and the client. For example, you can reduce the hop count by designing the network structure as shown in Figure 121 on page 310.

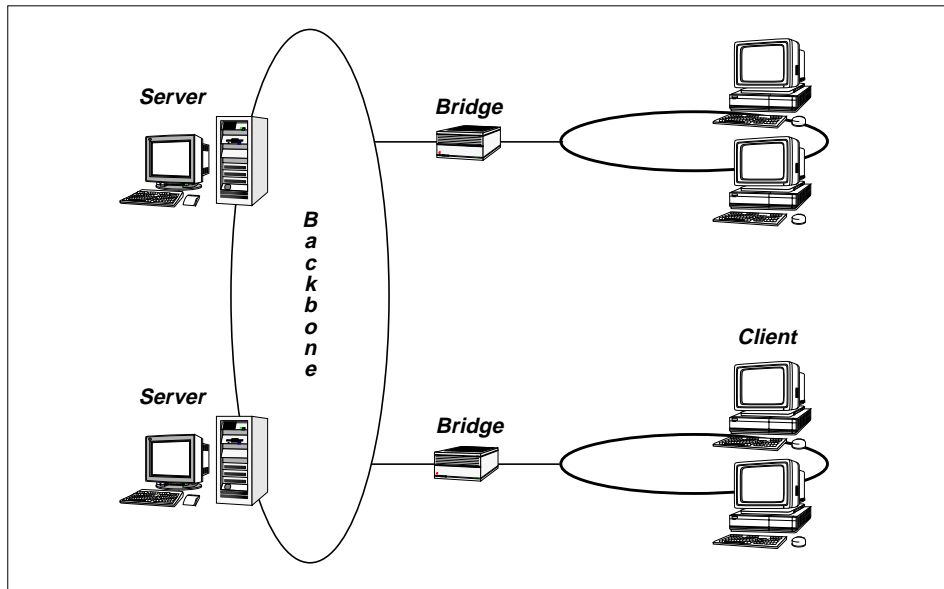


Figure 121. LAN Segments Connected by a Backbone

In the network shown above, the maximum hop count is two, with a maximum of one bridge between a client and its WorkSpace On-Demand server. This reduces the likelihood of lost frames and failed boot attempts.

#### 11.2.7.3 Remote Bridges

If you have LAN segments in separate locations connected by remote bridges over a leased line with a low bandwidth, you should place a WorkSpace On-Demand server in every location since the bridge/line capacity is unlikely to provide acceptable performance. If you have a line-speed comparable to a LAN transmission rate, you may try to connect a WorkSpace On-Demand client to a server across the remote bridge, but it is still advisable to place a server in each location. You can manage the different servers from a central site using the WorkSpace On-Demand administration tool.

#### 11.2.7.4 Broadcast Filters

If you set filters to avoid broadcast transmissions across bridges, you will need to disable these filters if you want to locate clients on a different segment from their WorkSpace On-Demand servers. The RPL FIND requests used by WorkSpace On-Demand will not cross bridges with broadcast filters enabled. If you want to leave the filters active in your bridges, you must locate a WorkSpace On-Demand server on each LAN segment.



---

## 11.3 Performance Planning for WorkSpace On-Demand

This section discusses the issues related to WorkSpace On-Demand performance from two perspectives: server and client.

### Note

See Section 11.2, "Infrastructure Considerations" on page 306, for a discussion of the network issues that directly affect WorkSpace On-Demand performance.

### 11.3.1 Server Performance Considerations

When you consider that all WorkSpace On-Demand clients obtain their applications and data from the WorkSpace On-Demand server, the importance of providing a robust and responsive server platform becomes obvious. The following issues should be considered in order to obtain maximum performance and responsiveness from the WorkSpace On-Demand server.

#### 11.3.1.1 Processor

You should ensure that your WorkSpace On-Demand server has at least a 90 MHz Pentium, or compatible, processor.

#### 11.3.1.2 Memory

You should ensure that your WorkSpace On-Demand server has sufficient memory to accommodate the requirements of OS/2 Warp Server and its services, as listed in Table 11 on page 104. However, your server may require additional memory for optimal performance depending upon the number of clients it is supporting.

You should ensure that your server has sufficient memory and that the HPFS386 cache is configured to service approximately 95 percent of all read requests from cache. This will require 12-14 MB of cache memory for the client operating system image, plus additional memory for client-specific files, application code and data, depending on your particular installation.

#### 11.3.1.3 Disk I/O Subsystem

Because WorkSpace On-Demand clients receive all of their system and application code from the server, client performance is directly affected by the speed, responsiveness and throughput of the server's disk subsystem. For this reason, the WorkSpace On-Demand server should be equipped with the most robust disk subsystem, in terms of access time and total data

throughput, that is practical for your installation. There is also the issue of data integrity to consider, which may dictate implementing RAID on your WorkSpace On-Demand server(s). Some of the factors relating to disk subsystem selection that you may wish to consider include:

- Controller Type

The type of disk subsystem you choose to implement also effects the overall performance of your WorkSpace On-Demand installation. We recommend a SCSI controller for the following reasons:

- Bandwidth. (*Wide SCSI 16-bit* or *Ultra-wide SCSI 32-bit* data path)
- *Tagged* command queuing (overlapping disk I/O)

- Disk Performance

Total disk performance is an important consideration in optimizing overall system performance. In general, a greater throughput rate is the more critical selection criterion; however, reduced access time contributes to overall system performance as well.

- Sustained throughput rate
- Total access and seek times

- File system

OS/2 Warp Server gives you three choices of file system:

- HPFS386 - (recommended)
- HPFS
- FAT

The choice of which file system to implement is also an important performance consideration. We recommend implementing an HPFS386 file system for several reasons. The most important reason is that the HPFS386 file system provides the capability of configuring an HPFS386 disk cache large enough to accommodate a client's software image and its application(s). Loading a client's software image and applications from RAM instead of disk is a great performance enhancement. A typical WorkSpace On-Demand's image is approximately 14 MB in size.

As a general rule-of-thumb, you should plan on 48 MB as a minimum memory requirement, with 64 MB as a more typical WorkSpace On-Demand server configuration. However, as you will see in the discussion that follows, optimal memory size should be determined by the size of the HPFS386 cache, assuming that you choose to follow our recommendation and implement a HPFS386 file system on your WorkSpace On-Demand server. The amount of memory available to a

WorkSpace On-Demand server determines how much memory can be allocated to the 386 HPFS cache.

We recommend that the size of the HPFS386 cache should be made large enough so that at approximately 95% of all read requests are serviced by the cache. You can use the /STATS parameter of the CACHE386 command to query cache-hit statistics.

For additional information on the HPFS386 cache, refer to the *LAN Server Network Administrator Reference Volume 2: Performance Tuning*, S10H-9681.

The percentage of cache read hits will increase as the HPFS386 cache increases in size so that more and more of the WorkSpace On-Demand image is contained in within the cache. Things that increase the size of the WorkSpace On-Demand image include:

- Number of supported machine classes
  - Number of supported clients
  - Number of supported users
  - Number of resource definitions, aliases, network printers, home directories, and so forth
  - Number of supported applications
  - Complexity and size of supported applications
  - Size and complexity of FIT files
  - Therefore, on a robust WorkSpace On-Demand server, it is quite possible that 64 MB of memory is not sufficient to insure that 95 percent of disk read requests are serviced from cache; so you should plan your WorkSpace On-Demand server's memory requirements accordingly.
- RAID

If data integrity and ease of recovery are important considerations, you should consider implementing RAID on your WorkSpace On-Demand server(s). While all the benefits and implications of implementing RAID are beyond the scope of this chapter, you should also consider that there may be some performance implications when implementing RAID as well. For example:

- RAID level
- Disk mirroring, disk duplexing, parity, and so forth
- Data striping

- Mirrored reads
- Write policy
- Write-through vs. write-back cache

#### **11.3.1.4 Network Adapter**

We recommend the use of 32-bit Streaming LAN adapters because of their enhanced data throughput. You may also wish to consider using multiple network adapters on the same LAN segment for enhanced throughput and load balancing. However, you should note that in most cases on 10 Mbps Ethernet or 16 Mbps Token-Ring networks, the network bandwidth becomes a constraining factor before a single network adapter can be saturated. Adding a second adapter in such circumstances will not provide any performance enhancement. However, if you are implementing WorkSpace On-Demand in a 100 Mbps network, the adapter throughput may become a constraining factor before you achieve network saturation. You should evaluate the use of multiple adapters in your own environment.

### **11.3.2 Client Performance Considerations**

Ensuring that the WorkSpace On-Demand client is provided with sufficient memory to keep swapping to a minimum is the most effective means of improving performance. The next most effective means of improving performance is to provide a hard disk for use as a local swapping device.

From an end-user's perspective, the perception of how well a given WorkSpace On-Demand client's performs, depends, for the most part, on the responsiveness of the WorkSpace On-Demand server and the network infrastructure supporting it.

#### **11.3.2.1 Processor**

The optimum processor speed for your clients is primarily dependent upon the application workload that the client will support. However, we recommend a minimum 33 MHz Intel '486 or compatible processor.

#### **11.3.2.2 Memory**

If you configure your WorkSpace On-Demand clients to use a local hard disk for swapping, you should install a minimum of 16 MB of RAM in each client. If you configure your clients to swap across the network, you should install a minimum of 20 MB of RAM in each client.

**Note**

This parameter is application-dependent. The characteristics of the application(s) configured for a specific end-user may invalidate these recommendations. See Section 4.1.2.2, "Memory" on page 107, for more information on client memory requirements.

### 11.3.2.3 Disk

A hard disk is not mandatory in your WorkSpace On-Demand clients since WorkSpace On-Demand allows you to swap across the network. However, you can use a local hard disk for swapping, and we strongly recommend that you do so since local swapping reduces network traffic and server workload.

**Note**

The client's hard disk must be formatted before it can be used as a local swapping device. You should preformat the hard disk prior to installing WorkSpace On-Demand.

Note that the choice of file system (HPFS or FAT) does not measurably affect performance.

### 11.3.2.4 Network Adapter

The throughput of the client's network adapter is not normally a constraining factor on performance. However, because WorkSpace On-Demand makes significant demands on network bandwidth when a server is booting multiple clients concurrently, you should consider the use of *Wake On LAN* network adapters. With this type of adapter, a system administrator can program groups of WorkSpace On-Demand clients to boot at different times prior to the start of the business day. This avoids a "boot storm" by staggering the initialization process while still having all clients available for logon when the end-users arrive for work.

---

## 11.4 Automation

The deployment of WorkSpace On-Demand in a large enterprise network may consist of thousands of clients spread over several hundred locations. To successfully rollout and support a large scale WSoD environment, a high degree of automation is very important. Many of the common administration tasks, such as creating requesters, creating public applications and assigning public applications to users, are slow and labor intensive when done using

the LAN Server Administration GUI. The GUI is also error prone and can lead to inconsistencies on servers throughout the deployment since an administrator could easily enter incorrect data. Automation provides a cleaner and more stable deployment. This section provides practical examples of some ways in which common WorkSpace On-Demand administration tasks can be automated.

#### **11.4.1 Unattended Installation of WorkSpace On-Demand**

If a deployment of OS/2 Warp Servers already exists, an efficient way to upgrade to WorkSpace On-Demand is to run an unattended installation process. The unattended installation process can also be used to build new servers. Any necessary prerequisites, such as an OS/2 FixPak, and post-installation procedures, such as running GETRPL, can be built into an unattended CID process. The CD-ROM accompanying this redbook contains a sample CID LCU command file, named WSODINST.CMD, and the associated CID response files which will install the OS/2 RIPL component of LAN Server, the required FixPak for OS/2 and WorkSpace On-Demand, and then run the GETRPL utility.

Review Chapter 5, “Installing WorkSpace On-Demand” on page 115, for additional information on how to perform an unattended installation.

#### **11.4.2 Creating Client Workstations Automatically**

A major task during a WorkSpace On-Demand rollout is defining the client workstations on each server. This can be done using the LAN Server Administration GUI; however, it is a slow and tedious process. It is also error prone, since some of the parameter fields could be accidentally missed or filled in with incorrect data. A more efficient way is to use the network Command Line Interface (CLI). The `NET RIPLMACH` command can be used to create a client as described in Section 6.2, “Defining Client Workstations Automatically” on page 151.

The `NET RIPLMACH` command is long and tedious to type out. A more efficient way is to use REXX scripts and response files to generate and execute the `NET RIPLMACH` command. A REXX program named `ADDREQ.CMD` is included on the CD-ROM accompanying this redbook as an example of how the creation of clients can be automated. `ADDREQ.CMD` reads in a response file which contains all of the information necessary to create a client.

### 11.4.3 Creating Public Applications

The two key `NET` commands used in the creation of public applications are `NET APP` and `NET APPPARM`. Below is an example of a `NET APP` command which will create an OS/2 Window public application.

```
NET APP /ADD OS2WIN /REMARK:"OS/2^Window" /COMMAND:"CMD.EXE" /APPDIR:C:\OS2
/INTERFACE:PM /DOMAIN:DOMAIN1 /TYPE:WSOD
```

The `NET APPPARM` is used this way:

```
NET APPPARM OS2WIN NCC_SETUP_POST=ICONPOS=90,90;Title=
NET APPPARM OS2WIN NCC_FOLDER =
Title="Command^Prompts";ICONPOS=50,20;ICONRESOURCE=61,PMWP;
```

It would become very tedious for an administrator to type commands like the examples above every time a public application was added or modified.

A better method is to create a REXX program that generates and executes the necessary `NET` commands. A program named `ADDAPPL.COM` is included on the CD which accompanies this redbook. `ADDAPPL` reads in and parses a response file and then executes the necessary `NET` commands to create the applications.

In a large-scale environment, it is a good idea to build all public applications from a master response file. This helps to maintain consistency throughout the environment. Any new applications or changes to existing applications can be made in the master response file and `ADDAPPL` can be run to recreate the applications. This eliminates the possibility of the same applications with different definition on different servers.

`ADDAPPL` can be used to initially create public application definitions on a new `WSOD` server. It can also be used to modify existing applications on servers as user needs and requirements change over time. The best approach is to standardize the applications as much as possible.

Be mindful when using user-specific environment variables within application definitions. Customizing applications for different users, such as placing the icon in different positions on the desktop, starts to add complexity and makes administration more difficult.





---

## Part 5. Appendices

This part of the redbook includes code listings and sample response files for many of the automation techniques discussed in previous chapters.

Appendix A, "Automated Client Definition Programs" on page 321 provides examples of REXX programs that allow you to query client hardware configurations and automatically create client definitions under WorkSpace On-Demand.

Appendix B, "Application Definition Scripts and Response Files" on page 329 includes examples of REXX programs and command files that allow you to automatically create network application definitions on your WorkSpace On-Demand servers.

Appendix C, "DOS and WIN-OS2 Settings" on page 335 provides a summary of the DOS and WIN-OS2 environment settings available to an application running on the WorkSpace On-Demand client.

Appendix D, "Video Handling" on page 339 describes the way in which various definition files interact to define the graphics capabilities of a WorkSpace On-Demand client.

Appendix E, "Tools and Utilities" on page 353 describes a number of the tools and utilities included on the CD-ROM that accompanies this redbook.



## Appendix A. Automated Client Definition Programs

This appendix contains source code listings for the REXX programs described in Chapter 6, "Working with Client Workstations" on page 137.

### A.1 Client Hardware Query Program – QUERYCFG.CMD

```
'z:'
'cd \util'
ok="new"
do while ok<>"
  say "Please enter the client name :"
  pull id
  id=translate(id)
  idfile= "Z:\" || id || ".data"
  okfile = "Z:\" || id || ".OK"
  call stream okfile, 'c', 'query exists'
  ok=result
  if ok<>" then
  do
    say ""
    say "Duplicate name, Please try again !"
    say ""
  end /* do */
end /* do */

say "Querying system information. "
say "Please wait..."
'cd \netfin.40\bin'
cmd = 'sinfg30 /P:' || idfile ||' /B /NOLOGO'
cmd
cmd = 'echo =====>>' idfile
do 3
  cmd
end
cmd = ' echo LAN adapter info added by LANTRAN.LOG >> 'idfile
cmd
cmd = 'echo =====>>' idfile
do 3
  cmd
end
cmd = 'type lantran.log >> ' idfile
cmd

okfile = "Z:\" || id || ".OK"
```

```

say "Waiting for Server Process to create my definition"
do while stream(okfile,'c','query exists') = ""
  NOP
end
'Tshutdwn'

```

---

## A.2 Client Definition Program – ADDNEWCL.CMD

```

/* Sample program to automatically add new clients to an
   existing WorkSpace On-Demand Server */

call RxFuncAdd 'SysLoadFuncs', 'RexxUtil', 'SysLoadFuncs'
call SysLoadFuncs

/* This program requires an administrator LOGON */
'logon admin /p:wsod /v:d'

NewClBase='D:\NEWCLTS' /* Base Directory for Client.DATA */
RPL='d:\ibmlan\rpl\rpl.map'
AdpLine='Adapter 0 is using node address '
LogFile1='D:\NEWCLTS\NEWCLTS.LOG'

starttime=time()
stardate=date()
/* The names are copied from
   D:\IBMLAN\RPL\BB10.US\IBMCOM\MACS\*.NIF */

AdpList.0=2
AdpList.1.model="MODEL"
AdpList.1.name="IBM PCI Token-Ring Adapter (IBMTRP.OS2)"
AdpList.1.dummy="00062???????"
AdpList.2.model="MDGMODEL"
AdpList.2.name="Madge FastMac OS/2 NDIS driver for Smart 16/4 Ringnodes"
AdpList.2.dummy="0000F???????"

do while(1) /* Run forever */
  say "AddClts: [StartDate : "Startdate"/"Starttime "],
      " [Last Scan:"date()"/"time() "]"
  Cltlist.0=0
  call SysFileTree NewClBase'\*.data', 'CltList', 'SFO'
  i=0
  if CltList.0 <> 0 then
  do
    call Logmsg "*****"
    call logmsg "**** Creating New Client ****"

```

```

call Logmsg "*****"
do CltList.0
  i=i+1
  cltfile=filespec("name",CltList.i);
  CltList.i=left(cltfile,pos('.',cltfile)-1)
  call logmsg "Found data file :\"CltList.i
end /* do */

/* CltList.0 contains the number of .DATA files found */

actclt.0=0
i=0
do while queued() > 0
  pull entry
end /* do */
'net riplmach | rxqueue' /* Querying info to pipe */
/* Reading list of active clients */

do while queued() > 0
  pull entry
  i=i+1
  actclt.i=entry
  say "actClt.\"i\" = \"actclt.i
  actclti = translate(actclti)
  if pos('ENABLED',actclti) = 0 & pos('DISABLED',actclti) = 0 then
    do
      i=i-1
    end
  else
    do
      /* reduce line to the client's name */
      actclti=word(actclti,1)
      call logmsg "Active Client <i> :\"actclt.i
    end
  end /* do */
actclt0=i
/* Testing the .DATA files against the existing clients */
i=0
do CltList.0
  found=0
  i=i+1
  do j=1 to actclt.0
    if CltList.i = actclt.j then
      do
        found=1
        leave
      end /* do */

```

```

        end /* do */
        if found=0 then
        do
            call logmsg "Client name :\"ClList.i
            call ScanClList.i
            call CreateClList.i
/* Now rename the .DATA file to .OK
            to show it's correctly created
            and to reduce scan time for the
            next run */
            'ren 'NewClBase'\\"ClList.i'.DATA *.OK'
        end /* do */
        end /* do */
    end
    call SysSleep 60 /* Waiting one minute */
end /* do */

exit

/*****/

LogMsg: procedure expose Logfile1
parse arg msg
if (stream( Logfile1 , 'c', 'query exists') = "") then
do
    call stream Logfile1, 'c', 'open write'
    call lineout Logfile1 ,,1
end
else
do
    call stream Logfile1, 'c', 'open write'
    call lineout Logfile1, ""
end
msg = '[' || date() || ' ' || time() || ' ] ' || msg
say msg
call lineout Logfile1, msg
call stream Logfile1, 'c', 'close'
return
/*****/

CreateClList: procedure expose NewClListBase RPL AdpLine ,
                Logfile1 address AdpList.

parse upper arg clListName

tmpRPL='D:\IBMLAN\RPL\rpl.tmp'

```

```

'del 'tmp_rpl

call stream RPL , 'c' , 'open read'
if result <>'READY:' then
do
    call LogMsg 'Error opening file : 'RPL
    exit x2d('0812',4)
end /* do */
call stream tmp_rpl , 'c' , 'open write'
if result <>'READY:' then
do
    call LogMsg 'Error opening file : 'tmp_rpl
    exit x2d('0812',4)
end /* do */

call LogMsg "Reading: "RPL
call LogMsg "Writing: "tmp_rpl

adptype=""

do while lines(RPL)
    dataline=linein(RPL)
    if pos(address,dataline) = 0 then
do
        call lineout tmp_rpl, dataline
    end /* do */
    else
do
        /* Adapter Address line found */
        call LogMsg "Address entry found !"
        call LogMsg "dataline : "dataline
        /* The second field contains the name of the model */
    end /* do */
    adp=0
    do AdpList.0
        adp=adp+1
        if pos(AdpList.adp.dummy,dataline) <> 0 then
do
            tmpname=word(dataline,2)
            call QueryAdp(tmpname)
        end /* do */
    end /* do */
end /* do */
call stream RPL , 'c', 'close'
call stream tmp_rpl , 'c', 'close'
cmd = 'copy 'tmp_rpl RPL

```

```

cmd
if rc<>0 then
do
    call LogMsg "Error copying files: "cmd
    exit x2d('1604',4)
end /* do */
class = 'ISAVGA'
remark = '"WsoD Client ' || cltname ||'"'
call LogMsg "Creating new Client"
call LogMsg "Client Name      : " cltname
call LogMsg "Adapter-type     : " adptype
call LogMsg "Adapter address : " address
call LogMsg "Machine class  : " class
call LogMsg "Remark         : " remark

cmd = 'net ripmach ' ,
      || cltname ,
      || ' /add /os:bb10.us ' ,
      || ' /c:'class ,
      || ' /mac:'address ,
      || ' /rem:'remark ,
      || ' /ada:"'adptype || '" ' ,
      || ' 1>>'LogFile1 ,
      || ' 2>>&1'
call LogMsg "Using command : "cmd
cmd
rc2=rc
if rc2<>0 then
do
    call Logmsg "Error creating new client : "cltname
    call LogMsg "Return Code : "rc2
    return
end /* do */
else
do
    call LogMsg "Client <"cltname"> succesfully defined !"
    return
end /* do */

/*****/

ScanCl: procedure expose NewClBase RPL AdpLine ,
                Logfile1 address AdpList.

parse upper arg cltname

address = ""

```



```

cltdata=NewCltdBase || '\' || cltname || '.DATA'

call stream cltdata , 'c' , 'open read'
if result <> 'READY:' then
do
    call LogMsg 'Error opening file : 'cltdata
    exit x2d('0812',4)
end /* do */
call LogMsg "Client's Data file : "cltdata
do while lines(cltdata)
    dataline=linein(cltdata)
    if pos(AdpLine,dataline) <> 0 then
    do
        /* Adapter Address line found */
        address=right(dataline,13)
        /*The end of line is a '.' */
        address=left(address,12)
        leave
    end /* do */
end /* do */
call stream cltdata , 'c', 'close'
return

/*****/

QueryAdp: procedure expose AdpList. Logfile1 adptype

parse upper arg cltname
i=1
do AdpList.0
    if AdpList.1.model = cltname then
    do
        adptype=AdpList.i.name
        leave
    end
end /* do */
if adptype="" then
do
    call logmsg "No definition in AdpList. found "
    exit x2d('1604',4)
end /* do */
return

```



## Appendix B. Application Definition Scripts and Response Files

This appendix provides aREXX code listing and sample response file to define a series of network public applications, as discussed in Section 7.4, "Defining the Application Using the Command Line Interface" on page 193.

### B.1 Application Definition Script – ADDAPPL.CMD

```
/* Create WorkSpace On-Demand application definitions
   with response files */
call RxFuncAdd 'SysLoadFuncs', 'RexxUtil', 'SysLoadFuncs'
call SysLoadFuncs

Logfile   = ""
RspFile   = ""
BootDrive = ""

parse upper arg parms

call Defaults
call ParseArgs(parms)
say "Logging to :" LogFile
call LogMsg "Invocation parameters: "parms

select
  when RspFile = "" then
  do
    call Logmsg "/R parameter missing"
    call Usage
  end /* do */
  otherwise
  do
    call Logmsg "Logfile       : " || Logfile
    call Logmsg "ResponseFile : " || RspFile
  end /* do */

end /* select */
call Logmsg "Processing Response-File"
call CreateApp

exit

/*****/

Defaults: procedure expose BootDrive Logfile

  BootDrive = 'E:'
  Logfile   = BootDrive || "\MC_Class.log"
  return
/*****/
CreateApp: procedure expose Logfile RspFile

  call stream RspFile, 'c', 'open read'
  if result <> "READY:" then
  do
    call Logmsg "Error opening responsefile " || RspFile
    call Logmsg "exiting program"
```

```

        exit -1
    end

do while lines(RspFile)
    appline=linein(RspFile)
    if appline="" | left(appline,1)="*" then
        iterate
    if pos("[",appline) <> 0 then
        do
            call Logmsg "Application entry found"
            param.0 = 0
            wsodparam.0 = 0
            ws = 0
            i = 0
            usr = 0
            do while left(appline,1) <> '(' /* Data starts after ( */
                appline= translate(linein(RspFile))
                iterate
            end
            appline = translate(linein(RspFile))
            do while left(appline,1) <> ')' /* Data ends with ) */
                if left(appline,1) = '*'then
                    do
                        appline = translate(linein(RspFile))
                        iterate
                    end /* do */
                    if left(appline,2) = 'ID'then
                        do
                            applid = word(appline,3)
                            appline = translate(linein(RspFile))
                            iterate
                        end /* do */
                        if pos('PARAM', appline) <> 0 then
                            do
                                ws = ws + 1
                                wsodparam.0 = ws
                                wsodparam.ws.p = word(appline,2)
                                wsodparam.ws.v = word(appline,4)
                            end /* do */
                        else
                            if pos('USER', appline) <> 0 then
                                do
                                    usr = usr + 1
                                    wsodusr.0 = usr
                                    wsodusr.usr.u = word(appline,3)
                                end /* do */
                            else
                                do
                                    i=i+1
                                    param.0 = i
                                    param.i.p = word(appline,1)
                                    param.i.v = word(appline,3)
                                end /* do */
                                appline= translate(linein(RspFile))
                            end /* do */
                        call Logmsg "Application found : "applid
                    cmd = 'cmd /c net app /add ' || applid
                    j=0
                    do param.0
                        j = j + 1
                        cmd = cmd || ' /' || param.j.p || ':' || param.j.v
                    end /* do */
                    cmd = cmd || ' 1>>'Logfile || ' 2>>&1'
                end
            end
        end
    end
end

```

```

call logmsg "Executing command " cmd
cmd
rc2=rc
if rc2 <> 0 then
do
    call logmsg "Error executing command "
    call stream RspFile , 'c' , 'close'
    call logmsg "RC = "rc2
    exit x2d('1604',4)
end /* do */
if ws <> 0 then
do
    j=0
    do wsodparam.0
        j=j+1
        cmd = 'cmd /c net appparm ' || applid || ' ' ,
            || wsodparam.j.p || '=' ,
            || wsodparam.j.v ,
            || ' /ADD /F:E ' ,
            || ' /PLACEMENT:E' ,
            || ' 1>>' Logfile || ' 2>>&1'
        call logmsg "Executing command " cmd
        cmd
        rc2=rc
        if rc2 <> 0 then
        do
            call logmsg "Error executing command "
            call logmsg "RC = "rc2
            call stream RspFile , 'c' , 'close'
            exit x2d('1604',4)
        end /* do */
    end /* do */
end /* do */
if wsodusr.0 <>0 then
do
    j=0
    do wsodusr.0
        j=j+1
        cmd = 'cmd /c net user ' || wsodusr.j.u ,
            || ' /ASSIGN PUBLIC:' || applid
        call logmsg "Assigning appl. to user :" cmd
        cmd
        /* Waiting for net.exe removed from memory */
        call SysSleep 2
        rc2=rc
        if rc2 <> 0 then
        do
            call logmsg "Error executing command "
            call logmsg "RC = "rc2
            call stream RspFile , 'c' , 'close'
            exit x2d('1604',4)
        end /* do */
    end /* do */
end /* do if wsodusr.0 <> 0 */
end /* do */

call Logmsg "Applications successully created "
call stream RspFile , 'c' , 'close'
return
/*****/

ParseArgs: procedure expose MC_Model RspFile MC_Name Logfile ReadMc

```

```

parse arg parms

parms = translate(parms)
numparms = words(parms)
do i=1 to numparms by 1
  parm = word(parms,i)
  select
    when left(parm,3) = "/R:" then
      do
        say "/R: Parameter found: "
        RspFile=right(parm,length(parm)-3);
        say "RspFile value : "RspFile
      end
    when left(parm,3) = "/L:" then
      do
        say "/L: Parameter found: "
        Logfile=right(parm,length(parm)-3);
        say "Logfile value : " Logfile
      end
    when left(parm,2) = "/"? then
      do
        call Usage
      end
    otherwise
      do
        say "Incorrect parameter was used"
        call Usage
      end /* do */
  end
end
return
/*****/

Usage:

say ""
say "Correct Program Invocation"
say "*****"
say "ADDAPPL [parameters]"
say ""
say "The following parameters are possible:"
say ""
say "/R:Responsefile"
say "  The parameters for the new applications"
say "/L:Logfile"
say "  The processing is logged in the given Logfilename"
say "  Defaults to <BootDrive>:\MC_Class.log"
say "/"?
say "This output is shown."
exit x2d("1600",4)
/*****/

LogMsg: procedure expose Logfile
parse arg msg

if (stream( Logfile , 'c', 'query exists') = "") then
do
  call stream Logfile, 'c', 'open write'
  call lineout Logfile ,,1
end
else
do

```

```

    call stream Logfile, 'c', 'open write'
    call lineout LogFile, ""
end
msg = '[' || date() || ' ' || time() || ' ] ' || msg
say msg
call lineout Logfile, msg
call stream Logfile, 'c', 'close'
return

```

---

## B.2 Application Definition Response File – ADDAPPL.RSP

```

*Sample Response file for adding applications
[App]
(
* a simple command line
ID = Wsod_CMD
TYPE = WSOD
INTERFACE = VIO
APPDIR = PROGRAM
COMMAND = "CMD.EXE"
APPDRIVE = K:
DOMAIN = WSOD_1
PARAM DPATH = K:\
REMARK = "Command Line"
USER = USR1
USER = USR2
)

[App]
(
* PM Camera
ID = PMCamera
APPDIR = PROGRAM\PMCAM200
COMMAND = "PMCAM200.EXE"
TYPE = WSOD
INTERFACE = PM
APPDRIVE = P:
REMARK = "PM Camera"
DOMAIN = WSOD_1
PARAM DPATH = D:\
PARAM LIBPATH = D:\
USER = USR3
USER = USR4
)

```





## Appendix C. DOS and WIN-OS2 Settings

Table 29 provides complete listing of DOS and WIN-OS2 settings and their allowed values. You can use these settings when defining applications in conjunction with the Parameters page in the Application Definition notebook, or when using the `NET APPARM` command. See Chapter 7, "Working with Network Applications" on page 167, for more information on defining network applications.

Table 29. DOS and WIN-OS2 Settings

Keyname	Choice
WIN_DDE	1 0
WIN_CLIPBOARD	1 0
AUDIO_ADAPTER_SHARING	1 0
COM_DIRECT_ACCESS	1 <0>
COM_HOLD	1 <0>
COM_RECEIVE_BUFFER_FLUSH	Valid settings: <NONE> ALL RECEIVE DATA INTERRUPT ENABLE SWITCH TO FOREGROUND
COM_SELECT	Valid settings: <ALL> COM1 COM2 COM3 COM4 NONE
DOS_AUTOEXEC	C:\AUTOEXEC.BAT Use full BATCH file name. You can also pass parameters.
DOS_BACKGROUND_EXECUTION	<1> 0
DOS_BREAK	1 <0>
DOS_DEVICE	Default: empty. Remember to separate any used with "," for new line.
DOS_FCBS	Limits: 0-255, default 16
DOS_FCBS_KEEP	Limits: 0-255, default 8
DOS_FILES	Limits: 20-255, default 20
DOS_HIGH	1 <0>
DOS_LASTDRIVE	Limits: last physical drive to Z, default Z

Keyname	Choice
DOS_RMSIZE	Limits: 128-640, default 640, increments of 16
DOS_SHELL	Default: "?:\OS2\MDOS\COMMAND.COM " "?:\OS2\MDOS\ /P" where ? is the boot drive.
DOS_STARTUP_DRIVE	Default: empty Accepts text; like A: or C:\DISKS\DRDOS.IMG
DOS_UMB	1 <0>
DOS_VERSION	Default: DCJSS02.EXE,3,40,255 DFIA0MOD.SYS,3,40,255 DXMA0MOD.SYS,3,40,255 IBMCACHE.COM,3,40,255 IBMCACHE.SYS,3,40,255 ISAM.EXE,3,40,255 ISAM2.EXE,3,40,255 ISQL.EXE,3,40,255 NET3.COM,3,40,255 EXCEL.EXE,10,10,4 PSCPG.COM,3,40,255 SAF.EXE,3,40,255 WIN200.BIN,10,10,4 Remember what you put here will replace the existing list of items; so be careful. Also remember to use carets in front of any commas you need. For example: SET DOS_VERSION=IBMCACHE.SYS,3,40,255
DPMI_DOS_API	Valid settings: <AUTO> ENABLED DISABLED
DPMI_MEMORY_LIMIT	Limits: 0-512, default 4
DPMI_NETWORK_BUFF_SIZE	Limits: 1-64, default 8
EMS_FRAME_LOCATION	Valid settings: <AUTO> NONE C000 C400 C800 CC00 D000 D400 D800 DC00 8000 8400 8800 8C00 9000
EMS_HIGH_OS_MAP_REGION	Limits: 0-96, default 32, note increments of 16
EMS_LOW_OS_MAP_REGION	Limits: 0-576, default 384, note increments of 16

Keyname	Choice
EMS_MEMORY_LIMIT	Limits: 0-32768, default 2048, note increments of 16
HW_NOSOUND	1 <0>
HW_ROM_TO_RAM	1 <0>
HW_TIMER	1 <0>
IDLE_SECONDS	Limits: 0-60, default 0
IDLE_SENSITIVITY	Limits: 1-100, default 75
INT_DURING_IO	1 <0>
KBD_ALTHOME_BYPASS	1 <0>
KBD_BUFFER_EXTEND	<1> 0
KBD_CTRL_BYPASS	Valid settings: <NONE> ALT_ESC CTRL_ESC
KBD_RATE_LOCK	1 <0>
MEM_EXCLUDE_REGIONS	Initially empty, you can specify a range of memory to exclude, or you can supply a single address for the beginning of a 4 KB region. If you need several regions, separate them with a comma (don't forget to use the caret since commas are special setup string parameters). Example: SET MEM_EXCLUDE_REGIONS=C0000, D0000-D8000
MEM_INCLUDE_REGIONS	Initially empty, you can specify a range of memory to include, or you can supply a single address for the beginning of a 4 KB region. If you need several regions, separate them with a comma (don't forget to use the caret since commas are special setup string parameters). Example: SET MEM_INCLUDE_REGIONS=C0000, D0000-D7FFF NOTE: The include region D0000-D8000 will include the entire memory between D8000 and D8FFFF.
MOUSE_EXCLUSIVE_ACCESS	1 <0>

Keyname	Choice
NETWARE_RESOURCES	Valid settings: NONE PRIVATE GLOBAL Special note: You use the words to change the value, BUT the string MUST be 7 characters long! Example: SET NETWARE_RESOURCES=GLOBAL
PRINT_SEPARATE_OUTPUT	<1> 0
PRINT_TIMEOUT	Limits: 0-3600, default 15
TOUCH_EXCLUSIVE_ACCESS	1 0
VIDEO_8514A_XGA_IOTRAP	<1> 0
VIDEO_FASTPASTE	1 <0>
VIDEO_MODE_RESTRICTION	Valid settings: <NONE> CGA MONO Special note: You use the words to change the value, BUT the string MUST be 15 characters long! Example: SET VIDEO_MODE_RESTRICTION=CGA
VIDEO_ONDEMAND_MEMORY	<1> 0
VIDEO_RETRACE_EMULATION	<1> 0
VIDEO_ROM_EMULATION	<1> 0
VIDEO_SWITCH_NOTIFICATION	1 <0>
VIDEO_WINDOW_REFRESH	Limits: 1-600, default 1
XMS_HANDLES	Limits: 0-128, default 32
XMS_MEMORY_LIMIT	Limits: 0-16384, default 2048, increment of 4
XMS_MINIMUM_HMA	Limits: 0-63, default 0

## Appendix D. Video Handling

When creating a new machine class to support a video adapter, it is useful to understand how the WorkSpace On-Demand client operating system handles the configuration of video monitors, screen resolution and color depth.

---

### D.1 Video Modes

A video mode is defined as the combination of a screen resolution, color depth and refresh rate. The screen resolution is measured in pixels, the color depth in number of colors, and the refresh rate in cycles per second (Hz). A video adapter's ability to support particular video modes is determined in part by the amount of memory, or video RAM, available to it. For example:

- 16 colors = 4 bits (0.5 bytes) per pixel
- 256 colors = 8 bits (1 byte) per pixel
- 64K colors = 16 bits (2 bytes) per pixel
- 16M colors = 24 bits (3 bytes) per pixel

Therefore: 640 pixels X 480 pixels X 16 colors equates to 640 x 480 x 0.5 bytes = 153,600 bytes, which can be supported by 0.5MB of video RAM.

The video modes that can be supported by a particular video adapter are listed in the SVGADATA.PMI file that is typically shipped with the adapter or generated dynamically by the SVGA.EXE utility when the display drivers are installed. Occasionally, however, you may find errors in the SVGADATA.PMI file. For example, the file that accompanied the CL-GD546X video chip set in our machine classing example included a number of video modes that required 4 MB of video RAM, even though the adapter built into the IBM PC 300GL Model 42U has only 2 MB of video RAM. This means that the Hardware page in the Client Definition notebook allowed us to define video modes that the adapter could not support and necessitated editing the SVGADATA.PMI file to remove these modes. See Section D.3, "Editing SVGADATA.PMI" on page 345, for more information.

#### Note

If you using a display driver that supports the GRADD architecture, you will have a GRADD.MOD file, which is supplied by the adapter manufacturer or generated dynamically by the MODEINFO.EXE utility.

Table 30 shows the approximate amounts of video RAM required to support different video modes. Note that specific values will change slightly based on the chip set in question and might not fit into the value listed in Table 30.

Table 30. Video Resolutions and Video RAM

Width (Pixels)		Height(Pixels)		Colors	Video RAM R
640	X	480	X	16	0.5 MB
				256	0.5 MB
				64 K	1 MB
				16 M	2 MB
800	X	600	X	16	0.5 MB
				256	0.5 MB
				64 K	1 MB
				16 M	2 MB
1024	X	768	X	16	0.5 MB
				256	1 MB
				64 K	2 MB
				16 M	4 MB
1280	X	1024	X	16	1 MB
				256	2 MB
				64 K	4 MB
				16 M	8 MB
1600	X	1280	X	16	1 MB
				256	2 MB
				64 K	4 MB
				16 M	8 MB

**Note**

One resolution that has given us trouble during testing is 800 x 600 x 64K. Simple mathematics shows that 960000 bytes of video RAM are required to support this video mode, which is slightly less than 1 MB. However, certain chip sets would not support this mode with only 1 MB of video RAM.

When in doubt, always consult the manufacturer's documentation to determine the video modes that are supported by a particular video adapter or chip set.

---

## D.2 Video Configuration Files

There are two files that are initially required in order to define video monitors, screen resolution, color depth, and refresh rate for a WorkSpace On-Demand client. These files are:

- The adapter definition file. For regular SVGA adapters, this file is named SVGADATA.PMI, and for GRADD adapters, it is named GRADD.MOD. The adapter definition file is either shipped with the adapter or generated dynamically using the SVGA.EXE or MODEINFO.EXE utilities.
- The monitor definition file, which is named MONITOR.DIF. This file resides in the \IBMLAN\RPL\BB20.US\OS2 directory. Some video adapters may provide their own MONITOR.DIF file, but you should discard this file and use the default file provided by the WorkSpace On-Demand client operating system.

The adapter definition file contains an enumerated list of all the video modes that a particular video adapter supports. The monitor definition file contains a similar list of known video monitors and the video modes supported by each monitor. By taking the intersection of these two files, it is possible to determine the video modes that a particular adapter/monitor combination is capable of supporting.

This information is stored in a file named VIDEO.CFG. The VIDEO.CFG file also contains a *modeId* field that defines the selected video mode in which a client will boot. In most cases, this file is provided along with a video adapter.

Figure 122 on page 342 shows the relationship between these video configuration files.

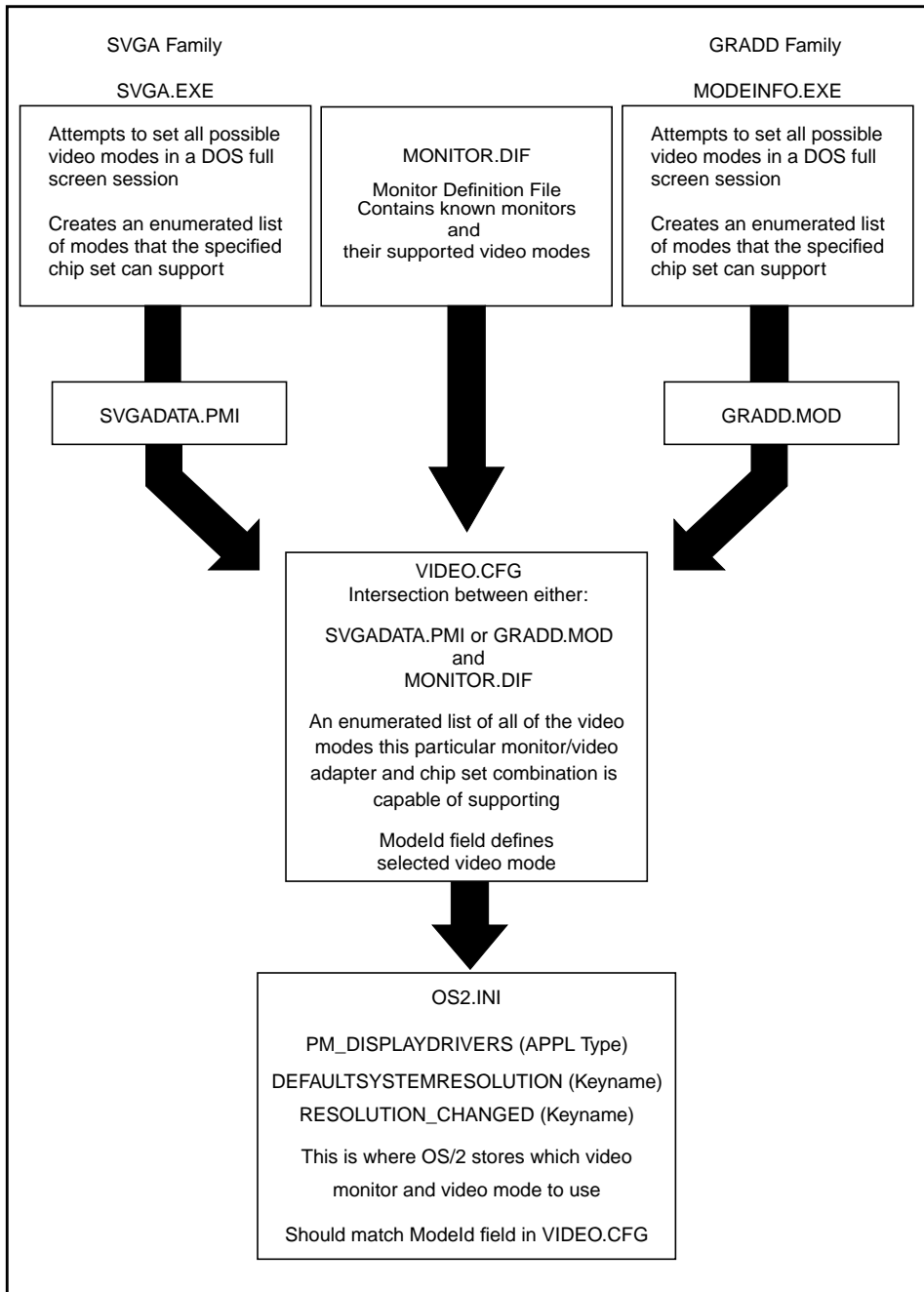


Figure 122. Video Configuration Files



When you boot a client, the client operating system checks the client's OS2.INI file. If the RESOLUTION\_CHANGED keyname is set within the PM\_DISPLAYDRIVERS application type, the operating system checks the *modeld* field in the VIDEO.CFG file to determine the video mode it should use and modifies the DEFAULTSYSTEMRESOLUTION keyname in the OS2.INI file accordingly.

This process *must* occur during the first boot sequence for each client in order to set the DEFAULTSYSTEMRESOLUTION keyname, and therefore the correct video mode, in the client's OS2.INI file. For this reason, it is important to ensure that the RESOLUTION\_CHANGED keyname is present with a value of 1 (TRUE) when you create the OS2.INI or OS2.RCH file for your machine class and that you delete the DEFAULTSYSTEMRESOLUTION keyname if it is present when you create the OS2.INI or OS2.RCH file.

**Note**

Note, however, that you can choose to set the video mode for an entire machine class in the Machine Class - Create notebook. If you do this, the DEFAULTSYSTEMRESOLUTION keyname is set when the Machine Class Create utility creates the OS2.INI file for the machine class, and the RESOLUTION\_CHANGED keyname is *not* set.

You may wish to do this in order to fix the video mode for a specific machine class and not allow different video modes to be specified for individual clients within a machine class.

Before the client operating system updates OS2.INI from the *modeld* field in the VIDEO.CFG file, it performs an integrity check on the VIDEO.CFG file by checking it against the SVGADATA.PMI (or GRADD.MOD) file and MONITOR.DIF file. After determining that the information in VIDEO.CFG is valid, it takes the information from the *modeld* field in VIDEO.CFG and updates the DEFAULTSYSTEMRESOLUTION keyname in OS2.INI.

This is the reason that you should use the MONITOR.DIF file provided by the client operating system, rather than the file provided with a video adapter. If the data included in the VIDEO.CFG file is inconsistent with that provided in the MONITOR.DIF file (which occurs in a number of cases), the consistency check fails. The operating system sets the DEFAULTSYSTEMRESOLUTION keyname to the minimum resolution and color depth (typically 640x480x16 colors) supported by the video adapter and monitor.

Figure 123 on page 344 provides a flow diagram of the consistency checking mechanism that determines the video mode in which a client boots.

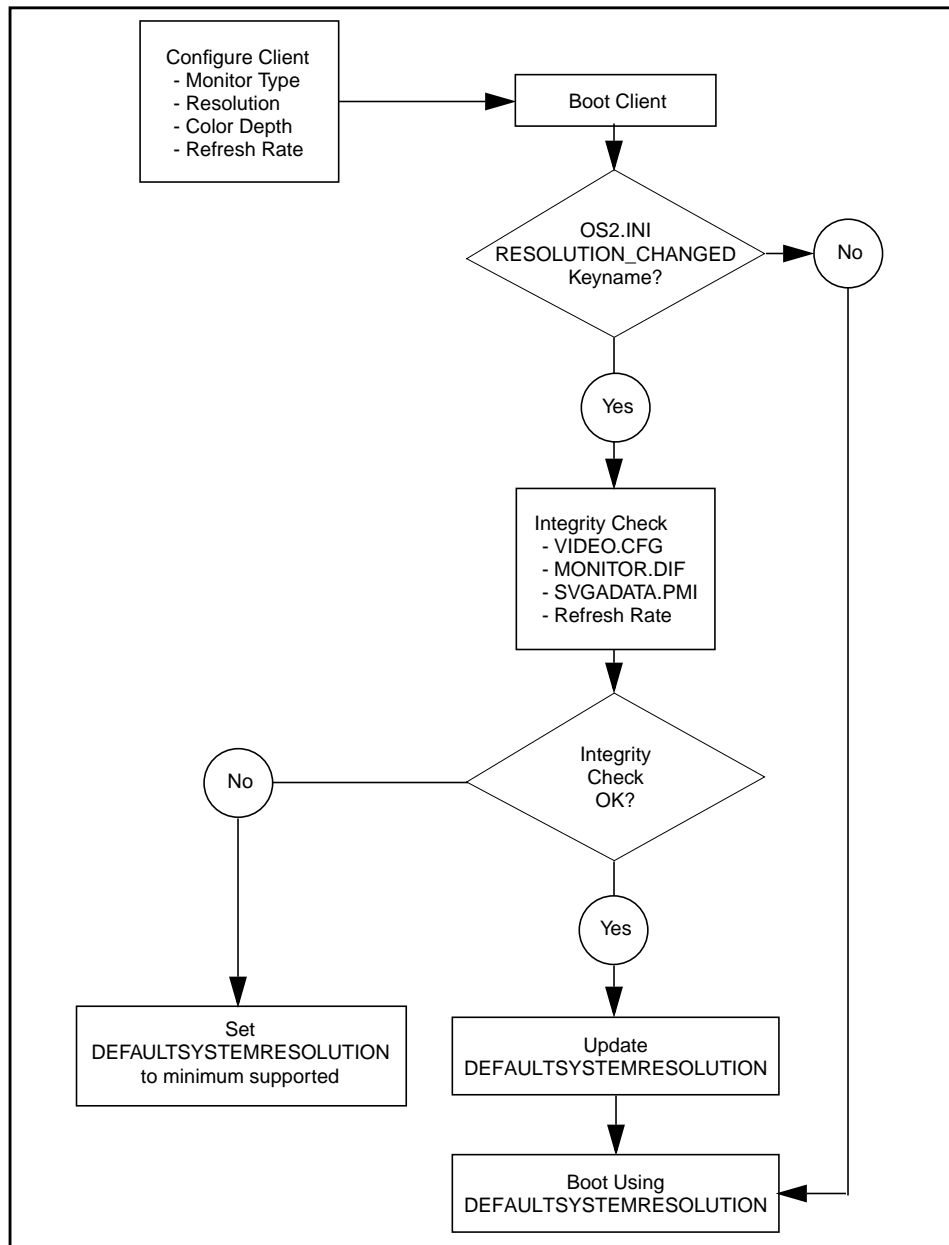


Figure 123. Video Mode Determination Flow

---

## D.3 Editing SVGADATA.PMI

The SVGADATA.PMI file is used in two ways by the WorkSpace On-Demand client operating system:

- It is used to create an updated VIDEO.CFG file to enumerate the video modes supported by the video adapter/monitor combination installed in a client.
- It is used to populate the Video Resolution drop-down list on the Hardware page in the Client Definition notebook.

In certain cases, the SVGADATA.PMI file may contain extraneous entries, or may not contain all of the entries that you require, depending on the level of testing that the manufacturer has carried out. In such cases, you may need to edit the SVGADATA.PMI file to remove extraneous entries or add new ones.

### D.3.1 Deleting Entries

It is possible that certain SVGADATA.PMI files supplied by video adapter manufacturers may not be entirely accurate. For example, the file supplied with the drivers for the CL-GD546X chip set used in our machine classing example includes two video modes that cannot be supported by the video adapter provided on the motherboard of the IBM PC 300GL Model 42U, since they require 4 MB of video RAM while the adapter includes only 2 MB of video RAM. However, the documentation accompanying the drivers states that you should only use the SVGADATA.PMI file shipped with the drivers and should not create the file dynamically using SVGA.EXE.

Including video modes in SVGADATA.PMI that are not supported by the video adapter introduces a risk that you may inadvertently select such a video mode, causing the client's boot process to fail. You may therefore need to edit the SVGADATA.PMI file in order to delete the unsupported video modes and remove this risk.

You should refer to the documentation that accompanies your adapter hardware or the drivers that support it, and compare the video modes described therein with those included in the SVGADATA.PMI file. For example, the README.1ST file on the driver diskette for the CL-GD546X chip set includes a list of the video modes supported by the adapter. Figure 124 on page 346 shows an extract from this file.

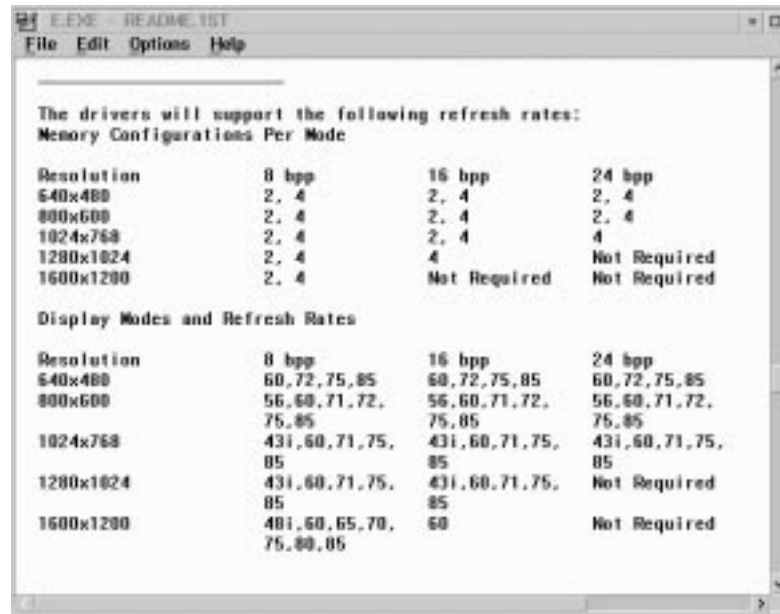


Figure 124. README.1ST File for CL-GD546X Video Chip Set

Comparing this list with the video modes in SVGADATA.PMI reveals two modes that are not supported by the adapter with its 2 MB of video RAM:

- 1024 x 768 x 16 million colors
- 1280 x 1024 x 64K colors

Fortunately, SVGADATA.PMI is an ASCII file that you can edit using a normal text editor. Figure 125 on page 347 shows the entry in the SVGADATA.PMI file that defines the 1280x1024x64K video mode. In our example, we removed this entry along with the entry that defines the 1024 x 768 x 16M video mode.

```
[comment]
    Graphics Mode: 1280 x 1024 x 64k colors.

[ModeInfo]
    ModeAttributes = 0x18
    BytesPerScanLine = 128
    XResolution = 1280
    YResolution = 1024
    TextRows = 48
    BitsPerPixel = 16
    NumberOfPlanes = 1
    PageLength = 1572864
    SaveSize = 1572864
    TotalMemory = 2097152
    InterlaceMode = 0
    BufferAddress = 0x000a0000
    ApertureSize = 0x00010000
    Int10ModeSet = 0x075

[MonitorModeInfo]
    VerticalRefresh = 85

[SetMode]
    call pfnPMISetMode;
```

Figure 125. Editing SVGADATA.PMI - Removing Entries

### D.3.2 Adding Entries

In some cases, you may find that the SVGADATA.PMI file is incomplete and requires additional entries in order to function correctly in conjunction with the Client Definition notebook. For example, the SVGADATA.PMI file provided with the CL-GD546X chip set in our machine classing example did not include the *color* parameter for many of its video modes. This parameter was not required for the file's original purpose since the CL-GD546X determines color depth by examining the *BitsPerPixel* parameter.

However, WorkSpace On-Demand uses the color parameter explicitly when populating the Video resolution drop-down list in the Client Definition notebook. If the color parameter is missing for a particular video mode, that mode appears in the Video resolution drop-down list with a color depth of 0, as shown in Figure 126 on page 348.

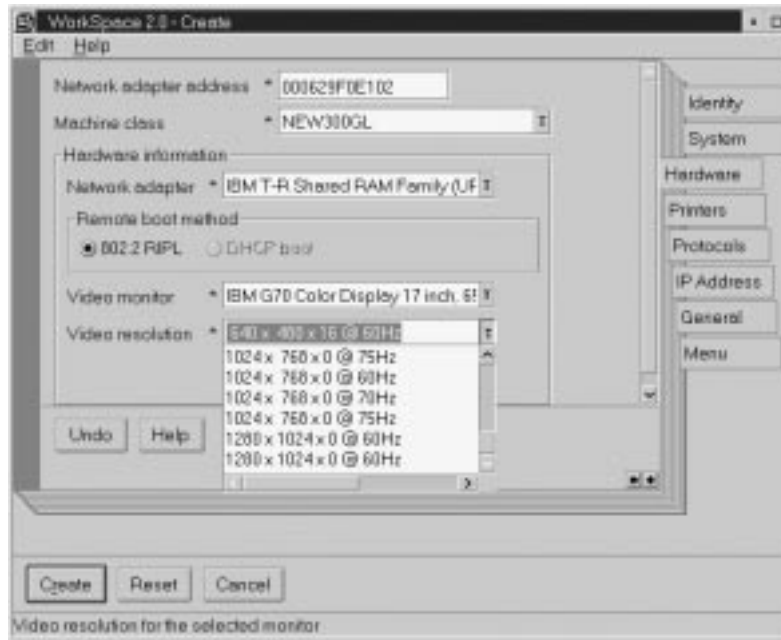


Figure 126. Example of Missing Color Depth

If you choose a video mode with a "zero" color depth, this results in an unsupported video mode in the *modeId* field of the VIDEO.CFG file since the correct video mode cannot be correctly determined. This results in the client booting in its lowest supported video resolution (typically 640x480x16 colors).

For this reason, we recommend that you edit the SVGADATA.PMI file to add the color parameter for all video modes. Figure 123 on page 344 shows a video mode in SVGADATA.PMI with the *color* parameter added.

```
[comment]
    Graphics Mode: 800 x 600 x 64k colors.

[ModeInfo]
    ModeAttributes = 0x18
    BytesPerScanLine = 100
    XResolution = 800
    YResolution = 600
    TextRows = 37
    BitsPerPixel = 16
    NumberOfPlanes = 1
    PageLength = 960000
    SaveSize = 960000
    TotalMemory = 2097152
    InterlaceMode = 0
    BufferAddress = 0x000a0000
    ApertureSize = 0x00010000
    Int10ModeSet = 0x065
    Colors = 65536

[MonitorModeInfo]
    VerticalRefresh = 75

[SetMode]
    call pfnPMISetMode;
```

Figure 127. Editing SVGADATA.PMI to Include Color Depth

Figure 128 on page 350 shows the Video Resolution drop-down list for the same video adapter, after editing the SVGADATA.PMI file to add the *color* parameter to all video modes.

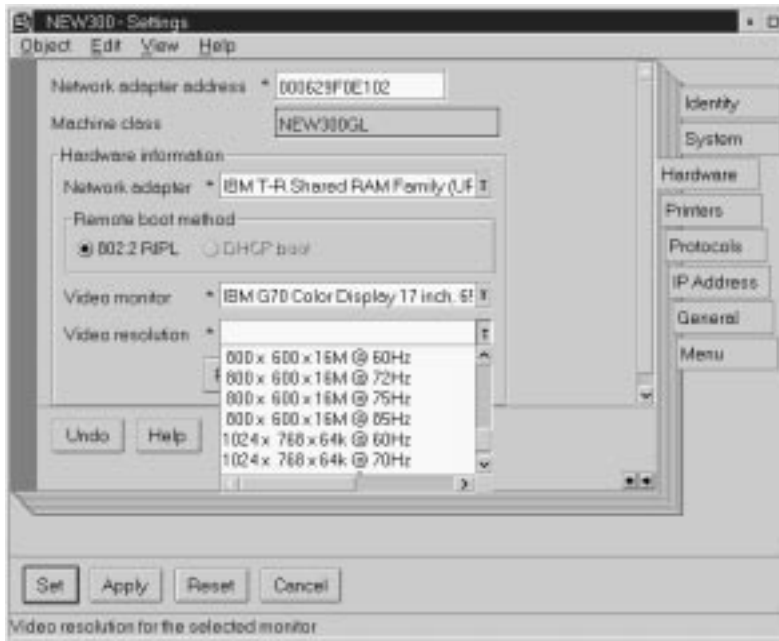


Figure 128. Supported Video Modes



**Note**

As of the publication of this redbook, a problem existed with WorkSpace On-Demand Manager 2.0 and the way that it processed VIDEO.PMI.

Unlike OS/2 V4.00, which is backward compatible with earlier versions, WorkSpace On-Demand Manager 2.0 is designed to work with PMI version 3.0. The SVGADATA.PMI provided with the CL-GD546X software is written in the version 2.15 format (even though it is the latest release available with compilation dates of mid-1998).

This situation results in the remote IPL requester's hardware configuration GUI reverting back to a lower vertical refresh rate than was originally selected when the machine was configured. Testing we've done appears to confirm that the workstation continues to reboot in the desired video mode, including the refresh rate selected, and that only the GUI is configured erroneously.



---

## Appendix E. Tools and Utilities

A number of tools are available to aid in problem-determination on a WorkSpace On-Demand client. This section lists the various tools available and the types of problems they can be used to solve.

---

### E.1 FWATCH

FWATCH is one of the most important problem determination tools for WorkSpace On-Demand. Many problems encountered in WorkSpace On-Demand are directly related to failed file access attempts. When a WSoD client fails to boot or an application fails to start, FWATCH is the best tool to use for problem determination. FWATCH is shipped on the WorkSpace On-Demand CD-ROM, but is not copied over as part of the default installation.

FWATCH logs file open requests and their result codes to a COM port or in a text window. FWATCH has two output formats: with filter and no filter. With filtering (default), the driver only prints the names and result codes of the files that were never opened successfully. This reduces the amount of output. With no filtering, the driver prints the file name and result code of every file open attempt.

The FWATCH output has two output destinations: the COM port or the screen. The FWATCH.SYS driver must be loaded after COM.SYS because it uses the COM driver to set the serial communications line characteristics. The baud rate is 9600, no parity bit, 8 data bits, 1 stop bit. FWATCH initially sends output to the COM port, over a null-modem cable, to a debug terminal. The debug terminal can be an asynchronous dumb terminal or a terminal emulator program, such as LOGICOMM. Text appears on the debug terminal as events happen on the client machine.

FWATCH should only be enabled on a client when problem determination is necessary due to the fact that it reduces the overall performance of the machine.

#### E.1.1 File Watch Utility (FWU.EXE)

After a successful boot, the FWATCH output can be viewed on the screen instead of the debug terminal using a file watch utility named FWU. FWU.EXE is included on the CD accompanying this redbook. FWU.EXE supports these command line parameters:

- STOP

Stop formatting and sending output to any destination.

- ALL

Start sending output, printing every file open attempt.

- START

Start sending output, printing only the "never opened" file attempts.

- COM

Use COM port as output destination.

- READ

Read data and print it on the screen (with default filtering).

- READALL

Read all data and print it on the screen (with no filtering).

The READ and READALL options send data to standard out. You can redirect it to a file, or else use the TEE.EXE utility to send the output both to a file and the screen. For example:

```
FWU READALL | TEE R.OUT
```

will send the output to a file called R.OUT and to the screen.

If you don't care about the data printed during IPL (data which can only be captured on the COM port), then skip the cable and debug terminal and use FWU.EXE after boot up.

### E.1.2 Installation requirements

FWATCH is a security SES driver. Because the OS/2 security kernel allows just one security SES driver at a time, FWATCH cannot coexist with PROTDISK.SYS.

**Note**

PROTDISK.SYS rejects all file opens to local disks. On the other hand, FWATCH.SYS allows file opens to the local disks, which very much changes the client's behavior. This could lead to problems when you remove FWATCH.SYS and return to PROTDISK.SYS.

The following changes must be made to the CONFIG.SYS of the WorkSpace On-Demand client on which you are running FWATCH.

- REM out the PROTDISK.SYS device driver statement.

- Add a DEVICE= statement for FWATCH.SYS somewhere after COM.SYS.
- It is generally better to place FWATCH.SYS near the top of CONFIG.SYS. (It is OK to put COM.SYS and FWATCH.SYS both at the top of CONFIG.SYS.)

### E.1.3 Syntax

The syntax for the FWATCH.SYS statement is as follows:

```
DEVICE=Z:\OS2\FWATCH.SYS [port] [option]

port = 1 for COM1 (default)
port = 2 for COM2

option = A for All files (No filtering)
option = N for No output
```

Filtering is the default. When **A** is present on the DEVICE= statement, no filtering is enabled and every file open attempt is logged along with its result code.

If **N** is present on the DEVICE= statement, all output will be suppressed until the File Watch Utility (FWU.EXE) tells the driver to do otherwise.

Below are some examples of the FWATCH.SYS statement:

```
REM Uses COM2 to log "never opened" files
DEVICE=Z:\OS2\FWATCH.SYS 2

REM Uses COM1 to log all open attempts and result codes
DEVICE=Z:\OS2\FWATCH.SYS A

REM Uses COM1 to log with "never opened" filtering
DEVICE=Z:\OS2\FWATCH.SYS

REM Uses COM2 to log all file open attempts
DEVICE=Z:\OS2\FWATCH.SYS 2A

REM Uses COM2 to log all file open attempts
DEVICE=Z:\OS2\FWATCH.SYS A 2
```

### E.1.4 Deciphering FWATCH output

There are four items on each output line: return code, open flag, open mode, and file name. Below is a sample output line:

```
00000000 00000001 00000020 C:\OS2\SYSTEM\COUNTRY.SYS
```

In this example, the file being accessed is C:\OS2\SYSTEM\COUNTRY.SYS. The result code from the open attempt is 00000000 (no error); the open flag is 00000001 (open the file if it already exists), and the open mode is

00000020 (deny neither read nor write access to other processes/caller requires read-only access).

The return code is the result returned from the DosOpen() API. The open flag is the same as the open flags parameter to the DosOpen() API function, and it represents the desired action the system will take depending on whether the files exists or does not exist. Open mode is the same as the open mode parameter to the DosOpen() API function, and among other things, it represents the file access and sharing modes.

Consult the OS/2 Programmers Toolkit for complete definitions of the DosOpen() API return codes and parameters. For your convenience, the possible values for the return codes and parameters are listed in the tables below.

Table 31. Possible DosOpen() API Return Codes

Return Code (hex)	Description
00000000	No error occurred.
00000002	File not found.
00000003	Path not found.
00000004	Too many open files (no handles left).
00000005	Access denied.
0000000C	Access code is not valid.
0000001A	Unknown media type.
00000020	Sharing violation.
00000024	Sharing buffer overflow.
00000052	Cannot make directory entry.
00000057	Parameter is not valid.
00000063	Device in use.
0000006C	Drive locked by another process.
0000006E	Open/create failed.
00000070	Not enough space on the disk.
000000CE	File name or extension is greater than 8.3 characters.
000000E7	Pipe is busy.

Table 32. Possible DosOpen() API Open Flags

Bits	Description
31 - 8	Reserved, must be 0.
7 - 4	The following flags apply if the file does not exist: 000 - Open an existing file; fail if the file does not exist. 001 - Create the file if the file does not exist.
3 - 0	The following flags apply if the file already exists: 000 - Open the file; fail if the file already exists. 001 - Open the file if it already exists. 010 - Replace the file if it already exists.

Table 33. Possible DosOpen() API Open Mode Values

Bits	Description
31 - 16	Reserved, must be zero.
15	Direct Open flag: 0 - File name represents a file to be opened normally. 1 - File name is a "drive:" (such as C: or A:) and represents a mounted disk or diskette volume to be opened for direct access.
14	Write-through flag: 0 - Writes to the file may go through the file-system driver's cache. 1 - Writes to the file may go through the file-system driver's cache, but the sectors are written before a synchronous write call returns.
13	Fail-errors flag: 0 - Media I/O errors are reported through the system critical-error handler. 1 - Media I/O errors are reported directly to the caller by way of a return code.
12	Cache flag: 0 - Indicates the file system driver should place data from I/O operations into its cache. 1 - Indicates I/O operates to the file need not be done through the file system driver's cache.
11	Reserved; must be 0.

Bits	Description
10 - 8	Reference flag: Contains information about how the application is to get access to the file. 000 - No locality known 001 - Mainly sequential access 010 - Mainly random access 011 - Random with some locality.
7	Inheritance flag: 0 - File handle is inherited by a process created from a call to DosExecPgm. 1 - File handle is private to the current process
6 - 4	Sharing Mode flags: Defines any restrictions to file access placed by the caller on other processes. 001 - Deny read/write access 010 - Deny write access 011 - Deny read access 100 - Deny neither read nor write access (deny none)
3	Reserved; must be 0.
2 - 0	Access Mode flags: Defines the file access required by the caller. 000 - Read-only access 001 - Write-only access 010 - Read/write access

**Notes:**

File sharing requires the cooperation of sharing processes. This cooperation is communicated through sharing and access modes. Any sharing restrictions placed on a file opened by a process are removed when the process closes the file with a DosClose request.

Sharing Mode specifies the type of file access that other processes may have. For example, if other processes can continue to read the file while your process is operating on it, specify Deny Write. The sharing mode prevents other processes from writing to the file, but still allows them to read it.

Access Mode specifies the type of file access (access mode) needed by your process. For example, if your process requires read/write access, and another process has already opened the file with a sharing mode of Deny None, your DosOpen request succeeds. However, if the file is open with a sharing mode of Deny Write, the process is denied access.



If the file is inherited by a child process, all sharing and access restrictions also are inherited. If an open file handle is duplicated by a call to `DosDupHandle`, all sharing and access restrictions also are duplicated.

---

## E.2 Network Analyzers

Network analyzers can be used for low-level debugging of difficult system and application problems on a WorkSpace On-Demand client. Network analyzers are especially useful because they are non-invasive and do not affect the performance of applications running on the client.

With a network analyzer, such as IBM's DatagLANce, you can capture full frames of data on the network and view the data within the frames. DatagLANce provides powerful analysis features, such as protocol filtering, symbolic names, and so forth. Since the majority of operations on a WSoD client involves network traffic, a trace can give you much insight into the system behavior and aid in solving many problems.

DatagLANce can also be used to monitor traffic on the network and take performance measurements such as frame rate, frame utilization, and so forth. This can be very useful in stress testing environments of WorkSpace On-Demand.

---

## E.3 SMBTOOL

SMBTOOL can be used to analyze application behavior on a WorkSpace On-Demand client. To debug an application problem, such as failure to find a necessary file, do the following steps on the WSoD client:

1. Start SMBTOOL on the client.
2. Select **Trace - Redirector - On** from the menu.
3. Press **Enter** and specify a file name with write access.
4. Reproduce the application problem.
5. Select **Trace - Redirector - Off** from the menu.
6. Select **File - Open** from the menu and specify your saved file name.
7. Look in the list box for an error. If you do not see one, you can select **Options - Filter - By Error - Only SMBs with Error** to see a list of only those commands that produced an error.
8. After you have found the SMB with an error response, choose **Options - Filter - By Error - All SMBs** to list all SMBs and find the SMB again.

9. Double-click on the SMB request that precedes the SMB response that contains the error. This will cause the translated data for that request, including file name if appropriate, to be displayed.
10. Double-click on the SMB response that contains the error to display more information about the error.

SMBTOOL can also be used to analyze SMB data in binary network traces produced by network analyzers, such as DatagLANce. Consult the SMBTOOL help for more information about analyzing SMB data.

---

## E.4 SYSLOGPM

SYSLOGPM is a utility included in the WorkSpace On-Demand client image (located in \BMLAN\RPL\BB20.US\OS2) that can be used to diagnose system or network application problems. Using SYSLOGPM, you can display the system error log, the details of the error records, and related diagnostic information that might have been collected for the error, such as dumps and traces.

The *system error log* is created automatically and contains detailed diagnostic information about system and application programs, including the following:

- The state of a program when an error occurred
- Possible causes for the error
- Possible solutions for the error

An *error record* is created by the system when an error in a system or application program triggers a probe in that program. Error records contain detailed information to help you diagnose the error.

A *probe* is a program call to an application program interface (API) that captures information about an error and creates an entry in the error log. A probe can capture the following error information:

- Process status
- Process environment
- Trace information
- User data

This information is saved in a First Failure Support Technology (FFST) dump. To display an FFST dump when one has been created for an error, do the following:

1. Double-click on the record in the SYSLOGPM Summary window.
2. Select **Tools** on the menu bar.
3. Click on **Display Dump File**.

---

## E.5 RDRDebug

RDRDebug is a utility that can be run on a WorkSpace On-Demand client to view the File Index Table (FIT) in memory. RDRDEBUG.EXE is included on the CD-ROM accompanying this redbook.

To view the FIT in memory, RDRDebug must be ran with the `/F` option. By default, the output is directed to the screen; so you may wish to redirect it to a log file.

If RDRDebug is ran with no options, it will output the open file list and information about the network drive and redirector threads.

---

## E.6 WSOD-CV

WorkSpace On-Demand Client View (WSOD-CV) is a tool which can be used to view client environment settings. WSOD-CV.EXE is included on the CD accompanying this redbook.

WSOD-CV can also be used to display the File Index Table (FIT) in memory. An option can be selected to **Execute RDRDebug and Format** (RDRDEBUG.EXE must be located somewhere in the path of the client). This presents the FIT in memory in a more readable format than `RDRDebug /F`. It can be used to examine the FIT and as a tool to explain how FITs work.



## **Appendix F. Special Notices**

This publication is intended to help IBM technical personnel, Business Partners and customers to plan, install and configure a WorkSpace On-Demand implementation. The information in this publication is not intended as the specification of any programming interfaces that are provided by WorkSpace On-Demand or OS/2 Warp Server. See the PUBLICATIONS section of the IBM Programming Announcement for WorkSpace On-Demand 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, 500 Columbus Avenue, Thornwood, NY 10594 USA.

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.

You can reproduce a page in this document as a transparency, if that page has the copyright notice on it. The copyright notice must appear on each page being reproduced.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

AIX	AS/400
AT	DB2
FFST	First Failure Support Technology
DatagLANce	IBM ®
On-Demand Client	On-Demand Server
Netfinity	OS/2
Presentation Manager	RACF
WIN OS/2	Workplace Shell

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc.

Java and HotJava are trademarks of Sun Microsystems, Incorporated.

Microsoft, Windows, Windows NT, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

Pentium, MMX, ProShare, LANDesk, and ActionMedia are trademarks or registered trademarks of Intel Corporation in the U.S. and other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Other company, product, and service names may be trademarks or service marks of others.





---

## Appendix G. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

---

### G.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see "How to Get ITSO Redbooks" on page 369.

- *Inside OS/2 LAN Server 4.0*, SG24-4428
- *Getting to Know OS/2 Warp 4*, SG24-4758
- *Prepare for OS/2 Engineer Certification*, SG24-4869
- *OS/2 Workplace Shell Configuration Techniques*, GG24-4201
- *The Guide to OS/2 Warp Device Drivers*, SG24-4627
- *OS/2 Installation Techniques: The CID Guide*, SG24-4295
- *Inside OS/2 Warp Server, Volume 1, Exploring the Core*, SG24-4602
- *Inside OS/2 Warp Server, Volume 2, System Management*, SG24-4702
- *LAN Problem Determination and Monitoring Using DatagLANce*, SG24-4546
- *Beyond DHCP - IBM's Guide to TCP/IP Network Communications with Dynamic IP*, SG24-5280 (in press)

---

### G.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
Lotus Redbooks Collection	SBOF-6899	SK2T-8039
Tivoli Redbooks Collection	SBOF-6898	SK2T-8044
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RS/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RS/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041

<b>CD-ROM Title</b>	<b>Subscription Number</b>	<b>Collection Kit Number</b>
RS/6000 Redbooks Collection (PDF Format)	SBOF-8700	SK2T-8043
Application Development Redbooks Collection	SBOF-7290	SK2T-8037

---

### **G.3 Other Publications**

These publications are also relevant as further information sources:

- *Getting to Know OS/2 Warp 4*, ISBN 0-13-842147-1
- *Prepare for OS/2 Engineer Certification*, ISBN 9-655-61119-1
- *OS/2 Warp UNLEASHED*, ISBN 0-672-30545-3
- *IBM Token-Ring Network Remote Program Load Version 1.0*, SK2T-0333
- *OS/2 LAN Server Network Administrator Reference Volume 1: Planning, Installation and SConfiguration*, S10H-9680
- *OS/2 LAN Server Network Administrator Reference Volume 3: Network Administrator Tasks*, S10H-9682
- *OS/2 WARP Server Easy Start*, S25H-8003
- *OS/2 WARP Server Up and Running*, S25H-8004
- *OS/2 WARP Server Advanced Easy Start (SMP)*, S28H-0151
- *OS/2 WARP Server Advanced Up and Running (SMP)*, S28H-0152

## How to Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, CD-ROMs, workshops, and residencies. A form for ordering books and CD-ROMs is also provided.

This information was current at the time of publication, but is continually subject to change. The latest information may be found at <http://www.redbooks.ibm.com/>.

---

## How IBM Employees Can Get ITSO Redbooks

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Redbooks Web Site on the World Wide Web**

<http://w3.itso.ibm.com/>

- **PUBORDER** – to order hardcopies in the United States

- **Tools Disks**

To get LIST3820s of redbooks, type one of the following commands:

```
TOOLCAT REDPRINT
TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET SG24xxxx PACKAGE
TOOLS SENDTO CANVM2 TOOLS REDPRINT GET SG24xxxx PACKAGE (Canadian users only)
```

To get BookManager BOOKs of redbooks, type the following command:

```
TOOLCAT REDBOOKS
```

To get lists of redbooks, type the following command:

```
TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
```

To register for information on workshops, residencies, and redbooks, type the following command:

```
TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1998
```

- **REDBOOKS Category on INEWS**

- **Online** – send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL

### Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (<http://www.redbooks.ibm.com/redpieces.html>). 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.

---

## How Customers Can Get ITSO Redbooks

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Online Orders** – send orders to:

In United States	<b>IBMMAIL</b>	<b>Internet</b>
In Canada	usib6fpl at ibmmail	usib6fpl@ibmmail.com
Outside North America	caibmbkz at ibmmail	lmannix@vnet.ibm.com
	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone Orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	(long distance charges apply)
(+45) 4810-1320 - Danish	(+45) 4810-1020 - German
(+45) 4810-1420 - Dutch	(+45) 4810-1620 - Italian
(+45) 4810-1540 - English	(+45) 4810-1270 - Norwegian
(+45) 4810-1670 - Finnish	(+45) 4810-1120 - Spanish
(+45) 4810-1220 - French	(+45) 4810-1170 - Swedish

- **Mail Orders** – send orders to:

IBM Publications	IBM Publications	IBM Direct Services
Publications Customer Support	144-4th Avenue, S.W.	Sortemosevej 21
P.O. Box 29570	Calgary, Alberta T2P 3N5	DK-3450 Allerød
Raleigh, NC 27626-0570	Canada	Denmark
USA		

- **Fax** – send orders to:

United States (toll free)	1-800-445-9269
Canada	1-800-267-4455
Outside North America	(+45) 48 14 2207 (long distance charge)

- **1-800-IBM-4FAX (United States) or (+1) 408 256 5422 (Outside USA)** – ask for:

Index # 4421 Abstracts of new redbooks  
Index # 4422 IBM redbooks  
Index # 4420 Redbooks for last six months

- **On the World Wide Web**

Redbooks Web Site	<a href="http://www.redbooks.ibm.com">http://www.redbooks.ibm.com</a>
IBM Direct Publications Catalog	<a href="http://www.elink.ibm.com/pbl/pbl">http://www.elink.ibm.com/pbl/pbl</a>

### Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (<http://www.redbooks.ibm.com/redpieces.html>). 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.





## 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.** A 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 will boot 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 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, which 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 an 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, which 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, which 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, which 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, which 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>NCP</b>	NetWare Core Protocol
<b>BINL</b>	Boot Information Negotiation Layer	<b>NPT</b>	Non-Programmable Terminal
<b>CID</b>	Configuration, Installation and Distribution	<b>NSM</b>	Network Station Manager
<b>CLI</b>	Command Line Interface	<b>PM</b>	Presentation Manager
<b>CPU</b>	Central Processing Unit	<b>PXE</b>	Preboot Execution Environment
<b>DB2/2</b>	Database 2 for OS/2	<b>RAID</b>	Redundant Array of Independent Disks
<b>DCAF</b>	Distributed Console Access Facility	<b>RIPL</b>	Remote Initial Program Load
<b>DDNS</b>	Dynamic Domain Name Services	<b>RPL</b>	Remote Program Load
<b>DHCP</b>	Dynamic Host Configuration Protocol	<b>SIT</b>	System Integration Testing
<b>DLC</b>	Data Link Control	<b>SMP</b>	Symmetric Multi-Processing
<b>FIT</b>	File Index Table	<b>TCP/IP</b>	Transmission Control Protocol/Internet Protocol
<b>FSD</b>	File System Driver	<b>TFTP</b>	Trivial File Transfer Protocol
<b>GUI</b>	Graphical User Interface	<b>UAA</b>	Universally Administered Address
<b>IBM</b>	International Business Machines Corporation	<b>UAT</b>	User Acceptance Testing
<b>IEEE</b>	Institute of Electrical and Electronics Engineers	<b>UNC</b>	Universal Naming Convention
<b>IS</b>	Information Systems	<b>WPS</b>	Workplace Shell
<b>ITSO</b>	International Technical Support Organization	<b>WSoD</b>	WorkSpace On-Demand
<b>LAN</b>	Local Area Network		
<b>LMU</b>	LAN Management Utilities		
<b>LAA</b>	Local Administered Address		
<b>MIPS</b>	Million Instructions Per Second		



## Index

### A

abbreviations 377  
 access control profiles 90, 93, 95, 173  
 acronyms 377  
 adapter address 141, 146  
 application FIT files 94  
 application management 26  
 application parameters 188, 202  
 application roaming 32  
 application. See network application  
 APPPOST.FIT 95  
 APPPRE.FIT 95  
 automation 315

### B

background bitmap 223  
 BBCOMMON.xxx file 71, 73  
 BCSTNAMES.LST file 74  
 BINL service 68, 75, 142  
 BINLSD.CFG file 77  
 boot block 47, 247  
 boot block definition file 57, 63, 247  
 Boot PROM 45, 47, 68  
 boot server 46, 67  
 bootstrap routine 40, 48  
 bridges 308  
 broadcast filters 310

### C

CACHE386 command 313  
 caching 106  
 client  
   automated definition 151  
   Client Definition notebook 139  
   defining 137, 147  
   deleting 161  
   directory structures 150  
   memory requirements 107  
   NETBEUI 145  
   printer definitions 143  
   querying 163  
   swap file 140  
   TCPBEUI 145  
   using DHCP 147  
   video monitor 142

video resolution 142  
 Client Definition notebook 139  
 client operating system  
   boot drive 140, 304  
   default shell 211  
   micro-FSD 42, 48  
   mini-FSD 42, 50  
   protect mode 41  
   real mode 41  
   redirector 43, 52  
 client.INF file 71, 72  
 client/server ratio 306  
 CNF file 57, 63, 247  
 common descriptor file 73  
 CONFIG.SYS entries 290

### D

data management 27, 30  
 desktop  
   background 223  
   folders 37, 228, 234  
   icon positioning 37, 226  
   Lockup icon 36, 235  
   Logoff icon 235, 238  
   Refresh icon 235  
   Shutdown icon 235, 238  
 DHCP PXE boot mechanism  
   BBCOMMON.xxx 71, 73  
   BINLSD.CFG 77  
   Boot PROM 68  
   boot server 67  
   client.INF 71, 72  
   coexistence with IEEE 8202. RPL 306  
   common descriptor file 73  
   definition 36, 66  
   DHCP.DCFG 75  
   redirector 72  
   requirements 66  
   RFCBCST.LST 74  
   RFCNAMES.LST 74  
 DHCP service 67, 75, 142  
 DHCP.DCFG file 75  
 DIFFTREE utility 176  
 directory structures 79, 150, 169, 271, 283, 291  
 disk caching 106, 118, 313  
 disk formatting 117  
 disk space requirements 104, 107

DOS applications 195, 196  
 DOS PROTOCOL.INI file 244  
 DOS settings 196  
 DSPINSTL.LOG file 267

**E**

environment variables 188, 224  
 Euro-currency support 111

**F**

Feature Installer 120  
 file index table  
   #include statement 96  
   access control profiles 90, 93, 95, 173  
   application files 172, 181  
   application FIT files 94  
   default FIT files 90, 93  
   definition 43, 85  
   FITs in memory 98  
   machine classing 279  
   machine FIT file 87  
   nesting FIT files 96  
   RDRDEBUG utility 99  
   supporting TCPBEUI 53  
   syntax 85  
   user FIT file 91  
   variables 87  
   wildcards 86  
 file index tables  
   APPPOST.FIT 95  
   APPPRE.FIT 95  
 FIT file 172  
 FixPaks 111, 115, 119  
 FM/2 utility 275  
 folders 37, 228, 229, 234  
 FORMAT command 117

**G**

GETRPL utility 128  
 Graphical File Comparison utility 272

**H**

hardware requirements 103  
 HPFS386 file system 106, 118, 313  
 HPFS386.INI file 118

**I**

IBMLAN.INI file 113  
 icon positioning 37, 226  
 IEEE 802.2 RPL boot mechanism  
   boot block 47  
   Boot PROM 47  
   boot server 46  
   coexistence with DHCP PXE 306  
   definition 45, 47  
   mini-FSD 53  
   NETBEUI 48  
   NetBIOS over TCP/IP 53  
   REMOTEBOOT service 46, 55  
   requirements 45  
   RPL.MAP 56, 249  
   TCPBEUI 53  
 INI files 203, 274  
 INIDIFF utility 180, 275  
 INIMERGE utility 180  
 INSTALL command 120, 129  
 installation  
   Administration Client 131  
   attended 119, 131  
   GETRPL utility 128  
   post-installation utility 128  
   selecting components 122  
   unattended 128, 133  
   using CID 130

**J**

Java  
   application support 31  
   defining applets 197  
   defining applications 199  
   version 36

**K**

kernel 40

**L**

local printer 143, 165  
 Locally Administered Address 146  
 Lockup icon 36, 235  
 Logoff icon 235, 238  
 logon bitmap 222  
 logon panel 218

**M**

- machine class
  - creating
    - bus type 287
    - CONFIG.HW file 272
    - CONFIG.SYS entries 290
    - device support 287, 288
    - directory structures 271, 283, 291
    - DSPINSTL.LOG file 267
    - feasibility tests 258
    - file index table 279
    - hardware requirements 258
    - Machine Class Create utility 257, 285
    - methodology 261
    - OS2.INI file 274
    - OS2.RCH file 275
    - OS2SYS.RCH file 279
    - peripherals 288
    - problem determination 292
    - SVGADATA.PMI file 269
    - SYSTEM.INH file 280
    - tools 259
    - video adapter support 289
    - VIDEO.CFG file 270
    - WIN.INH file 281
  - definition 251
  - planning 256
  - predefined classes 38, 108
  - querying 295
  - selecting 141
  - video adapter support 266
  - video resolution 289
  - video type 289
- Machine Class Create notebook 286
- Machine Class Create utility 283
- machine FIT file 87
- MAKEINI utility 236, 274
- MAKERC utility 277
- MCLASS.RSP file 283
- memory requirements
  - client 107
  - server 104
- Micro Channel support 73
- micro-FSD 42, 48
- mini-FSD 42, 50, 53
- MKTMPENV utility 182
- mobile users 29
- MONITOR.DIF file 268
- MPTMPENV utility 175

- MPTS 36
- multi-function server 35
- mutual failover 303

**N**

- NCAPPUTIL utility 212
- NCC\_CHILD\_SETUP environment variable 233
- NCC\_FOLDER environment variable 229, 231
- NCC\_SETUP\_POST environment variable 224, 227
- NDISDD.PRO file 248
- NET commands
  - ACCESS 153
  - APP 193, 317
  - APPPARM 193, 227, 317
  - RIPLMACH 147, 162, 163
  - RIPLMCLAS 295
  - SESS 182, 265
  - SHARE 153
  - START 128
  - STOP 120, 131
  - USE 201
- NETBEUI 48, 145
- NetBIOS over TCP/IP 53
- Netscape Communicator
  - version 36
- NETWKSTA.200 43, 72
- network
  - bridges 308
  - broadcast filters 310
  - existing infrastructure 306
  - protocols 308
  - routers 308
  - segmentation 302, 308
  - topology 307
  - traffic profiles 307
- network adapters
  - adapter address 141, 146
  - Boot PROM 45
  - device drivers 243
  - enabling unsupported adapters 242
  - NDISDD.PRO file 248
  - supported adapters 113, 241
- network application 172
  - access control profiles 173
  - application roaming 206
  - defining 182
  - DIFFTREE utility 176

- directory structures 169
- DOS applications 195, 196
- environment variables 188
- file index table 181
- file types 169
- granting access to users 199
- INI files 203
- INIDIFF utility 180
- INIMERGE utility 180
- installing 174
- MKTMPEENV utility 175, 182
- NCC\_CHILD\_SETUP environment variable 233
- NCC\_FOLDER environment variable 229, 231
- NCC\_SETUP\_POST 224
- NCC\_SETUP\_POST environment variable 227
- network resources 187
- parameters 188
- planning 302
- program mode 187
- reference client 177
- response files 195
- session types 192
- setup strings 224, 229
- structure 167
- types of application 168
- user access 199
- user-specific parameters 202
- WINOS2 applications 196
- work directory 186
- network availability 28
- network computing 32
- network printer 143, 165
- network resources 187

## O

- OS/2 PROTOCOL.INI file 246
- OS/2 Warp Server
  - BINL service 68, 75, 142
  - DHCP service 67, 75, 142
  - directory structures 79
  - FixPaks 110, 111, 115, 119
  - prerequisite features 117
  - PXEPROXY service 68, 75, 142
  - REMOTEBOOT service 46
  - SERVER service 75
  - version 116
- OS2.INI file 274

- OS2.RCH file 275
- OS2KRNL 42
- OS2LDR 42
- OS2SYS.RCH file 279

## P

- performance
  - client 314
  - client/server ratio 306
  - disk subsystem 311, 315
  - memory 311, 314
  - network adapter 314, 315
  - network traffic profiles 307
  - processor 311, 314
- planning 103, 256, 297
- PMLOGON shell
  - background 223
  - command line parameters 213
  - custom logon panel 218
  - customizing 212
  - definition 211
  - desktop background 223
  - logon bitmap 222
  - logon panel 218
  - modifying 212
  - replacing 238
  - user exits 215
- pre-deployment planning 297
- pre-installation planning 103, 256
- printer definitions 143
- printing 165
- problem determination 292
- processor requirements 103, 107
- program mode 187
- PROTDISK.SYS 263
- Protect mode 41
- protect mode redirector 43, 52, 72
- PROTOCOL.INI file
  - DOS 244
  - OS/2 246
  - server 113
- PXEPROXY service 68, 75, 142

## Q

- QUERYCFG.CMD 153



**R**

RAID 313  
 RAM requirements 104, 107  
 RDR\_TEXT utility 99  
 RDRDEBUG utility 99  
 real mode 41  
 redirector 43, 52, 72  
 reference client 175, 177, 258  
 Refresh icon 235  
 registration 127  
 REMOTEBOOT service 46, 55  
 response files 195  
 RFCNAMES.LST file 74  
 RIPL service. See REMOTEBOOT service  
 rollout 297  
 routers 308  
 RPL service. See REMOTEBOOT service  
 RPL.MAP 56, 249  
 RPLBOOT.SYS 48  
 RPLFILES alias 170  
 RUNWORKPLACE statement 211, 213, 238

**S**

security 31  
 segmentation 302, 308  
 server directory structures 79  
 SERVER service 55, 75  
 server/client ratio 306  
 server-managed clients 25  
 session types 192  
 setup strings 224, 229  
 Shutdown icon 235, 238  
 SNGLQUE\$ device driver 294  
 software management 25, 30  
 SVGA utility 269  
 SVGADATA.PMI file 269  
 swap file 140  
 SYSTEM.INH file 280

**T**

TCP/IP  
   configuring 146  
   version 36  
 TCPBEUI 53, 145  
 testing 305  
 TFTP service 68, 75  
 tools 259

**U**

UNIMAIN utility 275  
 Uninstalling 133  
 Universally Administered Address 141  
 user exits 215  
 user FIT file 91

**V**

video adapter support  
   configuring 289  
   directory structures 271, 283  
   DSPINSTL.LOG file 267  
   MONITOR.DIF file 268  
   OS2.INI file 274  
   OS2.RCH file 275  
   OS2SYS.RCH file 279  
   SVGADATA.PMI file 269  
   SYSTEM.INH file 280  
   VIDEO.CFG file 270  
   WIN.INH file 281  
 video monitor 142  
 video resolution 142  
 VIDEO.CFG file 270

**W**

WIN.INH file 281  
 WINOS2 Session 194  
 WINOS2 applications 196  
 WINOS2 settings 196  
 work directory 186  
 WorkSpace On-Demand  
   client operating system. See client operating system  
   components 34, 122  
   deleting 133  
   hardware requirements 103  
   Manager 34, 122  
   pre-deployment planning 297  
   pre-installation planning 103  
   registration 127  
   software prerequisites 109  
   software requirements 117  
 WorkSpace On-Demand client  
   disk space requirements 107  
   processor requirements 107  
   See also client  
 WorkSpace On-Demand server  
   backup 303

disk caching 106, 118, 313  
formatting 117  
memory requirements 104  
network resource requirements 113  
performance 311  
processor requirements 103  
WRKFILES alias 170

## Y

Year 2000 support 111

# ITSO Redbook Evaluation

WorkSpace On-Demand 2.0 Handbook  
SG24-5117-00

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)

**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!)**

---

---

---

---

---

