



Using RDM to Deploy Applications and Windows

A White Paper

October 11, 2005

Notes:

Visit www.ibm.com/pc/safecomputing periodically for the latest information on safe and effective computing.

Warranty Information: For a copy of applicable product warranties, write to: Warranty Information, P.O. Box 12195, RTP, NC 27709, Attn: Dept. JDJA/B203. IBM makes no representation or warranty regarding third-party products or services.

Before using this information and the product it supports, read the general information in "Notices," on page 71.

© Copyright International Business Machines Corporation 2005. All rights reserved.

U.S. Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Table of contents

1. PREFACE	5
1.1 WHO SHOULD READ THIS WHITE PAPER	5
1.2 ASSUMPTIONS	5
1.3 FURTHER REFERENCE	5
1.3.1 <i>Guides</i>	5
1.3.2 <i>White Papers</i>	6
1.3.3 <i>Online help</i>	6
1.3.4 <i>Links</i>	6
2. OVERVIEW	7
2.1 WHAT IS AN APPLICATION?	7
2.1.1 <i>Definition</i>	7
2.1.2 <i>Requirements</i>	7
2.2 <i>WINDOWS NATIVE INSTALL TASKS</i>	7
2.3 <i>WINDOWS CLONE INSTALL TASKS</i>	8
2.3.1 <i>Typical Windows Clone Install procedure</i>	8
2.3.2 <i>Customized Windows Clone Install procedure</i>	8
3. WINDOWS NATIVE INSTALL	9
3.1 INTERNAL TASK LOGIC	9
3.1.1 <i>Overview of task logic</i>	9
3.1.2 <i>Overview of application-install logic</i>	9
3.1.3 <i>Task folder</i>	10
3.1.4 <i>Explore the task logic</i>	11
3.2 APPLICATION IMAGE EXAMPLES	26
3.2.1 <i>Standard applications</i>	26
3.2.2 <i>Irregular application</i>	30
3.2.3 <i>MSI application</i>	33
3.2.4 <i>Collection of applications</i>	35
3.2.5 <i>IBM Director Agent</i>	38
3.3 INSTALLING APPLICATIONS	44
3.3.1 <i>Using RDM's built-in application-install capability</i>	44
3.3.2 <i>Customizing RDM's built-in application-install capability</i>	45
3.3.3 <i>Using RDM's command list</i>	57
3.3.4 <i>Using CMDLINES.TXT</i>	58
3.3.5 <i>Integrating updates or hotfixes into your operating-system image</i>	58
4. WINDOWS CLONE INSTALL	59
4.1 INTERNAL TASK LOGIC	59
4.1.1 <i>Find the task folder</i>	59
4.1.2 <i>Explore the task logic</i>	59
4.2 INSTALLING APPLICATIONS	64
4.2.1 <i>Procedure</i>	64
4.2.2 <i>Install logic</i>	69
5. NOTICES	71
5.1 EDITION NOTICE	71
5.2 TRADEMARKS	71
6. GLOSSARY	73

1. Preface

This White Paper explains how to include application deployment as part of your Windows Native Install tasks and Windows Clone Install tasks. It applies to IBM® Remote Deployment Manager (RDM) 4.20, and later releases.

The procedures described in this paper accomplish their desired functions in a variety of ways. There are alternate techniques available for doing most or all of these functions. The intent is to illustrate various methods as well as to describe a way to implement these particular functions. To use these procedures in your own environment will probably require some extrapolation on your part.

You can use this White Paper to learn how to do the following:

- Understand the internal logic of the *Windows Native Install* task.
- Understand the internal logic of the *Windows Clone Install* task.
- Create *Windows Native Install* application images.
- Customize *Windows Native Install* application images.
- Create a *Windows Native Install* task that can install applications.
- Modify a *Windows Native Install* task so that it installs applications in a nonstandard way.
- Modify a *Windows Clone Install* task so that it installs applications.
- Customize *Windows Native Install* tasks, in general.
- Customize *Windows Clone Install* tasks, in general.

1.1 Who should read this White Paper

This paper is intended to help skilled RDM administrators to create deployment procedures and to understand the concepts involved. To effectively use this paper, you should already have an extensive knowledge of your Network environment, your RDM environment, DOS batch files, and standard installation techniques for Windows applications.

1.2 Assumptions

This paper assumes that you have installed RDM in its default location: C:\Program Files\IBM\RDM. If you have installed RDM in a different location, you will have to make the necessary adjustments to the file paths.

1.3 Further reference

In addition to this paper, there are various other sources of information that you can consult for RDM and for RDM Custom tasks.

1.3.1 Guides

The following product documentation is available for RDM:

- *Remote Deployment Manager 4.20 Users Guide* – The main reference manual for RDM

- *Remote Deployment Manager 4.20 Installation Guide* – Describes the complete installation process of RDM
- *Remote Deployment Manager 4.20 Compatibility and Configuration Guide* – Lists RDM-supported hardware and software

Check the IBM Web site at <http://www-307.ibm.com/pc/support/site.wss/document.do?Indocid=MIGR-50575> to get the current versions of the above documents.

1.3.2 White Papers

The various RDM white papers are available on the IBM Web site at <http://www-307.ibm.com/pc/support/site.wss/document.do?Indocid=MIGR-53487>.

1.3.3 Online help

In general, every window has online help available (except for some message windows or other windows where no help is applicable), either using a **Help** menu or a **Help** button.

1.3.4 Links

The following links are available for further information:

- Support is available for supported systems (IBM and non-IBM) through e-mail or fee-based telephone support. Telephone support is not available in all countries. For more information about the fee-based telephone support, go to <http://www.ibm.com/support> or <http://service.software.ibm.com/supportline.html>. For more information about e-mail support, refer to the RDM home page.

Important: Before using RDM, check the compatibility test results and browse the rest of the RDM Web site for additional information and tips concerning the installation and use of RDM.

2. Overview

This section outlines, at a high level, the procedures and techniques for both kinds of RDM Windows deployment tasks.

2.1 What is an application?

2.1.1 Definition

For this document, an application is defined as being any collection of software that can be installed in Windows. That is, you install Windows first, and then you install the application.

Here are some typical examples:

- Microsoft Office 2003
- Microsoft Visio 2003
- Adobe Acrobat Reader 7.0.0
- WinZip 9.0
- Norton Antivirus 2005
- McAfee VirusScan Enterprise 8.0i
- A Microsoft hot fix for Windows
- A collection of Microsoft hot fixes for Windows

Notice that a single RDM application can actually be a collection of software products that are installed with a batch file.

2.1.2 Requirements

In order for an application to be installable by RDM, its install technique must meet certain requirements:

- Unattended – The application's install program must be able to run with no user interaction. Displaying the application's install windows during its installation process is allowed; that is, the install does not have to be silent.
- Controlled reboots – The application's install program must not reboot the system automatically; it must allow RDM to control the rebooting. If it requires a reboot to complete its installation, all work done after the reboot must happen automatically.
- Configuration – Any system-unique configuration must be doable via an ASCII text file.

2.2 Windows Native Install tasks

RDM contains built-in functionality that can install well behaved applications as part of a *Windows Native Install* task. The basic procedure is this:

1. Design how you will install the application under RDM.
2. Create an RDM *Windows Native Install* image of the application (described in section 3.2 below).
3. Create a *Windows Native Install* task (including its corresponding operating-system image) that uses the application image (described in section 3.3 below).
4. (Optional) Customize the application's install logic, if appropriate, to do system-unique configuration.
5. Test the *Windows Native Install* task to validate that the application installed correctly.

In some cases, it may be necessary to modify this procedure. For example, you might need to install an application at a different point in the process.

2.3 Windows Clone Install tasks

RDM contains no built-in functionality that can install well behaved applications as part of a *Windows Clone Install* task. The typical way to use this task is to use a donor system that contains all of the applications you need.

2.3.1 Typical Windows Clone Install procedure

The typical procedure is this:

1. Install Windows on your donor system.
2. Install applications on your donor system.
3. Test the donor system to validate that the applications are installed correctly.
4. Run Microsoft's SYSPREP.EXE on the donor system.
5. Create an RDM image of the donor system, using the *Get Donor* task.
6. Create a *Windows Clone Install* task that uses the donor image.
7. Test the *Windows Clone Install* task to validate that the applications are installed correctly.

The typical procedure's biggest advantage is that it is the fastest way to deploy Windows and applications. Its main disadvantages are that you may have a large number of large donor images (e.g., for different kinds of system uses) and that these donor images are cumbersome to change (e.g., to use newer versions of applications, to add a Windows service pack, etc.).

2.3.2 Customized Windows Clone Install procedure

It is possible to customize a *Windows Clone Install* task to use similar techniques to those used for a *Windows Native Install* task. That is, you can add or upgrade applications to the task without having to rebuild the donor image. The basic procedure is this:

1. Build and test a *Windows Clone Install* task, using steps 1 through 7 above.
2. Design how you will install the application under RDM.
3. Create an RDM *Windows Native Install* image of the application (described in section 3.2 on page 26).
4. Add logic to the *Windows Clone Install* task's command list to install the application image (described in section 4 on page 59).
5. (Optional) Customize the application's install logic, if appropriate, to do system-unique configuration.
6. Test the *Windows Clone Install* task to validate that the application installed correctly.

3. Windows Native Install

3.1 Internal task logic

To customize application install, and even just to be comfortable creating application images, it will be helpful to understand how RDM does it. In this section, we will explore a typical *Windows Native Install* task that installs Windows Server 2003 Standard plus several applications. Assume that we have completed the procedure outlined in section 2.2 on page 7.

The task logic is encapsulated in several files (that contain lists of commands). These files come from several sources:

- The DOS system environments – These are generated when you install RDM. They do not change, except perhaps when you install an RDM update or a new RDM version.
- The task folder – These are generated while creating the task.
- Generated while running the task.

By understanding the function of each file, you can understand the task logic to a level that will enable you to customize the task. We will describe some of these files in detail, in the sections below.

3.1.1 Overview of task logic

In this section, we describe the *Windows Native Install* task logic at the highest level. Then in later sections, we'll view portions of the logic at a deeper level of detail.

1. Like any RDM task, the *Windows Native Install* task starts with the command list, which contains the overall task logic.
2. The command list runs PRE_INST.BAT, which clears the hard drive and creates partitions.
3. The command list reboots the target system and runs INSTALL.BAT, which installs DOS on the target system.
4. The command list reboots the target system, which automatically boots DOS and runs GO.BAT.
5. GO.BAT runs WINNT.EXE to install Windows. It reboots the system automatically.
6. STARTUP.BAT runs, as a result of being in the startup folder. It installs a Windows service pack, applications, and some device drivers.
7. The system powers off.

3.1.2 Overview of application-install logic

In this section, we summarize how RDM installs applications as part of a Windows Native Install task. This process entails quite a bit of redirection, and it can seem hard to decipher, at first. The steps are the following:

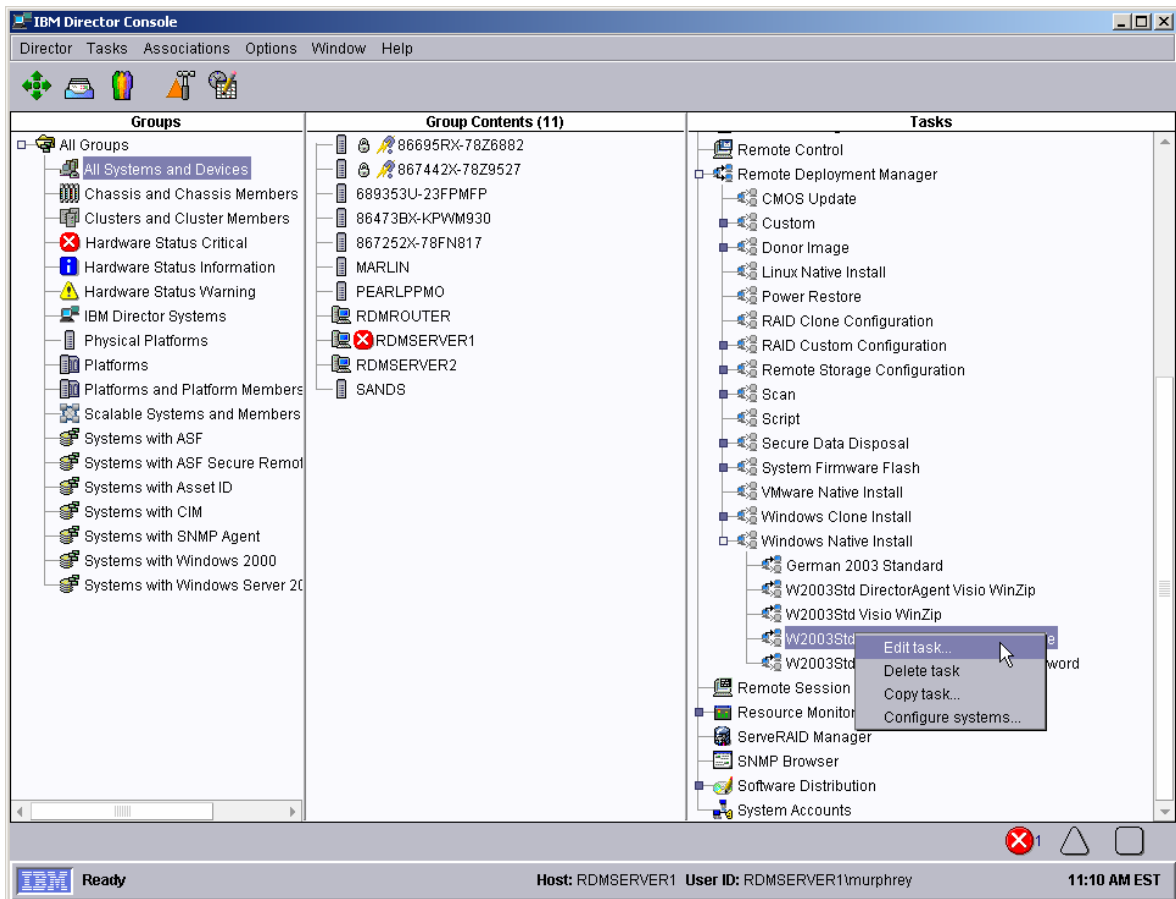
1. COPYAPP.BAT, which is run by INSTALL.BAT under DOS 7.1, downloads files for all the applications.
2. TASKWORK.BAT, which is run by STARTUP.BAT under Windows, initiates the application installs by running several programs.
3. UNZIPAPP.BAT, which is run by TASKWORK.BAT under Windows, unzips all of the applications into their install directories.
4. APPSINST.EXE, which is run by TASKWORK.BAT under Windows, initiates the application install through the use of an INI file.
5. RUNAPPS.INI defines the batch file that will start the set of application installs.

6. APPINST.CMD, which is run by APPSINST.EXE under Windows, runs a batch file APPn.BAT for applications 0, 1, ..., n in order.
7. APPn.BAT, which is run by APPINST.CMD under Windows, installs the nth application.
8. DELAPP.BAT, which is run by TASKWORK.BAT under Windows, deletes the no-longer-needed directories that were used to install the applications.

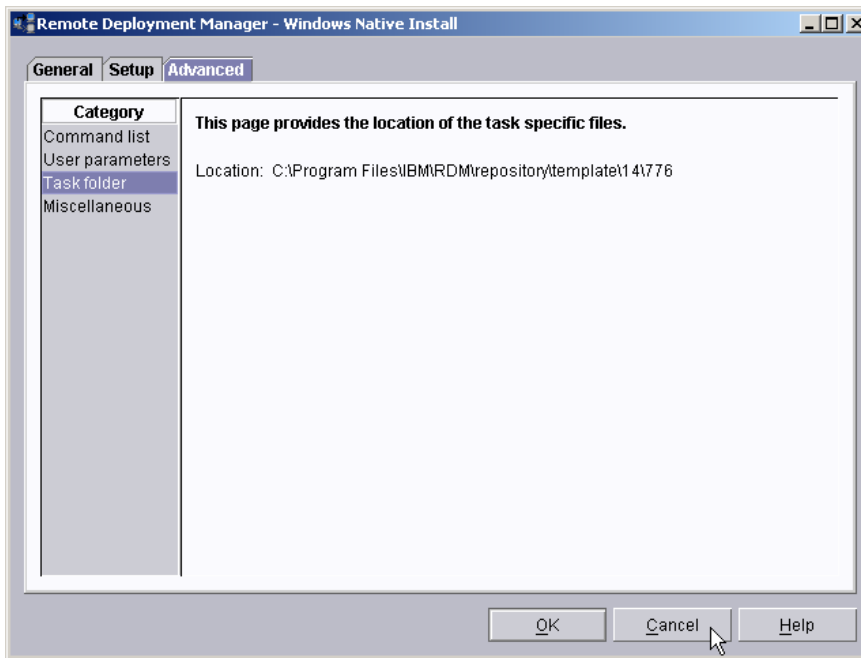
3.1.3 Task folder

Most of the files that you can customize are located in the task folder. Since these folder names are numeric, it is not obvious which folder goes with which task. RDM has a way to find the name of each task's folder. The procedure below shows how to find out the task folder's name.

1. Right click on the *Windows Native Install* task, and select the *Edit task...* menu item.



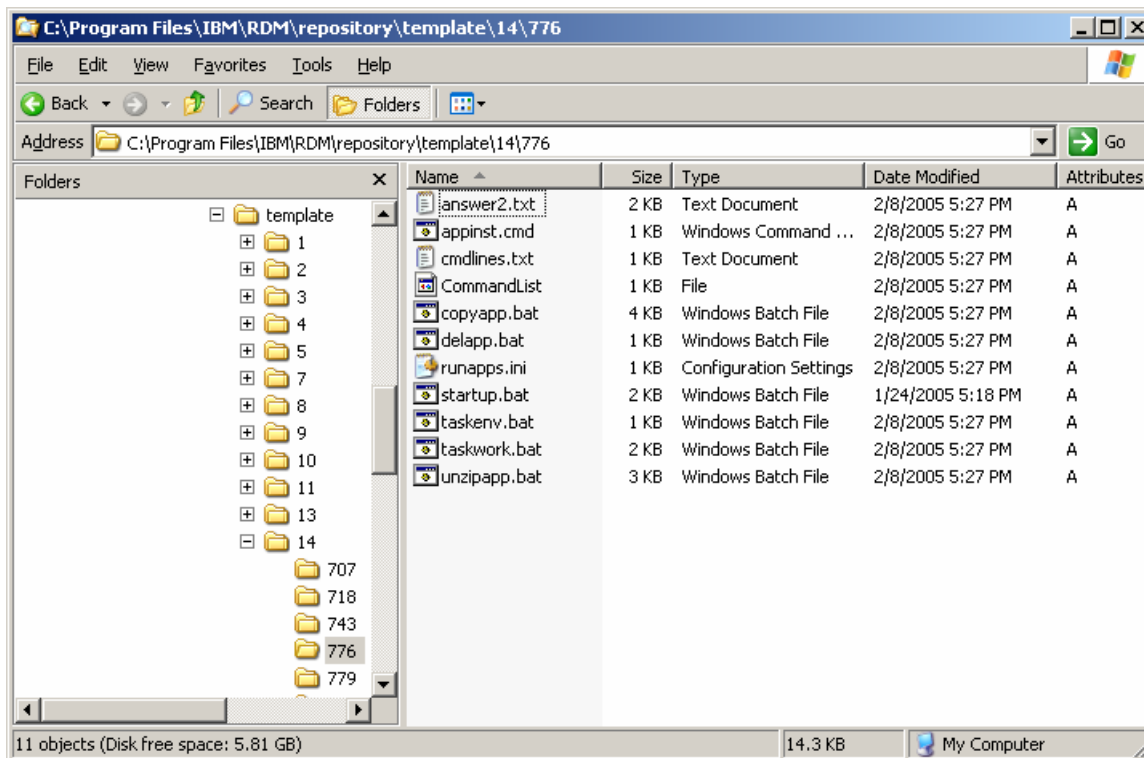
2. Select the *Advanced* page, and then select the *Task folder* category.



3. Select the *Cancel* button to exit without making any changes. This is important, because if you select the *OK* button, it will recreate most of the task's internal files (**and possibly lose some of your customizations**).

3.1.4 Explore the task logic

Open Windows Explorer to view the files in the task folder.



We'll briefly describe and view the contents of each file.

3.1.4.1 CommandList

This file contains the overall task logic. It is the key file for any RDM task. RDM runs the commands in top-to-bottom order, and then (typically) powers off the system. Some commands spawn quite a bit of task logic that is encapsulated in batch files. So you need to understand that logic, in addition to the command-list logic.

You can modify the CommandList file, if needed, using RDM's built-in command-list editor on the *Advanced* page of the task properties window (see section 3.1.1 above to learn how to open this window).

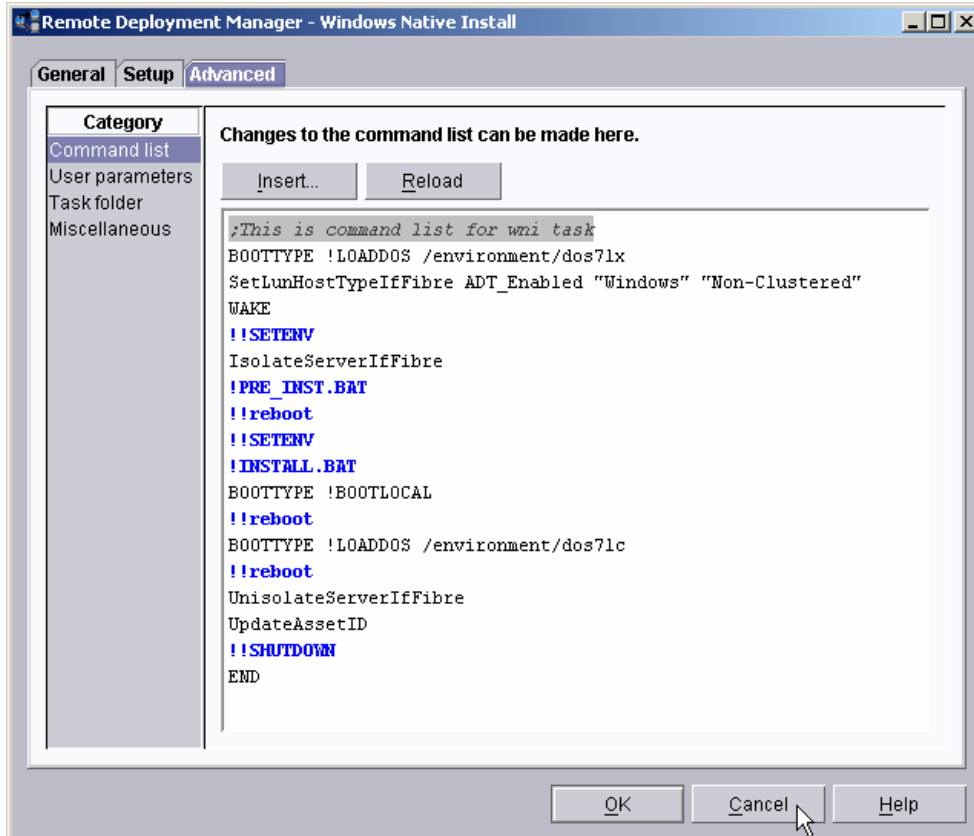
It is instructive to understand how this command list works. The command list syntax allows 4 kinds of (not case-sensitive) commands:

- First character is a semicolon (;) – This is a comment, and it is not part of the task logic. Comments are shown with a gray background in the RDM command-list editor.
- First character is an exclamation point (!) and second character is not an exclamation point – This is a command that is run, as is, on the target system. For example, !PRE_INST.BAT in the command list causes RDM to run the command PRE_INST.BAT on the target system.
- First and second characters are exclamation points (!!)

 - !!SETENV – RDAGENT.EXE initializes the task-specific environment variables on the target system.
 - !!REBOOT – RDAGENT.EXE reboots the target system. It is a warm reboot.
 - !!SHUTDOWN – RDAGENT.EXE powers off the target system.

- Any other first character – This is a command to run a built-in RDM function on the RDM server.

Here is a typical (unmodified) command list for an RDM *Windows Native Install* task.



Now we'll consider each command, in the context of this task.

1. **BOOTTYPE !LOADDOS /environment/dos71x** – The RDM server will force the target system to boot the DOS71X system environment the next time it does a PXE network boot.
2. **SetLunHostTypeIfFibre ADT_Enabled "Windows" "Non-Clustered"** – If Windows is being deployed to a FASTt fibre boot drive and RDM remote storage has been enabled via storage/switch entries in the RDM Network Storage tool, this command will set the host type of the FASTt fibre boot drive to Windows Non-Clustered with Automatic Data Transfer (ADT) enabled.
3. **WAKE** – The RDM server will tell the RDM Deployment Server (D-Server) to power on the target system. The target system will download and run the RDM Bootstrap Loader program, and it will eventually boot the DOS71X system environment (because the BOOTTYPE from step 1 above, which defined DOS71X as the environment, is in effect).
4. **!!SETENV** – The RDAGENT.EXE program will initialize the task's environment variables on the target system. That is, it will run several statements of the form SET NAME=VALUE under DOS 7.1 on the target system.
5. **IsolateServerIfFibre** – If Windows is being deployed to a FASTt fibre boot drive and RDM remote storage has been enabled via storage/switch entries in the RDM Network Storage tool, this command will reconfigure the fibre switch to ensure that only a single path exists between the fibre HBA on the target and the FASTt storage controller.
6. **IPRE_INST.BAT** – The RDAGENT.EXE program will run the PRE_INST.BAT file on the target system. This batch file partitions the hard disk drive, in preparation for the Windows installation.
7. **!!reboot** – This reboots the system. Because it is a warm reboot, it will be the same kind of boot that it did after step 3 above (i.e., a PXE network boot). Since the BOOTTYPE from step 1 above is still in effect, the target system will download and run the RDM Bootstrap Loader program, and it will eventually boot the DOS71X system environment.

It was necessary to reboot the target system in order to make the drive partitioning take effect.

8. **!!SETENV** – The RDAGENT.EXE program will initialize the task's environment variables on the target system. That is, it will run several statements of the form SET NAME=VALUE under DOS 7.1 on the target system.

RDM passes its parameter values to target systems as environment-variable values.

9. **IINSTALL.BAT** – The RDAGENT.EXE program will run the INSTALL.BAT file on the target system. This batch file, which contains or sets up much of the task logic, formats the partitions, installs DOS 7.1 on the boot partition, downloads the image files and programs to that partition, and generally prepares the partition to run the Microsoft program WINNT.EXE (that installs Windows) and then the application install programs.
10. **BOOTTYPE !BOOTLOCAL** – The RDM server will force the target system to boot the local hard drive the next time it does a PXE network boot.
11. **!!reboot** – This reboots the system. Because it is a warm reboot, it will be the same kind of boot that it did in step 3 above (i.e., a PXE network boot). Since the BOOTTYPE from step 10 above is now in effect, the target system will download and run the RDM Bootstrap Loader program, and it will eventually boot the local hard drive (which contains IBM DOS 7.1).

The target system's AUTOEXEC.BAT file first runs GO.BAT, which runs WINNT.EXE to install Windows. The windows installation reboots the system several times, and all of these reboots are out of RDM's control. Eventually the system finishes installing Windows and reboots to its local hard drive (which now contains Windows), and STARTUP.BAT initiates all of the application installs that are part of the task.

It also initiates the running of the RDAGENT.EXE program in a loop. This allows the user to add statements to the command list (after the !!reboot statement) for customization.

12. **BOOTTYPE !LOADDOS /environment/dos71c** – The RDM server will force the target system to boot the DOS71C system environment the next time it does a PXE network boot.
13. **!!reboot** – This reboots the system. Because it is a warm reboot, it will be the same kind of boot that it did in step 3 above (i.e., a PXE network boot). Since the BOOTTYPE from step 12 above is now in effect, the target system will download and run the RDM Bootstrap Loader program, and it will eventually boot the DOS71C system environment.

The purpose of this reboot is so that the target system can do its final handshake with the RDM server.
14. **UnisolateServerIfFibre** – If Windows is being deployed to a FAST fibre boot drive and RDM remote storage has been enabled via storage/switch entries in the RDM Network Storage tool, this command will reconfigure the fibre switch to restore multiple paths between the fibre HBA on the target and the FAST storage controller.
15. **UpdateAssetID** – This causes the RDM Server to initiate an update of 2 fields on the Asset ID EEPROM management chip, for systems (i.e., some IBM NetVista, ThinkCentre, and ThinkPad systems) that have this chip. It writes the first 16 characters of the RDM task name in the IMAGE field, and it writes the current date in the IMAGEDATE field.
16. **!!SHUTDOWN** – This powers off the system.
17. **END** – This tells the RDM server that the task is complete.

3.1.4.2 PRE_INST.BAT

This is the first of two batch files run from the command list. Those files make up all of the encapsulated task logic. The file is part of the DOS71X system environment. It is in RDM's local\env\o\i directory.

```
pre_inst.bat - Notepad
File Edit Format View Help
@ECHO OFF
REM *****
REM * Remote Deployment Manager
REM * (C) Copyright IBM Corp. 1999, 2004 All rights reserved.
REM *
REM *****
REM * PRE_INST.BAT
REM * Partition hard-drives and format partitions
REM *****

set STATUS=
Set TARGET=C:

SET STATUS="GET TASKENV.BAT"
mtftp get %SERVER_IP% template\%TASKTEMPLATEID%\%TASKTOID%\taskenv.bat TASKENV.BAT
if errorlevel 1 goto FAIL

CALL TASKENV.BAT

fdisk32 /status > fdiskout.txt

REM GENERATE RDMFDISK.BAT, RDMFORMAT.BAT AND SETNTFS.BAT
PREPDSKS.EXE

CALL RDMFDISK.BAT
goto END

:FAIL
@ECHO Failed to %STATUS%
call MTFTPRC.BAT
IF %RDRASLEVEL%==0 SET RDRASLEVEL=1
IF %RDSTATUS%="" SET RDSTATUS="RDINST000E Failed to %STATUS%"
goto END

:END
```

PRE_INST.BAT creates and formats the partitions. It does this in 3 steps:

1. It downloads and runs the TASKENV.BAT file, which sets some environment variables.

2. It runs the PREPDSKS.EXE program, which creates batch files that do the disk partitioning.
3. It runs the newly created RDMFDISK.BAT file to do the disk partitioning.

In order to view these batch files, you would have to step through a task execution and break out of the batch file after PREPDSKS.EXE runs. This might help you understand the details of this part of the task logic.

3.1.4.3 INSTALL.BAT

This is the second of two batch files run from the command list. Those files make up all of the high-level encapsulated task logic. The file is part of the DOS71X system environment. It is in RDM's local\env\lo\i directory.

Its logic is summarized as follows:

1. It downloads and runs the TASKENV.BAT file, which sets some environment variables.
2. It runs the DSKTASK.BAT FILE to do the disk formatting.
 - a. It runs the PREPDSKS.EXE program, which creates batch files that do the disk formatting.
 - b. It runs the newly created RDMFORMT.BAT file to do the disk formatting.
3. It copies DOS 7.1 files onto the target system's C: drive, and it creates CONFIG.SYS and AUTOEXEC.BAT files on the C: drive, thereby making it a bootable, DOS 7.1 drive. It also copies several DOS utilities and several RDM batch files.
4. It downloads (using Multicast TFTP) and unzips the Windows image file. This file contains the I386 directory from the Windows CD. It then deletes the zip file.
5. It downloads (using Multicast TFTP) the Windows service pack image file (if one exists). This file contains the I386 directory from the Windows service pack install directory. It then downloads the batch file that will later install the service pack.
6. It downloads (using Multicast TFTP for the larger files) and unzips device driver repositories and various utilities and RDM batch files. It then deletes the zip files.
7. It downloads and calls the COPYAPP.BAT file. This downloads each application image and its corresponding batch file that will later install the application. It also downloads the various files from the task folder that are related to application install.
8. It downloads the wallpaper image (if one exists), and it creates the other files related to wallpaper install.
9. It downloads the ANSWER2.TXT file, and sets up some other files used for RDM processing.
10. It runs RAIDCFG.EXE to obtain information about the existing RAID configuration.
11. It runs SCAN.EXE to obtain information about the current hardware configuration.
12. It modifies the ANSWER2.TXT file with customized hardware information.
13. It creates the textmode drivers specific to the target system's machine type and to the specific Windows version that the task installs.
14. It creates the CLIENT.INI and CLIENT.BAT files, which are used under Windows to set environment variables and parameter values.

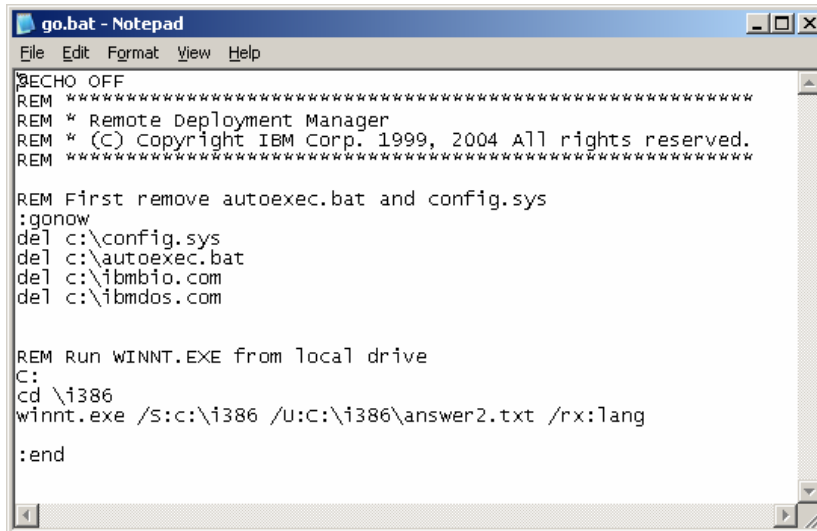
After INSTALL.BAT runs, the command list reboots the target system to its local hard drive. Because of step 3 above, the system boots IBM DOS 7.1, and its AUTOEXEC.BAT file runs GO.BAT, which performs the Windows install using Microsoft's WINNT.EXE program.

3.1.4.4 GO.BAT

This file removes IBM DOS 7.1 from the boot drive, and then it installs Windows. At that point, the Windows installer is controlling the system and its next reboots. Note the reference to ANSWER2.TXT in this file.

The DOS 7.1 AUTOEXEC.BAT file runs GO.BAT.

When GO.BAT completes, the system reboots (to Windows, now) and runs STARTUP.BAT to finish the setup of the hard drive and to install the applications.



```
go.bat - Notepad
File Edit Format View Help
ECHO OFF
REM *****
REM * Remote Deployment Manager
REM * (C) Copyright IBM Corp. 1999, 2004 All rights reserved.
REM *****

REM First remove autoexec.bat and config.sys
:gonow
del c:\config.sys
del c:\autoexec.bat
del c:\ibmbio.com
del c:\ibmdos.com

REM Run WINNT.EXE from local drive
C:
cd \i386
winnt.exe /S:c:\i386 /U:c:\i386\answer2.txt /rx:lang

:end
```

3.1.4.5 ANSWER2.TXT

This is the answer file (often called UNATTEND.TXT) used by Microsoft WINNT.EXE when installing Windows. When you create a *Windows Native Install* task, RDM creates this file in the task folder. If you later edit the task, RDM updates this file based on the changes you made to the task.

You can change this file to control what Windows components will be installed. For example:

- Many users prefer to change the values of XResolution to 1024 and YResolution to 768.
- We will change the value of AutoLogonCount as part of the procedures described in sections below.
- You can add statements that install other Windows components.

If you later edit the task after making such changes, RDM will update this file, but it will attempt to preserve your changes (if possible).



```
answer2.txt - Notepad
File Edit Format View Help

[Unattended]
OemSkipEula=yes
OemPreinstall=YES
TargetPath=*
UnattendSwitch=YES
NowaitAfterGUIMode=1
UnattendMode=FullUnattended
DriverSigningPolicy=Ignore
FileSystem=ConvertNTFS
NowaitAfterTextMode=1

[UserData]
OrgName="%CompanyName%"
ProductKey="%CDKey%"
ComputerName="%ComputerName%"
FullName="%UserName%"

[GuiUnattended]
OEMSkipRegional=1
TimeZone="%Timezone%"
AdminPassword=unity41
AutoLogon=Yes
AutoLogonCount=1
OEMSkipwelcome=1

[RegionalSettings]
Language="%LocaleLanguage%"
LanguageGroup="%LocaleLanguageGroup%"

[Display]
BitsPerPel=16
VRefresh=70
XResolution=800
YResolution=600

[LicenseFilePrintData]
AutoMode=PERSEAT
```

Notice the use of variable names (between two percent signs, such as %CompanyName%). RDM will use the LCCUSTOM.EXE program at the appropriate time in the procedure to replace the variable names with their actual values. LCCUSTOM.EXE gets the values from environment variables.

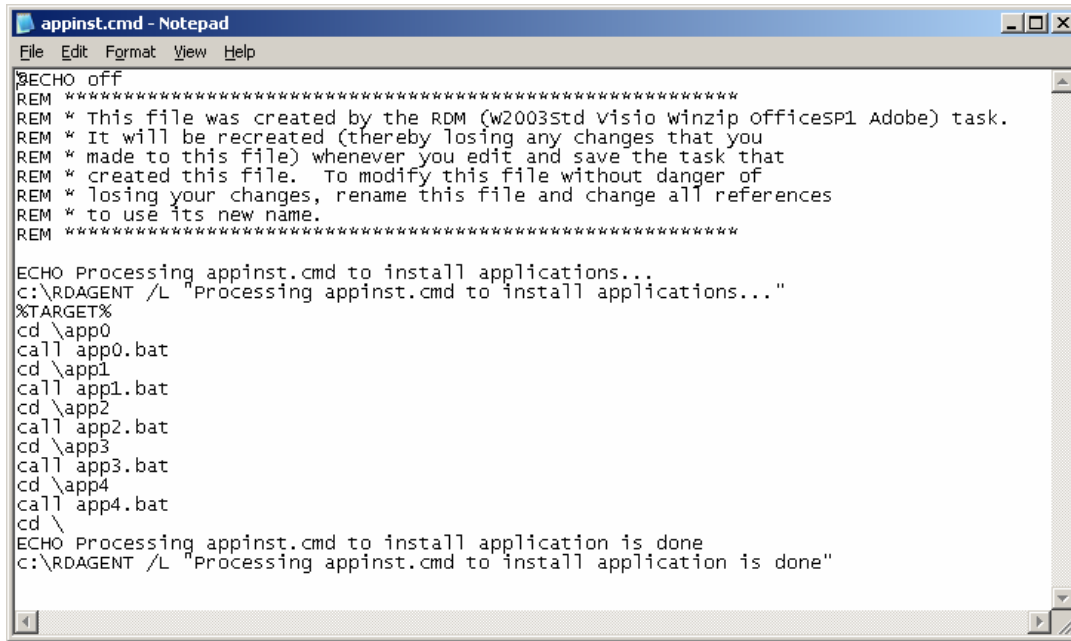
3.1.4.6 APPINST.CMD

This file does the application install. When you create a *Windows Native Install* task, RDM creates this file in the task folder. If you later edit the task, RDM recreates this file based on the current state of the task (i.e., based on which application images are part of the task).

It runs under Windows, and it installs each application in turn. Notice the following features:

- It uses a **C:\RDAGENT /L "<text message>"** command at the beginning and end of the file. This causes RDM to add the text message to the task execution history. It is a good practice to use this kind of command at all interesting points in the RDM processing logic, because it helps in debugging and it helps you follow the task logic in real time as it occurs.
- It installs each application with its own batch file (e.g., app2.bat) and from its own directory (e.g., \app2). That directory will contain the contents of the application image.

Note the use of the %TARGET% environment variable in this file. Its value is the drive letter followed by a colon. It is typically C:, the Windows boot drive.



```

appinst.cmd - Notepad
File Edit Format View Help
@ECHO off
REM *****
REM * This file was created by the RDM (w2003std visio winzip officesp1 Adobe) task.
REM * It will be recreated (thereby losing any changes that you
REM * made to this file) whenever you edit and save the task that
REM * created this file. To modify this file without danger of
REM * losing your changes, rename this file and change all references
REM * to use its new name.
REM *****

ECHO Processing appinst.cmd to install applications...
C:\RDAGENT /L "Processing appinst.cmd to install applications..."
%TARGET%
cd \app0
call app0.bat
cd \app1
call app1.bat
cd \app2
call app2.bat
cd \app3
call app3.bat
cd \app4
call app4.bat
cd \
ECHO Processing appinst.cmd to install application is done
C:\RDAGENT /L "Processing appinst.cmd to install application is done"

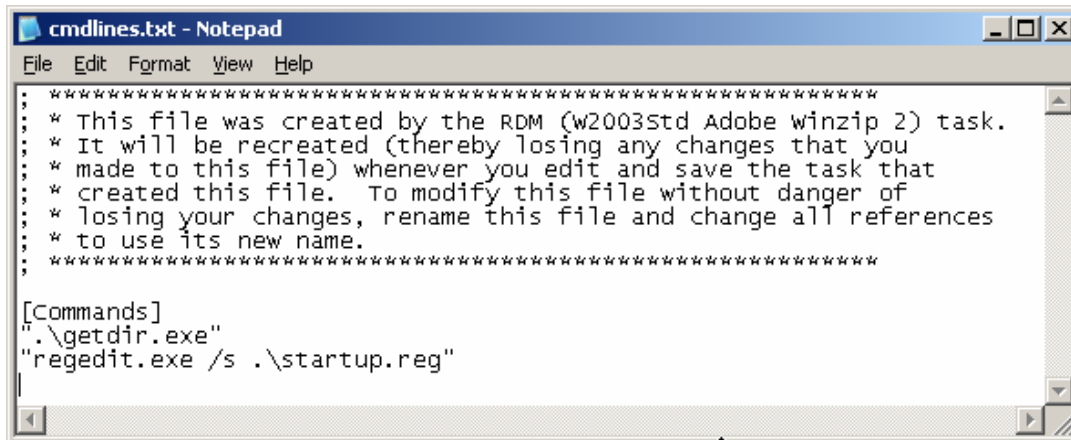
```

3.1.4.7 CMDLINES.TXT

This file updates the Windows registry. RDM creates this file in the task folder. It is a standard part of the Windows installation process. Windows Setup parses CMDLINES.TXT and runs the commands it contains.

In general, any program that can be run at an MS-DOS command prompt while running Windows can be run in CMDLINES.TXT. It runs at the end of the graphical portion of Setup, after the display settings have been set. Windows is running in kernel mode, and networking has been started.

Consult the Microsoft documentation for more information about running programs via CMDLINES.TXT.



```

cmdlines.txt - Notepad
File Edit Format View Help
: *****
: * This file was created by the RDM (w2003std Adobe winzip 2) task.
: * It will be recreated (thereby losing any changes that you
: * made to this file) whenever you edit and save the task that
: * created this file. To modify this file without danger of
: * losing your changes, rename this file and change all references
: * to use its new name.
: *****

[Commands]
".\getdir.exe"
"regedit.exe /s .\startup.reg"

```

3.1.4.8 STARTUP.REG

RDM uses this file to set up the Windows registry so that STARTUP0.BAT will run once, the next time the target system reboots. This file is part of the DOS71X system environment that RDM uses for the *Windows Native Install* task.

A screenshot of a Notepad window titled "startup.reg - Notepad". The window has a menu bar with "File", "Edit", "Format", "View", and "Help". The text inside the window is:

```
REGEDIT4
[HKEY_LOCAL_MACHINE\Software\Microsoft\windows\CurrentVersion\Runonce]
"runstartupbat"="C:\\startup0.bat"
```

3.1.4.9 STARTUP0.BAT

This batch file moves STARTUP.BAT into the Windows StartUp folder. The *Windows Native Install* task creates this file at run time.

As a result, STARTUP.BAT will run every time the target system boots (until such time as RDM deletes it from the StartUp folder).

A screenshot of a Notepad window titled "startup0.bat - Notepad". The window has a menu bar with "File", "Edit", "Format", "View", and "Help". The text inside the window is:

```
chcp 1252
move c:\startup.bat "C:\Documents and Settings\All Users\Start Menu\Programs\Startup"
```

3.1.4.10 COPYAPP.BAT

This batch file downloads the *Windows Native Install* task's built-in application images to the target system's boot drive. The task creates this file in the task folder.

Its functions are:

1. Create the appropriate directory (app0, app1, ...).
2. Download the application image file (the ZIP file).
3. Download its associated batch file (the file that will later be used to install the application).
4. Run LCCUSTOM.EXE on the batch file to substitute appropriate values for its parameters.

Notice some interesting things about this file's logic:

- Each application has its own directory (app0, app1, ...).
- It sets an environment variable, appdir, whose value is that directory. The image's batch file contains a statement that uses this environment variable. It uses LCCUSTOM.EXE to replace that variable with its correct value.

RDM does it this way because an application will not necessarily have the same directory name in every RDM task in which it is used.

Important: You should not modify this file. If you do, the next time you edit the task and click the OK button (to save your task changes), RDM will overwrite this file – thereby losing all of your customizations. If you really have to modify this file, then you need to do this:

1. Make a copy of this file, using a different filename, say MYCOPYAP.BAT.
2. Make your changes in the new copy.
3. Make appropriate changes in the file that calls COPYAPP.BAT so that it calls MYCOPYAP.BAT, instead.

In general, the best practice when you change any RDM file is to make the changes in a differently named copy.

```

copyapp.bat - Notepad
File Edit Format View Help
%ECHO off
REM *****
REM * This file was created by the RDM (w2003std visio winzip officesp1 Adobe) task.
REM * It will be recreated (thereby losing any changes that you
REM * made to this file) whenever you edit and save the task that
REM * created this file. To modify this file without danger of
REM * losing your changes, rename this file and change all references
REM * to use its new name.
REM *****

ECHO Starting copyapp.bat processing to get application install files...
RDAGENT /L "Starting copyapp.bat processing to get application install files..."
SET STATUS="MTFTP application file (053917266703.zip)"
ECHO MTFTP application file (053917266703.zip)
RDAGENT /L "MTFTP application file (053917266703.zip)"
mtftp get %SERVER_IP% image\053917266703.zip %TARGET%\temp\app0 -M
if errorlevel 1 goto FAIL
SET STATUS="MTFTP application launcher file (053917266703.bat)"
if not exist %TARGET%\app0 md %TARGET%\app0
mtftp get %SERVER_IP% image\053917266703.bat %TARGET%\app0\app0.bat
if errorlevel 1 goto FAIL
set appdir=c:\app0
LCCUSTOM c:\app0\app0.bat

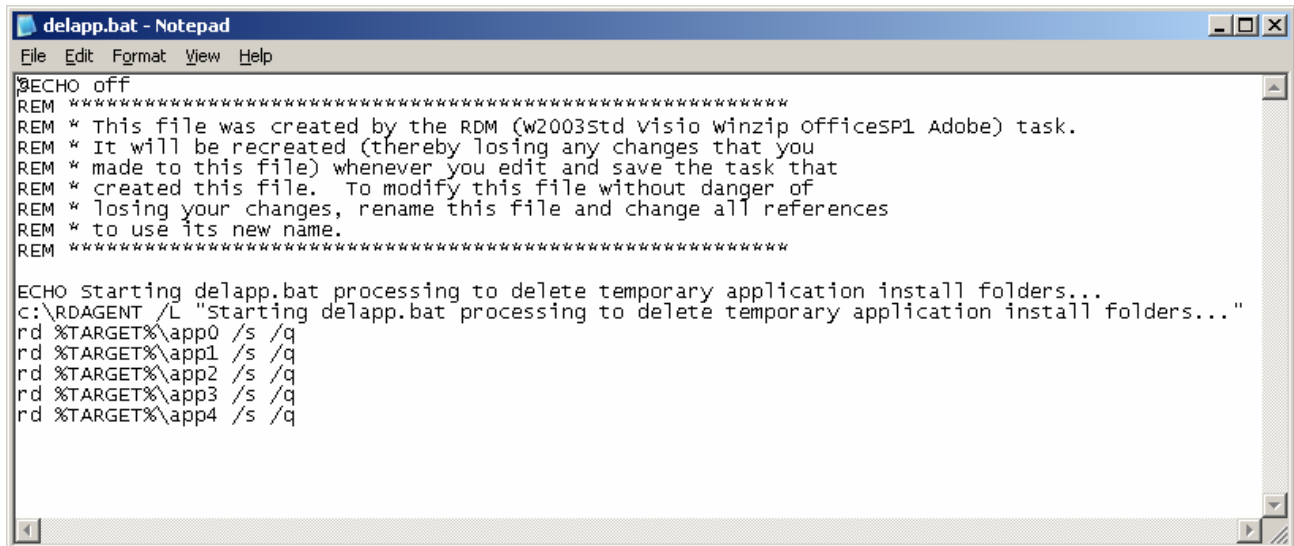
SET STATUS="MTFTP application file (052410107953.zip)"
ECHO MTFTP application file (052410107953.zip)
RDAGENT /L "MTFTP application file (052410107953.zip)"
mtftp get %SERVER_IP% image\052410107953.zip %TARGET%\temp\app1 -M
if errorlevel 1 goto FAIL
SET STATUS="MTFTP application launcher file (052410107953.bat)"
if not exist %TARGET%\app1 md %TARGET%\app1
mtftp get %SERVER_IP% image\052410107953.bat %TARGET%\app1\app1.bat
if errorlevel 1 goto FAIL
set appdir=c:\app1
LCCUSTOM c:\app1\app1.bat

SET STATUS="MTFTP application file (052592324875.zip)"
ECHO MTFTP application file (052592324875.zip)
RDAGENT /L "MTFTP application file (052592324875.zip)"
mtftp get %SERVER_IP% image\052592324875.zip %TARGET%\temp\app2 -M
if errorlevel 1 goto FAIL
SET STATUS="MTFTP application launcher file (052592324875.bat)"
if not exist %TARGET%\app2 md %TARGET%\app2
mtftp get %SERVER_IP% image\052592324875.bat %TARGET%\app2\app2.bat
if errorlevel 1 goto FAIL
set appdir=c:\app2
LCCUSTOM c:\app2\app2.bat

```

3.1.4.11 DELAPP.BAT

This file is used to delete the folders that RDM used for application install. The task creates this file in the task folder.



```

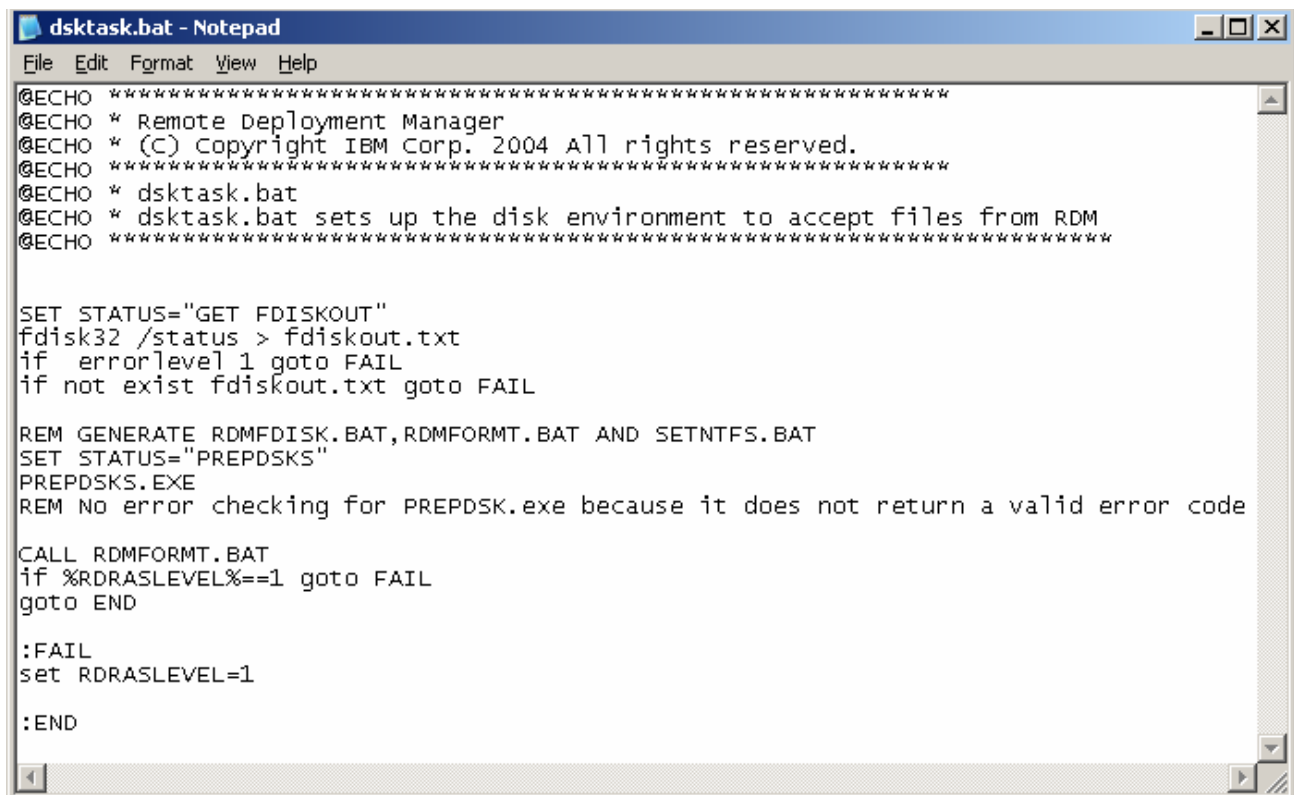
delapp.bat - Notepad
File Edit Format View Help
ECHO off
REM *****
REM * This file was created by the RDM (w2003std visio winzip officesp1 Adobe) task.
REM * It will be recreated (thereby losing any changes that you
REM * made to this file) whenever you edit and save the task that
REM * created this file. To modify this file without danger of
REM * losing your changes, rename this file and change all references
REM * to use its new name.
REM *****

ECHO starting delapp.bat processing to delete temporary application install folders...
c:\RDAGENT /L "starting delapp.bat processing to delete temporary application install folders..."
rd %TARGET%\app0 /s /q
rd %TARGET%\app1 /s /q
rd %TARGET%\app2 /s /q
rd %TARGET%\app3 /s /q
rd %TARGET%\app4 /s /q

```

3.1.4.12 DSKTASK.BAT

This file is used to format the boot partition of the target system's hard drive. The file is part of the DOS71X system environment.



```

dsktask.bat - Notepad
File Edit Format View Help
@ECHO *****
@ECHO * Remote Deployment Manager
@ECHO * (C) Copyright IBM Corp. 2004 All rights reserved.
@ECHO *****
@ECHO * dsktask.bat
@ECHO * dsktask.bat sets up the disk environment to accept files from RDM
@ECHO *****

SET STATUS="GET FDISKOUT"
fdisk32 /status > fdiskout.txt
if errorlevel 1 goto FAIL
if not exist fdiskout.txt goto FAIL

REM GENERATE RDMFDISK.BAT, RDMFORMT.BAT AND SETNTFS.BAT
SET STATUS="PREPDSKS"
PREPDSKS.EXE
REM No error checking for PREPDSK.exe because it does not return a valid error code

CALL RDMFORMT.BAT
if %RDRASLEVEL%==1 goto FAIL
goto END

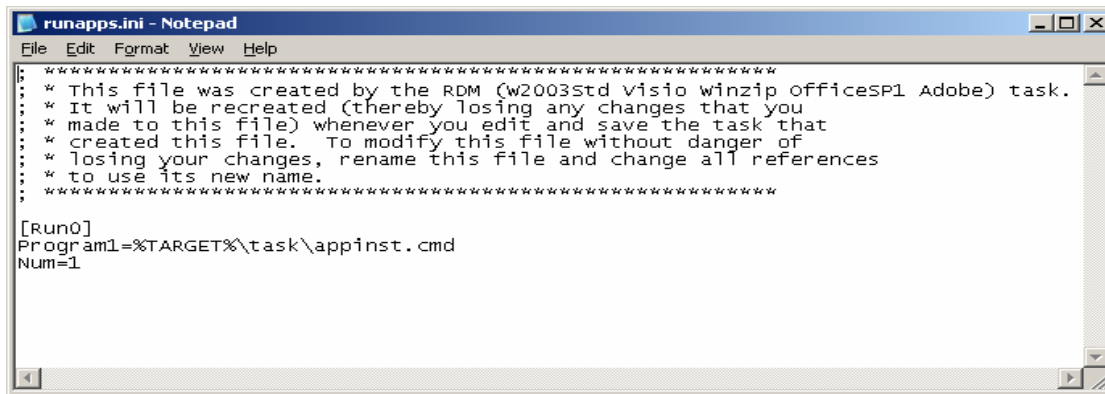
:FAIL
set RDRASLEVEL=1

:END

```

3.1.4.13 RUNAPPS.INI

This file is used to run the batch file APPINST.CMD, which will initiate installation of the applications. The task creates this file in the task folder.



```
runapps.ini - Notepad
File Edit Format View Help
*****
* This file was created by the RDM (w2003std visio winzip officesp1 Adobe) task.
* It will be recreated (thereby losing any changes that you
* made to this file) whenever you edit and save the task that
* created this file. To modify this file without danger of
* losing your changes, rename this file and change all references
* to use its new name.
*****
[Run0]
Program1=%TARGET%\task\appinst.cmd
Num=1
```

3.1.4.14 STARTUP.BAT

This file runs right after the Windows install completes. After Windows install, the system reboots. RDM has previously set the system up so that it runs STARTUP.BAT automatically. STARTUP.BAT does a lot of post-install setup and configuration, such as:

- Converts the file system to NTFS.
- Installs applications and RSA or ASR drivers (via file TASKWORK.BAT, below).
- Installs some other device drivers, such as ASM (via file POSTINST.BAT).
- Does some cleanup work.
- Starts running RDAGENT.EXE in a loop (via file RDMAGENT.BAT).

STARTUP.BAT can run multiple times, depending on how many controlled reboots the task does. You can see how many such reboots will occur by viewing the ANSWER2.TXT file. The following statements (in the ANSWER2.TXT file) control the number of reboots in our sample task:

```
AutoLogon=Yes
AutoLogonCount=1
```

If we change this task to install a Windows service-pack image, ANSWER2.TXT would contain the following statements:

```
AutoLogon=Yes
AutoLogonCount=2
```

```

startup.bat - Notepad
File Edit Format View Help
@ECHO off
REM *****
REM * This file is used immediately after windows completed the unattended
REM * OS install. The task folder contains this file which is copied from the
REM * template folder when the task is created. Subsequent editing of the
REM * task would not overwrite this file.
REM *****

ECHO Start processing startup.bat...
C:\RDAGENT /L "RDAGEN000I Start processing startup.bat..."
TITLE "RDM STARTUP.BAT - processing, please wait..."
C:\sleep.exe 40
call c:\setpath.bat

if exist c:\donentfs.txt goto SKIP_NTFS
ECHO Converting drive(s) to NTFS...
C:\RDAGENT /L "RDAGEN000I Converting drive(s) to NTFS..."
call c:\rdmbin\setNTFS.bat
ECHO y > c:\donentfs.txt

:SKIP_NTFS

REM *****
REM * This block of instruction should not be moved or removed
call %TARGET%\taskwork.bat
if %taskRC%==1 goto FAIL
REM *****

ECHO Running post-install batch file that might install driver e.g. ASM...
C:\RDAGENT /L "RDAGEN000I Running post-install batch file that might install driver e.g. ASM..."
call c:\postinst.bat 1
ECHO Cleaning files and folder
C:\RDAGENT /L "RDAGEN000I Cleaning files and folder"
del c:\postinst.bat
c:\cleanup.exe
ECHO startup.bat processing is done
C:\RDAGENT /L "RDAGEN000I startup.bat processing is done"
ECHO Calling RDMAGENT.bat for possible custom work
C:\RDAGENT /L "RDAGEN000I Calling RDMAGENT.bat for possible custom work"
call c:\RDMAGENT.bat
goto END

:FAIL
C:\RDAGENT /L "FATAL ERROR ENCOUNTERED - HALT PROCESSING OF STARTUP.BAT"
SET RDRASLEVEL = 1
SET RDSTATUS = "RDAGEN099E windows startup encountered fatal error."
C:\RDAGENT.EXE /fs

:END

```

3.1.4.15 TASKENV.BAT

This file sets some task-related environment variables and it downloads the TASKWORK.BAT file. The task creates this file in the task folder.

```

taskenv.bat - Notepad
File Edit Format View Help
@ECHO off
REM *****
REM * This file was created by the RDM (w2003Std visio winzip officesp1 Adobe) task.
REM * It will be recreated (thereby losing any changes that you
REM * made to this file) whenever you edit and save the task that
REM * created this file. To modify this file without danger of
REM * losing your changes, rename this file and change all references
REM * to use its new name.
REM *****

SET STATUS="MTFTP FILE %TASKTEMPLATEID%\%TASKTOID%\taskwork.bat"
mtftp get %SERVER_IP% template%\TASKTEMPLATEID%\%TASKTOID%\taskwork.bat taskwork.bat
if errorlevel 1 goto FAIL
set Disk_1=1
set InstallApp=1
set LocalGroup=Users
set PnPDriver=dnetpnp.zip
set TxtDriver=dnettxtm.zip
goto END

:FAIL
ECHO Processing of taskenv.bat failed
RDAGENT /L "Processing of taskenv.bat failed"
SET RDRASLEVEL=1
goto END

:End

```


3.1.4.16 TASKWORK.BAT

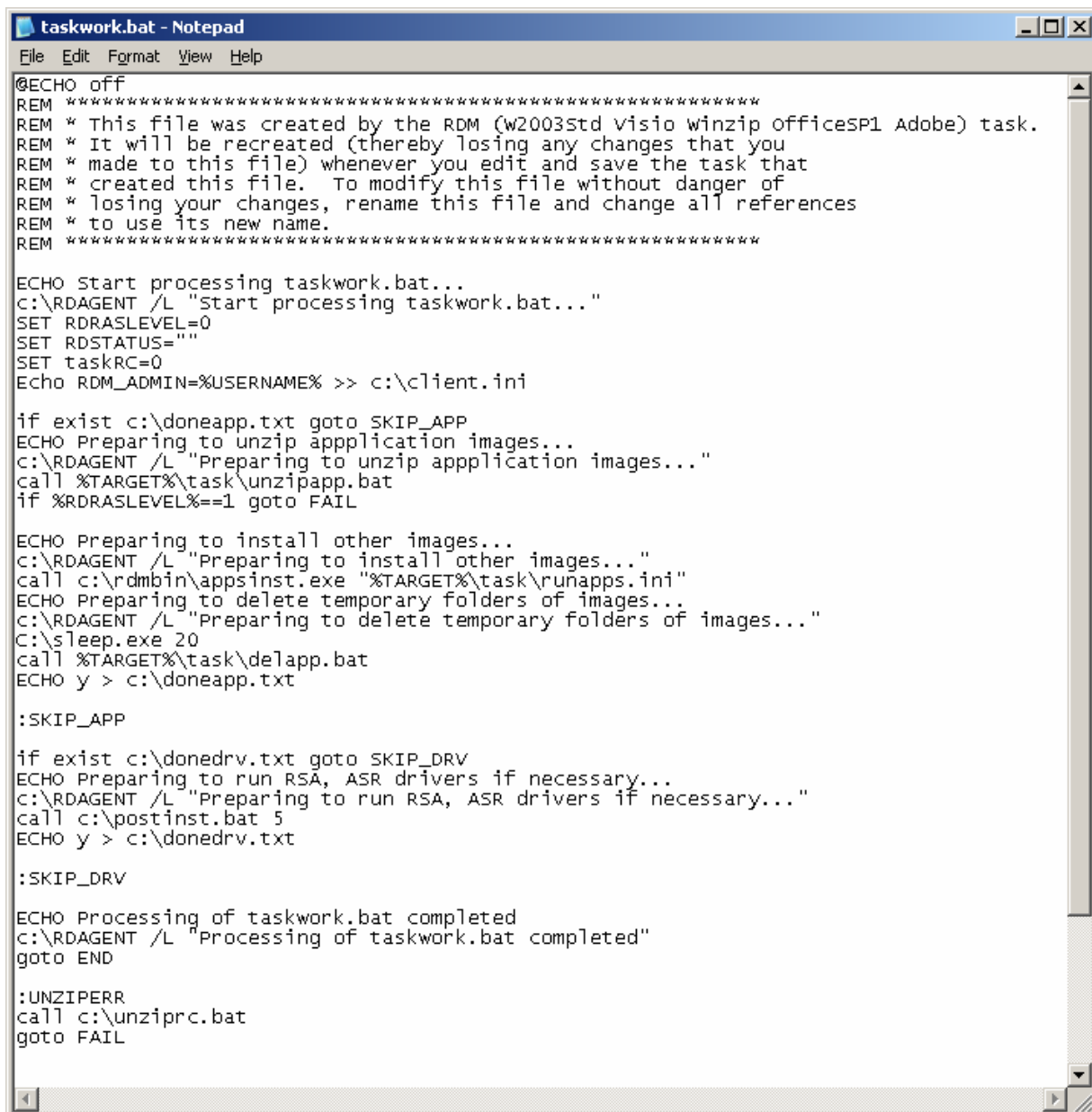
This file unzips the application images, installs the applications, cleans up the temporary folders used during application install, and installs RSA or ASR drivers, if necessary.

Notice that %USERNAME% is not getting the same value as the one we substituted in the ANSWER2.TXT by running LCCUSTOM.EXE in the DOS phase of the task. This time, the value is coming from the Windows environment (i.e., not from RDM).

This file will be called as many times as STARTUP.BAT runs. Compare the 2 versions of TASKWORK.BAT, below. The following file is from the Windows Native Install task that this document uses, which also installs some applications. It has 2 significant blocks of statements:

1. Unzip the applications (UNZIPAPP.BAT), install the applications (APPSINST.EXE), delete the install directories (DELAPP.BAT), and create the semaphore file.
2. Install device drivers (POSTINST.BAT), and create the semaphore file.

Each block of statements only runs if its semaphore file is not present.



```
taskwork.bat - Notepad
File Edit Format View Help
@ECHO off
REM *****
REM * This file was created by the RDM (W2003Std Visio winzip OfficesP1 Adobe) task.
REM * It will be recreated (thereby losing any changes that you
REM * made to this file) whenever you edit and save the task that
REM * created this file. To modify this file without danger of
REM * losing your changes, rename this file and change all references
REM * to use its new name.
REM *****

ECHO Start processing taskwork.bat...
c:\RDAGENT /L "start processing taskwork.bat..."
SET RDRASLEVEL=0
SET RDSTATUS=""
SET taskRC=0
ECHO RDM_ADMIN=%USERNAME% >> c:\client.ini

if exist c:\doneapp.txt goto SKIP_APP
ECHO Preparing to unzip application images...
c:\RDAGENT /L "Preparing to unzip application images..."
call %TARGET%\task\unzipapp.bat
if %RDRASLEVEL%==1 goto FAIL

ECHO Preparing to install other images...
c:\RDAGENT /L "Preparing to install other images..."
call c:\rdmbin\appsinst.exe "%TARGET%\task\runapps.ini"
ECHO Preparing to delete temporary folders of images...
c:\RDAGENT /L "Preparing to delete temporary folders of images..."
C:\sleep.exe 20
call %TARGET%\task\delapp.bat
ECHO y > c:\doneapp.txt

:SKIP_APP

if exist c:\donedrv.txt goto SKIP_DRV
ECHO Preparing to run RSA, ASR drivers if necessary...
c:\RDAGENT /L "Preparing to run RSA, ASR drivers if necessary..."
call c:\postinst.bat 5
ECHO y > c:\donedrv.txt

:SKIP_DRV

ECHO Processing of taskwork.bat completed
c:\RDAGENT /L "Processing of taskwork.bat completed"
goto END

:UNZIPERR
call c:\unziprc.bat
goto FAIL
```


Contrast the above with the following file, which is from a Windows Native Install task that also installs a Windows service pack and some applications. It has 3 significant blocks of statements:

1. Unzip the Windows service pack (UNZIP.EXE), Install the Windows service pack (LAUNCHSP.BAT), create the semaphore file, and reboot (LCREBOOT.EXE), and create the semaphore file.
2. Unzip the applications (UNZIPAPP.BAT, install the applications (APPSINST.EXE), delete the install directories (DELAPP.BAT), and create the semaphore file.
3. Install device drivers (POSTINST.BAT), and create the semaphore file

The first block of statements reboots the system. When the system then boots, it runs TASKWORK.BAT again, and the batch-file logic uses the semaphore file to skip the first block of statements.



```
taskwork.bat - Notepad
File Edit Format View Help
ECHO Start processing taskwork.bat...
c:\RDAGENT /L "Start processing taskwork.bat..."
SET RDRASLEVEL=0
SET RDSTATUS=""
SET taskRC=0
Echo RDM_ADMIN=%USERNAME% >> c:\client.ini

if exist c:\donesp.txt goto SKIP_SP
ECHO unzip service pack...
c:\RDAGENT /L "Unzip service pack..."
if not exist %TARGET%\sp md %TARGET%\sp
c:\unzip -o %TARGET%\temp\sp.zip -d %TARGET%\sp
if errorlevel 3 goto UNZIPERR
if errorlevel 1 call c:\unzipwrn.bat
if errorlevel 0 set RDSTATUS="unzip of image (%TARGET%\temp\sp.zip) successful"
del %TARGET%\temp\sp.zip
ECHO Preparing to install service pack...
c:\RDAGENT /L "Preparing to install service pack..."
call %TARGET%\sp\launchsp.bat
ECHO y > c:\donesp.txt
c:\rdmbin\lcreboot.exe
C:\sleep.exe 60
goto END

:SKIP_SP

if exist c:\doneapp.txt goto SKIP_APP
ECHO Preparing to unzip application images...
c:\RDAGENT /L "Preparing to unzip application images..."
call %TARGET%\task\unzipapp.bat
if %RDRASLEVEL%==1 goto FAIL

ECHO Preparing to install other images...
c:\RDAGENT /L "Preparing to install other images..."
call c:\rdmbin\appsinst.exe "%TARGET%\task\runapps.ini"
ECHO Preparing to delete temporary folders of images...
c:\RDAGENT /L "Preparing to delete temporary folders of images..."
C:\sleep.exe 20
call %TARGET%\task\delapp.bat
ECHO y > c:\doneapp.txt

:SKIP_APP

if exist c:\donedrv.txt goto SKIP_DRV
ECHO Preparing to run RSA, ASR drivers if necessary...
c:\RDAGENT /L "Preparing to run RSA, ASR drivers if necessary..."
call c:\postinst.bat 4
ECHO y > c:\donedrv.txt

:SKIP_DRV

ECHO Processing of taskwork.bat completed
c:\RDAGENT /L "Processing of taskwork.bat completed"
```

3.1.4.17 UNZIPAPP.BAT

This file unzips the application image file for each application. The task creates this file in the task folder.

```

unzipapp.bat - Notepad
File Edit Format View Help
ECHO off
REM *****
REM * This file was created by the RDM (w2003Std Visio winzip OfficesP1 Adobe) task.
REM * It will be recreated (thereby losing any changes that you
REM * made to this file) whenever you edit and save the task that
REM * created this file. To modify this file without danger of
REM * losing your changes, rename this file and change all references
REM * to use its new name.
REM *****

ECHO Starting unzipapp.bat processing to unzip application images...
c:\RDAGENT /L "Starting unzipapp.bat processing to unzip application images..."
if not exist %TARGET%\app0 md %TARGET%\app0
%TARGET%\unzip -o %TARGET%\temp\app0 -d %TARGET%\app0
if errorlevel 3 goto UNZIPERR
if errorlevel 1 call c:\unzipwrn.bat
if errorlevel 0 set RDSTATUS="unzip of image (053917266703.zip) successful"
del %TARGET%\temp\app0

if not exist %TARGET%\app1 md %TARGET%\app1
%TARGET%\unzip -o %TARGET%\temp\app1 -d %TARGET%\app1
if errorlevel 3 goto UNZIPERR
if errorlevel 1 call c:\unzipwrn.bat
if errorlevel 0 set RDSTATUS="unzip of image (052410107953.zip) successful"
del %TARGET%\temp\app1

if not exist %TARGET%\app2 md %TARGET%\app2
%TARGET%\unzip -o %TARGET%\temp\app2 -d %TARGET%\app2
if errorlevel 3 goto UNZIPERR
if errorlevel 1 call c:\unzipwrn.bat
if errorlevel 0 set RDSTATUS="unzip of image (052592324875.zip) successful"
del %TARGET%\temp\app2

if not exist %TARGET%\app3 md %TARGET%\app3
%TARGET%\unzip -o %TARGET%\temp\app3 -d %TARGET%\app3
if errorlevel 3 goto UNZIPERR
if errorlevel 1 call c:\unzipwrn.bat
if errorlevel 0 set RDSTATUS="unzip of image (0428919835750.zip) successful"
del %TARGET%\temp\app3

if not exist %TARGET%\app4 md %TARGET%\app4
%TARGET%\unzip -o %TARGET%\temp\app4 -d %TARGET%\app4
if errorlevel 3 goto UNZIPERR
if errorlevel 1 call c:\unzipwrn.bat
if errorlevel 0 set RDSTATUS="unzip of image (04275144422531.zip) successful"
del %TARGET%\temp\app4

```

3.2 Application image examples

This document will informally categorize applications as to their unattended-install properties. We will give examples of how to create an RDM image of each type.

3.2.1 Standard applications

A standard application is one that has an install program (e.g., SETUP.EXE) that, given a specific set of command-line parameters, can do an unattended install of the application using any directory containing the install files as input.

We will use 2 applications in this example: Microsoft Office 2003 and Microsoft Office 2003 Service Pack 1.

Although each will be a separate RDM application, they have interdependency: You must install Office first, and then you can install the Service Pack afterwards. RDM installs applications in alphabetical

order, based on the RDM image name. Therefore, we will have to name the images so that they install in the correct order.

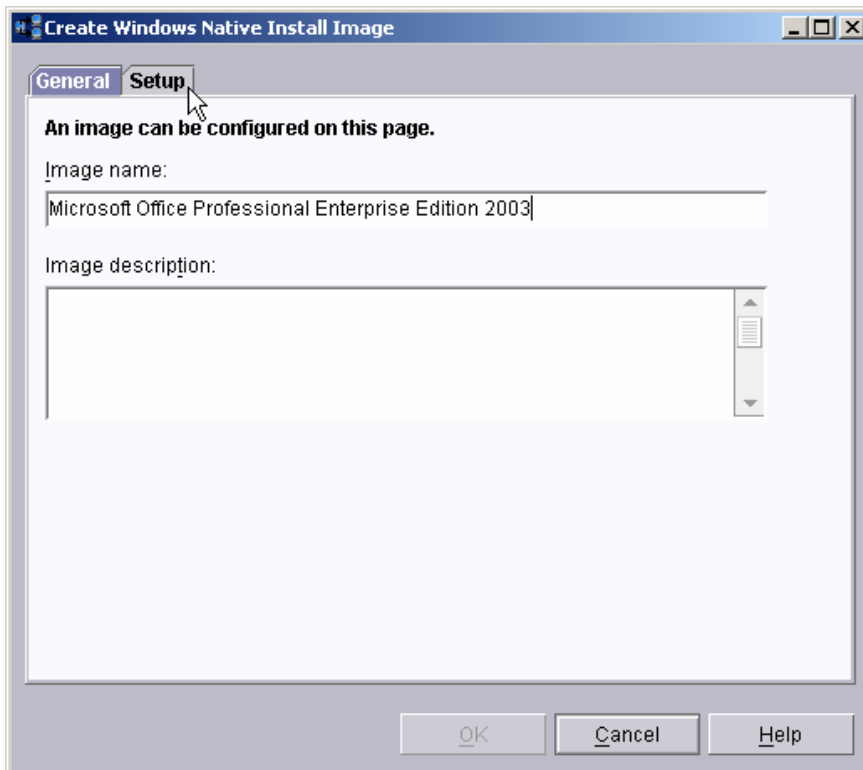
Here is the procedure:

3.2.1.1 Obtain the Microsoft Office install media

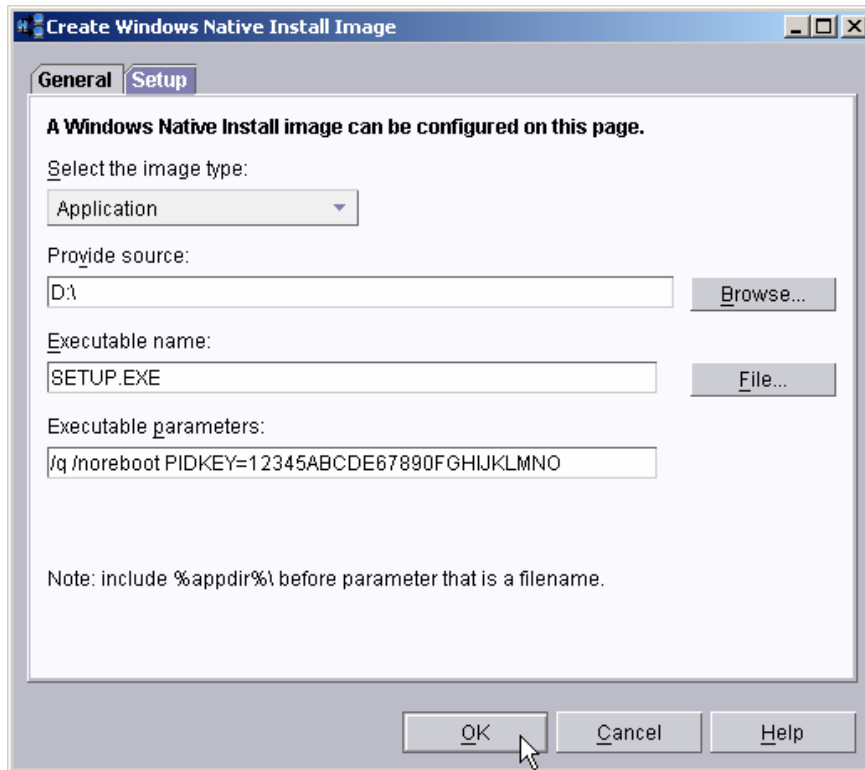
1. We used the Microsoft Office Professional Enterprise Edition 2003 CD from MSDN.

3.2.1.2 Create the RDM Windows Native Install application image

2. Open the RDM Image Management window, using the *Tasks*→*Remote Deployment Manager*→*Image Management*→*Create and Modify Images...* menu.
3. Select the *Create* button. Then select *Windows Native Install* on the dropdown menu, and select the *OK* button. This displays the *Create Windows Native Install Image* window.



4. Enter an image name on the *General* page, and then select the *Setup* page.
5. Select *Application* as the image type, from the dropdown menu.
6. Select the *Browse...* button, and navigate to the CD drive (D:\ on our server).
7. Select the *File...* button, and then select *SETUP.EXE*. Then select the *OK* button.
8. Remove D:\ from the executable name. (Including the drive letter when the source data is a CD root is an RDM bug that will be fixed, later.)
9. Enter the executable parameter as shown. The value to the right of "PIDKEY=" is the 25-character CD key for Office 2003 (without the embedded hyphens). You must use your own CD key instead of the key (it is not the real key) we used in the picture, below.



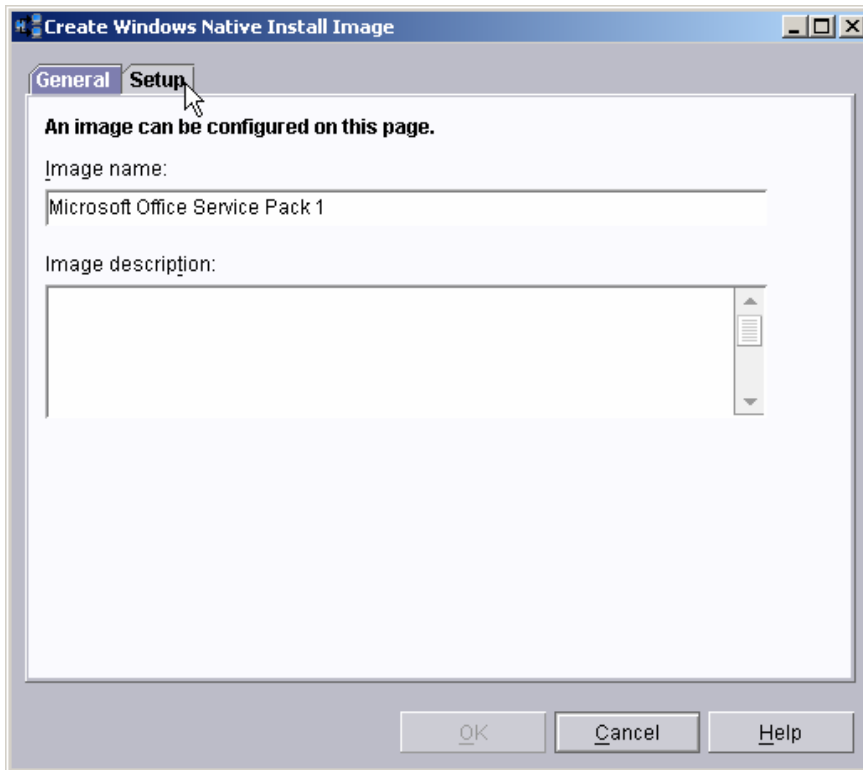
10. Select the *OK* button to create the image.

3.2.1.3 Obtain the Microsoft Office service-pack install media

11. We used a service-pack CD from MSDN that we burned from the `en_office_2003_sp1.iso` file.

3.2.1.4 Create the RDM Windows Native Install application image

12. Open the RDM Image Management window, using the *Tasks* → *Remote Deployment Manager* → *Image Management* → *Create and Modify Images...* menu.
13. Select the *Create* button. Then select *Windows Native Install* on the dropdown menu, and select the *OK* button. This displays the *Create Windows Native Install Image* window.



14. Enter an image name on the *General* page. We chose our image names carefully, to ensure that the service-pack image alphabetically follows its prerequisite office image:

Microsoft Office Professional Enterprise Edition 2003

Microsoft Office Service Pack 1

Then select the *Setup* page.

Note: Another good way to control the application installation order is to index their names with a sequence number. For example, we could name our images like this:

01 Microsoft Office 2003

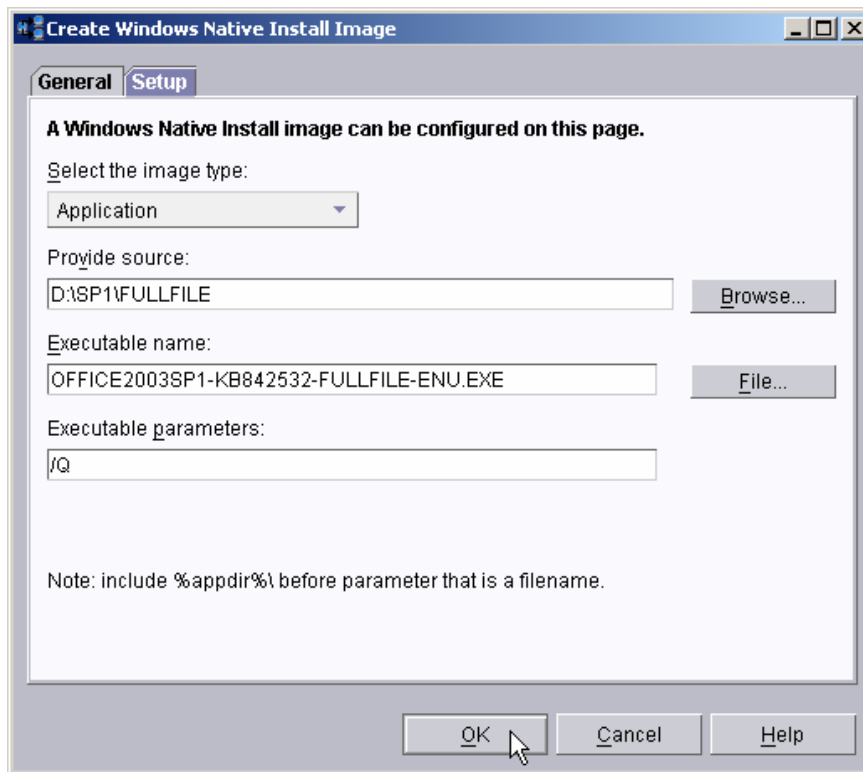
02 Microsoft Office SP 1

03 WinZip 9.0

04 Adobe Reader 6.1

05 IBM Director Agent 4.21

15. Select *Application* as the image type, from the dropdown menu.
16. Select the *Browse...* button, and navigate to the CD drive (D:\ on our server) and to the D:\SP1\FULLFILE directory.
17. Select the *File...* button, and then select the executable name shown in the picture, below. Then select the *OK* button.
18. Enter the /Q executable parameter.



19. Select the *OK* button to create the image.

3.2.2 Irregular application

An irregular application is similar to a standard application, except that it requires some customization in order to accomplish a successful unattended install.

Our example application is WinZip 9.0 SR-1. As we will see below, there is a requirement to install WinZip from install files that are in its final directory, so this application requires some customization in order to install it with RDM. Here is the procedure:

3.2.2.1 Download WinZip

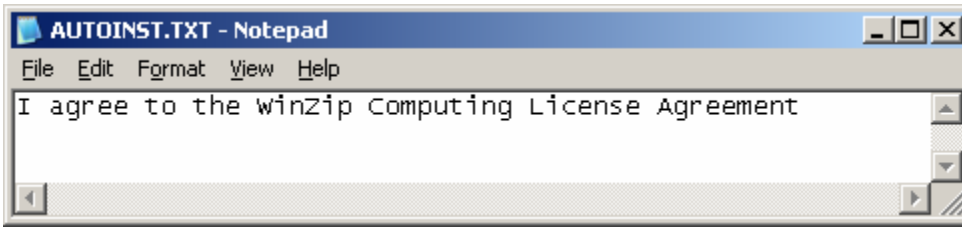
1. Download the install file from the WinZip web site. Our file is winzip9.0.exe, whose size is 2,366 KB.

3.2.2.2 Extract the install directory

2. On your RDM console system, extract the install directory from the winzip9.0.exe file. We did this by running this file and processing its user interface up to the point where it is ready to install WinZip. Select the Setup button on the first window, and select the OK button on the second window. On the third window (it says "Thank you for installing WinZip!" near the top of the window), do not select the Next button; select the Close button. Then select Yes on the next window. The result was that the (default) C:\Program Files\WinZip directory contained 26 files.

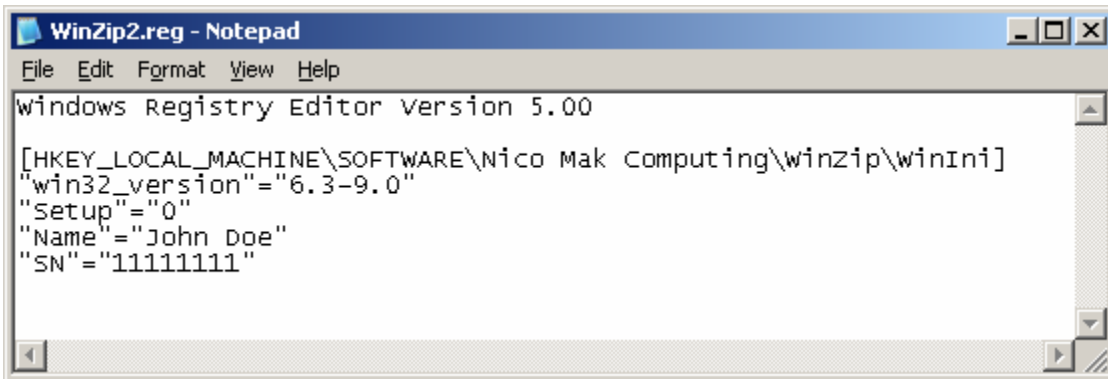
3.2.2.3 Customize the install directory

3. Create a file C:\Program Files\WinZip\AUTOINST.TXT with the following content:



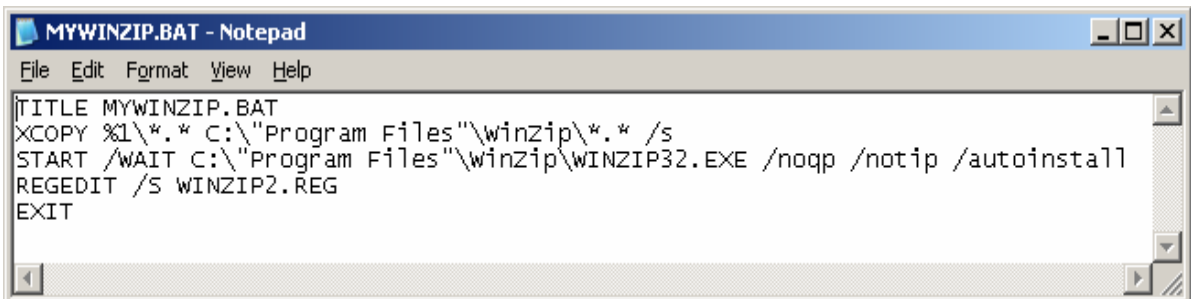
This prevents the license screen from appearing each time you run WinZip.

4. Create a file C:\Program Files\WinZip\WINZIP2.REG with content like the following:



The easiest way to create this file is to export that key from the registry of a system that already has a licensed version (in which you have registered your name and serial number) of WinZip correctly installed. You must use your correct name and serial number.

5. Create a file C:\Program Files\WinZip\MYWINZIP.BAT with the following content:



We will use this file to install WinZip. We did it this way because RDM creates the install directory with a name like C:\APP2, and the WINZIP32.EXE file needs to run from its final install directory, C:\Program Files\WinZip. So RDM will run a command like

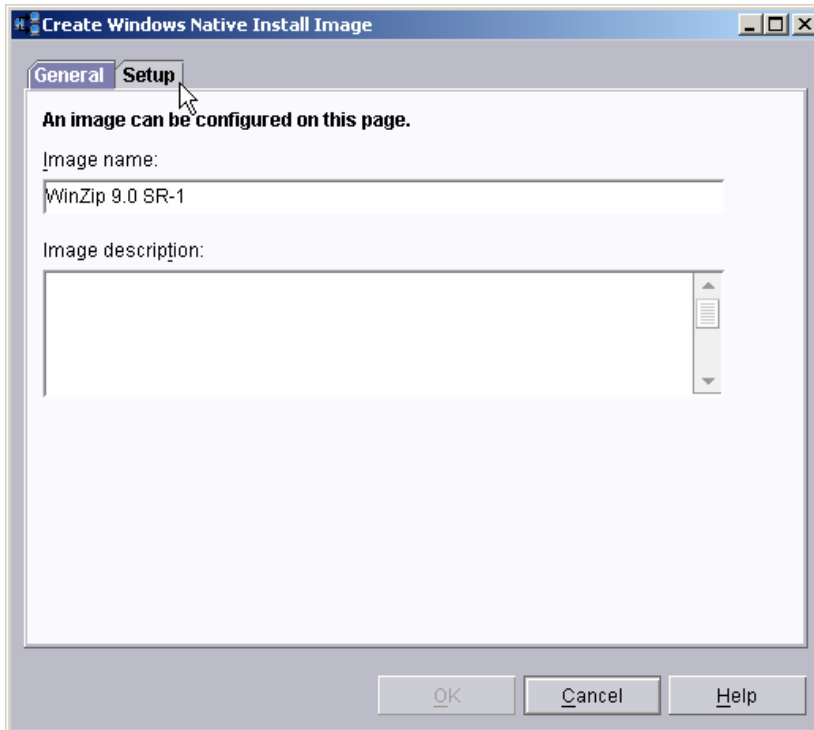
```
MYWINZIP.BAT "C:\Program Files\WinZip"
```

to install WinZip.

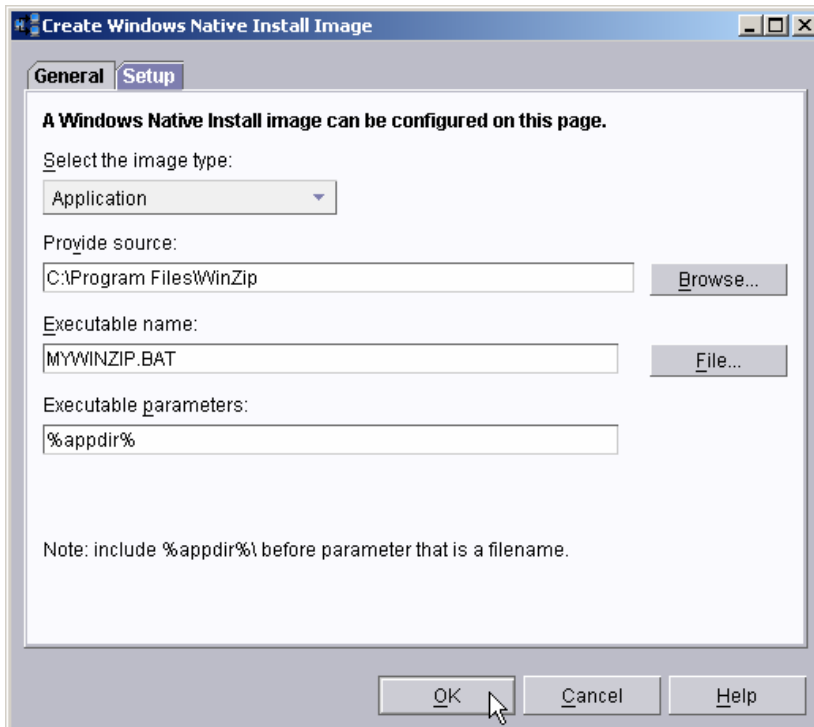
Note the use of the EXIT statement in the batch file. You must do this for any batch file that is used as an application's install program. The TITLE statement is optional, but it is useful when debugging.

3.2.2.4 Create the RDM Windows Native Install application image

6. Open the RDM Image Management window, using the *Tasks*→*Remote Deployment Manager*→*Image Management*→*Create and Modify Images...* menu.
7. Select the *Create* button. Then select *Windows Native Install* on the dropdown menu, and select the *OK* button. This displays the *Create Windows Native Install Image* window.



8. Enter an image name on the *General* page, and then select the *Setup* page.
9. Select *Application* as the image type, from the dropdown menu.
10. Select the *Browse...* button, and navigate to the C:\Program Files\WinZip install directory that you created in step 2 above.
11. Select the *File...* button, and then select *MYWINZIP.BAT*. Then select the *OK* button.
12. Enter the executable parameter `%appdir%`, as shown.

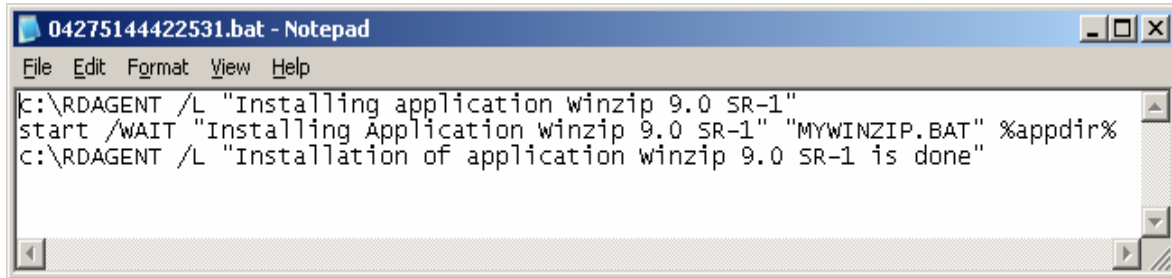


13. Select the *OK* button to create the image.

The resulting RDM image contains 2 files: a batch file that is used to kick off the application install, and a zip file that contains the application install directory.

3.2.2.5 RDM install logic

When RDM installs this application, via the APPSINST.BAT command, it runs this batch file (which will be renamed to something like APP4.BAT at run time).



```

c:\RDAGENT /L "Installing application winzip 9.0 SR-1"
start /WAIT "Installing Application winzip 9.0 SR-1" "MYWINZIP.BAT" %appdir%
c:\RDAGENT /L "Installation of application winzip 9.0 SR-1 is done"

```

The variable %appdir% gets replaced with a value like C:\APP4 when the COPYAPP.BAT command runs the LCCUSTOM.EXE program. Note that at run time, in Windows, %appdir% is not an environment variable.

Important: Although RDM contains built-in logic to handle the %appdir% value in the application-install command line (i.e., the “start /WAIT ...” statement in the above batch file), it has no logic to handle that value in any of the other files in the install directory. For example, if your application install directory contains an INI file that needs the %appdir% value in one of its statements, then you have to add customized logic to do the substitution. The logic you add would be to download LCCUSTOM.EXE and to use it to change %appdir% to its correct run-time value.

3.2.3 MSI application

An MSI application is one that can use the Microsoft MSIEXEC.EXE program to do the unattended install. MSIEXEC.EXE is not part of the directory that contains the application’s install files.

Our example application is Adobe Reader 7.0.0. Here is the procedure:

3.2.3.1 Download Adobe Reader

1. Download the full install file from the Adobe web site. Don’t use Adobe Download Manager. We got the English version, a file named AdbeRdr70_enu_full.exe, whose size is 20,311 KB.

3.2.3.2 Extract the install directory

2. On your RDM console system, extract the install directory from the AdbeRdr70_enu_full.exe file. We did this by running this file and processing its user interface up to the point where it is ready to install Adobe Reader. Do not select the *Install* button; select the *Cancel* button. Then select *Yes* on the next window and *Finish* on the final window.

The result of this procedure was that it created the following directory:

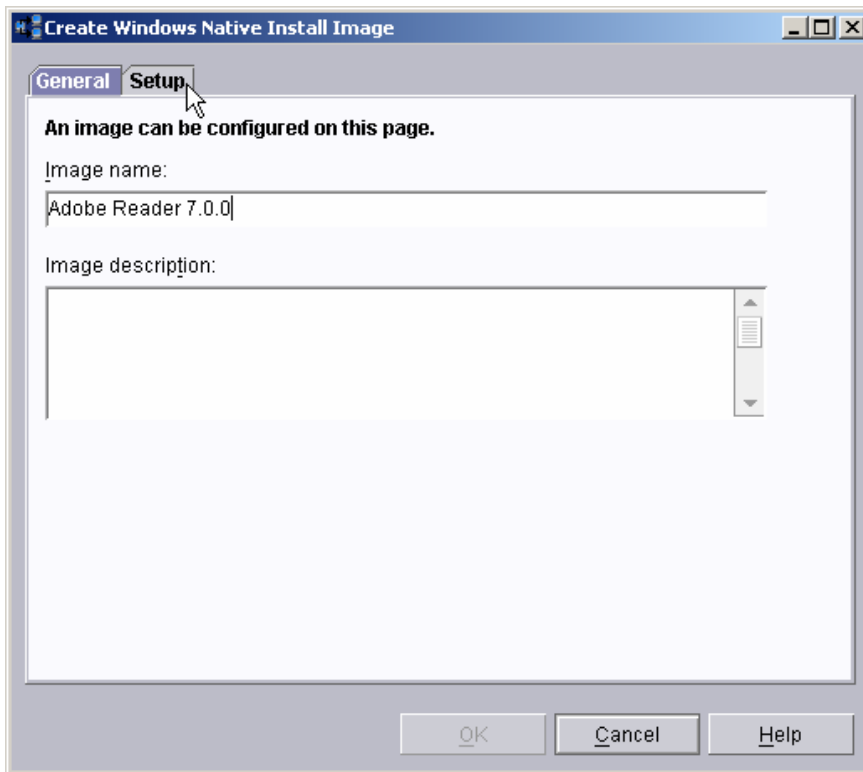
```
C:\Program Files\Adobe\Acrobat 7.0\Setup Files\RdrBig\ENU
```

This directory contains the install files that we need.

3.2.3.3 Create the RDM Windows Native Install application image

3. Open the RDM Image Management window, using the *Tasks* → *Remote Deployment Manager* → *Image Management* → *Create and Modify Images...* menus.

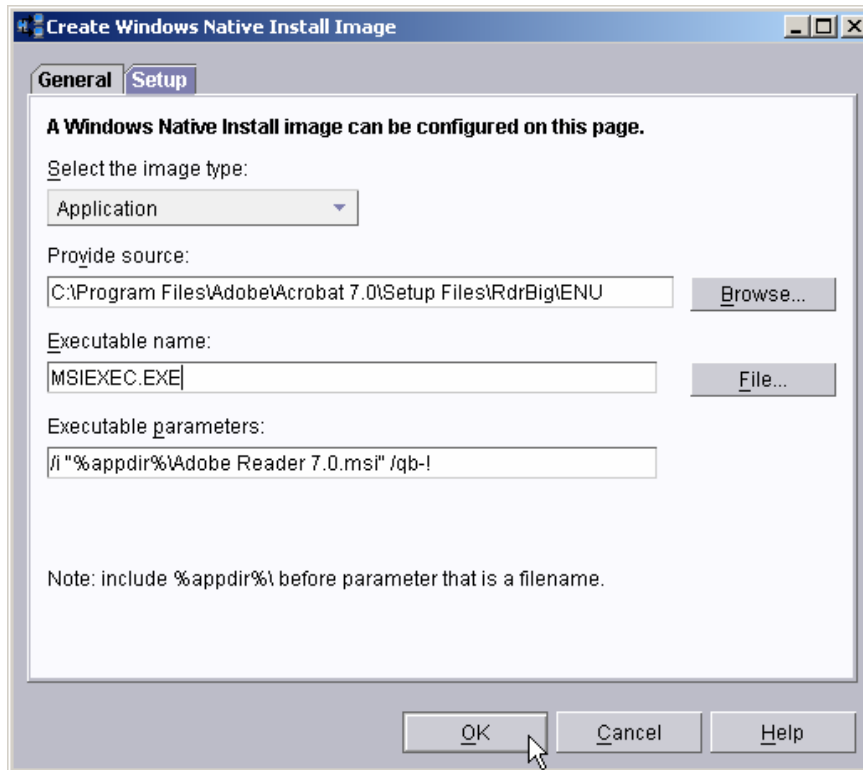
4. Select the *Create* button. Then select *Windows Native Install* on the dropdown menu, and select the *OK* button. This displays the *Create Windows Native Install Image* window.



5. Enter an image name on the *General* page, and then select the *Setup* page.
6. Select *Application* as the image type, from the dropdown menu.
7. Select the *Browse...* button, and navigate to the install directory that you created in step 2 above.
8. Select the *File...* button, and then select one of the files in the list (e.g., SETUP.EXE). Then select the *OK* button.
9. In the text field, change *SETUP.EXE* to **MSIEXEC.EXE**.
10. Enter the executable parameters as shown:

```
/i "%appdir%\Adobe Reader 7.0.msi" /qb-!
```

Note the use of `%appdir%`, here. Because *Adobe Reader 7.0.msi* is a file name, you must provide a path name (this is an RDM requirement).



11. Select the *OK* button to create the image.

3.2.4 Collection of applications

You can treat a set of applications, for RDM purposes, as a single application. You put their installation directories into a single tree, and you use a batch file to install all the applications in the set.

A typical example for this kind of application is a set of Microsoft updates or hotfixes. Since it is common to install many (e.g., 20 or 30) hot fixes, it is simpler to bundle them as a single RDM application. We'll use this example, here. Here is the procedure:

3.2.4.1 Download the hotfixes

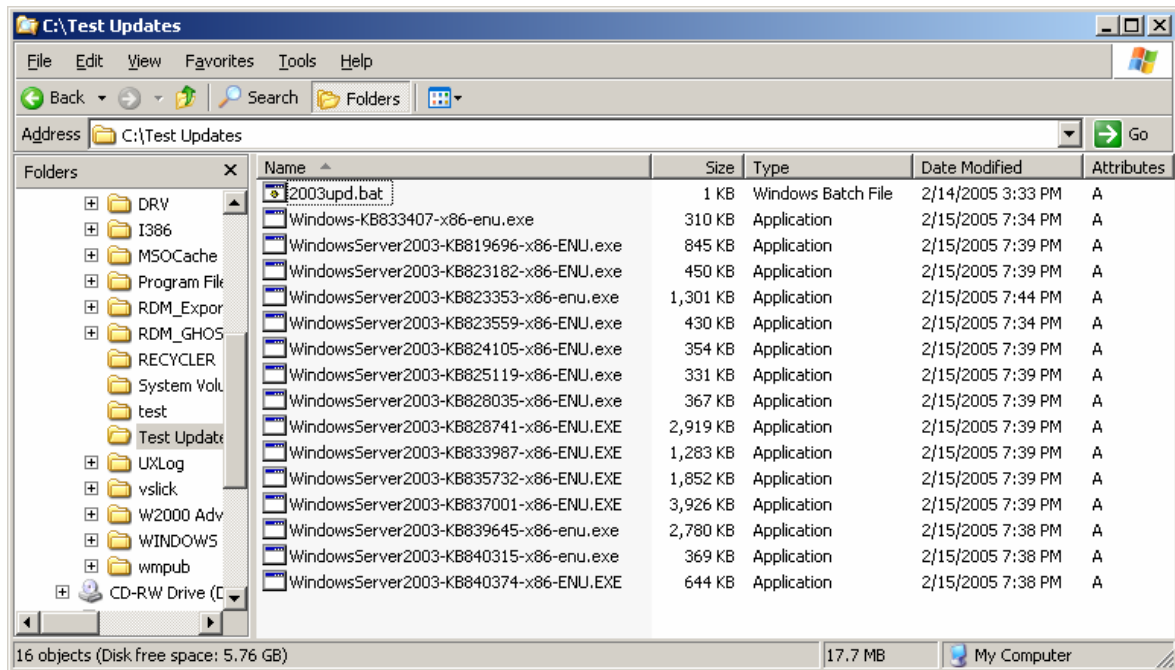
1. Download the updates or hotfixes from the Microsoft web site.

3.2.4.2 Create an install directory

2. On your RDM console system, create a directory for the downloaded executables. We used this directory:

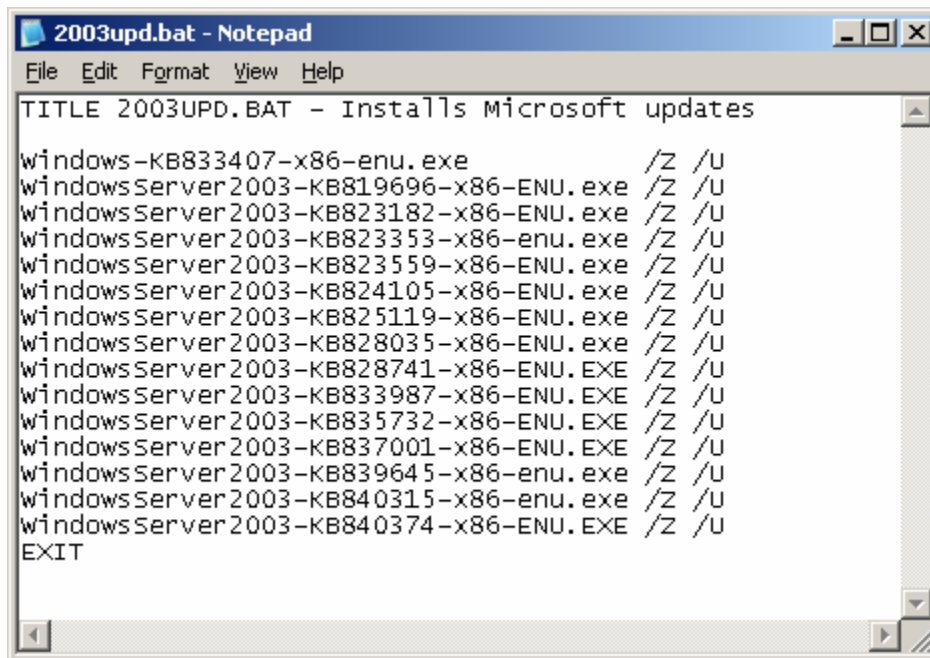
C:\Test Updates

3. Copy the executable file for each update from step 1 into C:\Test Updates.



3.2.4.3 Create a batch file that installs the updates

4. Create a batch file with a single command for each executable.



Note that these example updates all use the same command-line syntax. The updates that you install may use different syntax. Consult the Microsoft documentation for the appropriate syntax.

We used /Z (do not restart the computer) and /U (unattended setup mode). With these parameters, you will see the user interface on the target system's monitor. If you want to suppress the user interface, you may add the /Q parameter.

Note that there is no error handling in this batch file. Ideally, you would add error handling for each of the updates. But unless you know the return codes (if any) that each install program uses, it would be difficult to check for an error in the batch file. So before using an RDM

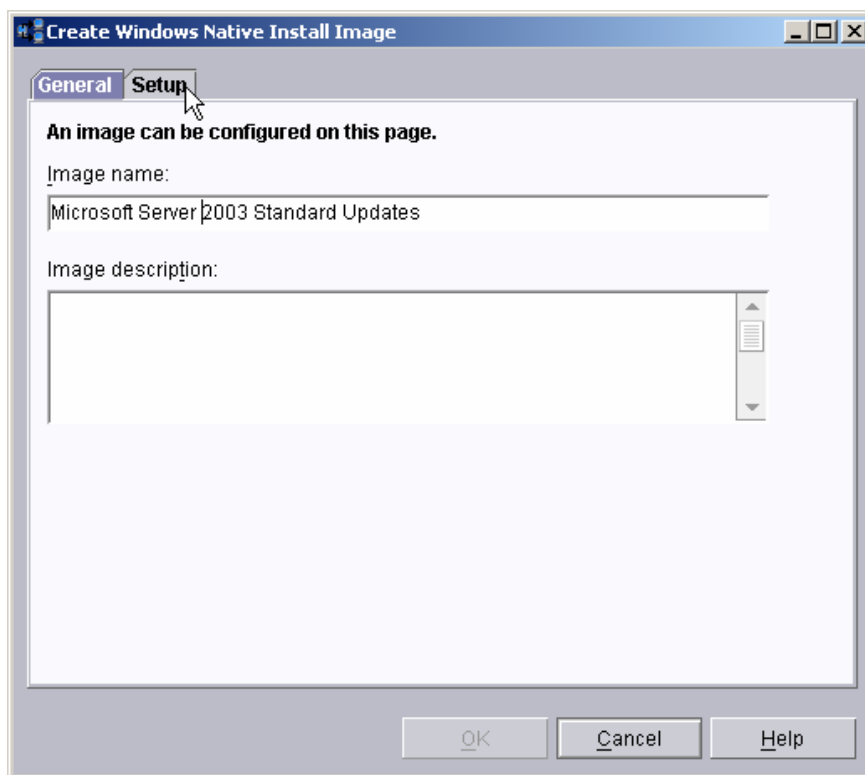
application like this one in production, you should thoroughly test it, validating that each update installed properly in your tests.

Note the use of the EXIT statement in the batch file. You must do this for any batch file that is used as an application's install program. The TITLE statement is optional, but it is useful when debugging.

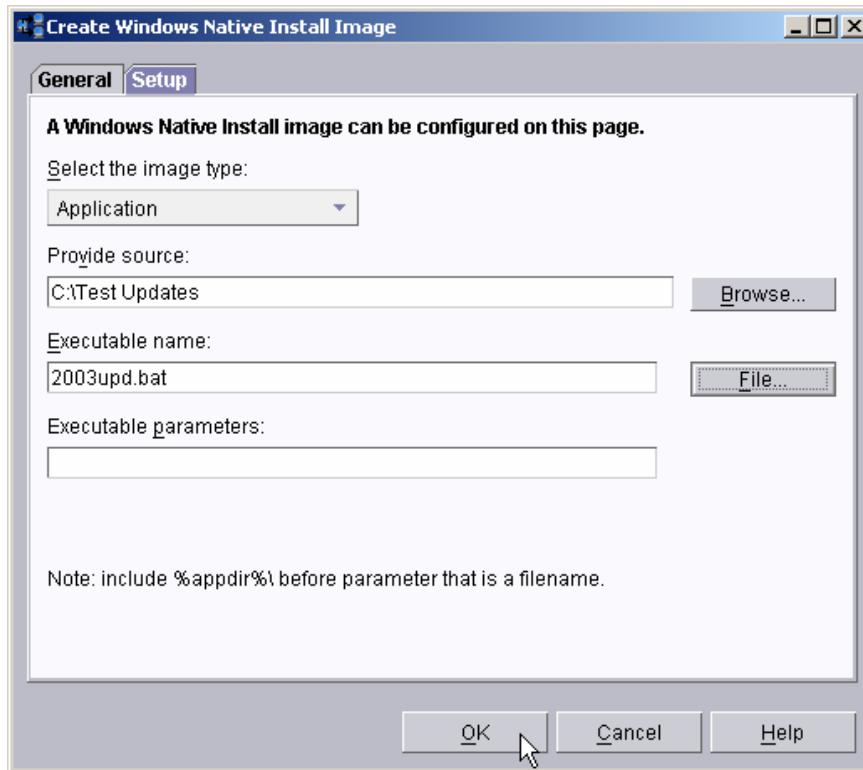
Also note that our example is for Windows 2003. For Windows 2000, you would have to add a QCHAIN.EXE statement to the end of the batch file. Again, consult the Microsoft documentation for details.

3.2.4.4 Create the RDM Windows Native Install application image

5. Open the RDM Image Management window, using the *Tasks*→*Remote Deployment Manager*→*Image Management*→*Create and Modify Images...* menus.
6. Select the *Create* button. Then select *Windows Native Install* on the dropdown menu, and select the *OK* button. This displays the *Create Windows Native Install Image* window.



7. Enter an image name on the *General* page, and then select the *Setup* page.
8. Select *Application* as the image type, from the dropdown menu.
9. Select the *Browse...* button, and navigate to the install directory that you created in step 2 above.
10. Select the *File...* button, and then select the batch file that you created in step 4 above (e.g., 2003UPD.BAT). Then select the *OK* button.
11. This batch file has no executable parameters, as shown:



12. Select the *OK* button to create the image.

3.2.5 IBM Director Agent

This section describes the steps for installing IBM Director Agent.

The unattended install procedure for IBM Director Agent changed in version 5.1, versus the procedure used in versions 4.22 and earlier. We will describe both procedures in this section. We used version 4.21 and version 5.1 in this example.

We treat IBM Director Agent as a standard application, and we use the same technique shown in section 3.2.1 above. (We could have chosen to install as a MSI application, instead.)

3.2.5.1 Obtain the install files

1. You can get the IBM Director Agent install files from an IBM Director CD or from the public IBM web site:
 - On the CD, these files are in this directory:
 - Version 4.22 and earlier: D:\director\agent\windows\i386
 - Version 5.1 and later: D:\director\agent\windows\i386\FILES
 - You can download the dir4.21_agent_windows.zip file from this web site:
<http://www-1.ibm.com/support/docview.wss?uid=psg1MIGR-58219>

3.2.5.2 Create an install directory

For some users, this step is optional. For example, if you want to install the default configuration, then you can use the CD as your install directory. However, in order to install a different configuration, you must create an install directory (so that you can later modify the response file).

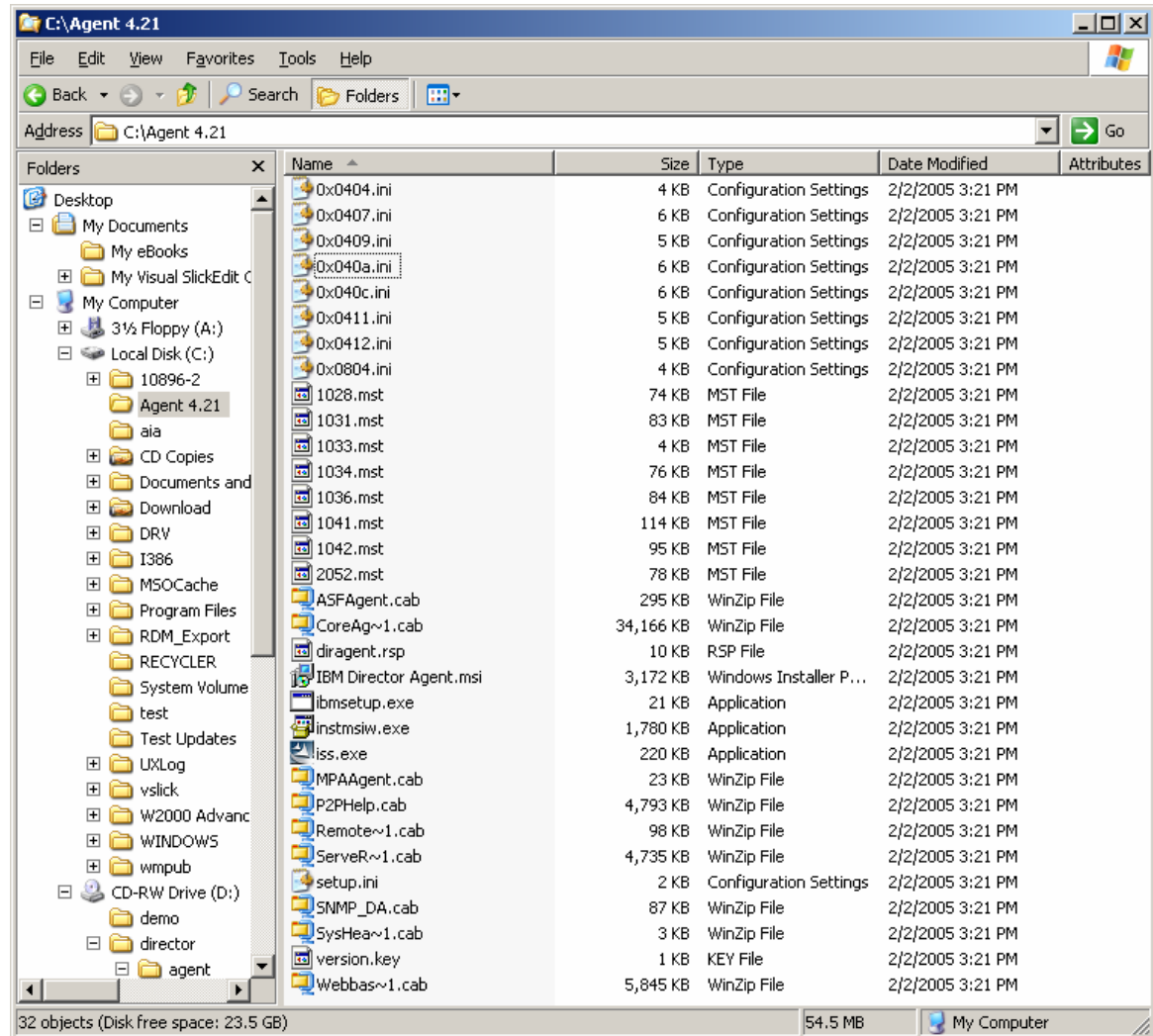
- On your RDM console system, create a directory for the downloaded executables. We used this directory:

C:\Agent 4.21

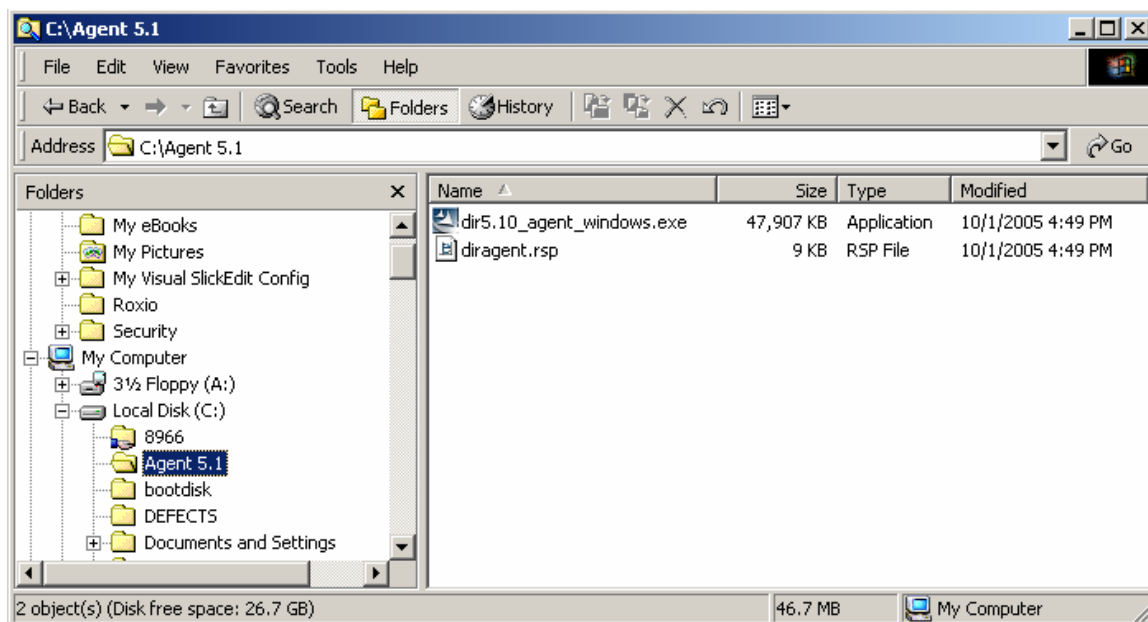
- Copy the install files to the above directory:

- From the CD, copy the files from the appropriate directory:
 - Version 4.22 and earlier: D:\director\agent\windows\i386
 - Version 5.1 and later: D:\director\agent\windows\i386\FILES
- From the web download, unzip the files

The result will be the following, for version 4.22 and earlier:



The result will be the following, for version 5.1 and later:



3.2.5.3 Modify the response file

Depending on your needs, you will probably have to modify the response file, DIRAGENT.RSP.

4. **Important:** Make sure that you use a value of N for the following parameter:

```
RebootIfRequired = N
```

This line suppresses the reboot at the end of the IBM Director Agent install. If you used a version 4.22 or earlier CD in section 3.2.5.1 above, it's default value is N and you don't have to change it; but if you used a downloaded ZIP file, then you need to change this value from Y to N.

5. Make any other changes you need. In our example, we changed the following line:

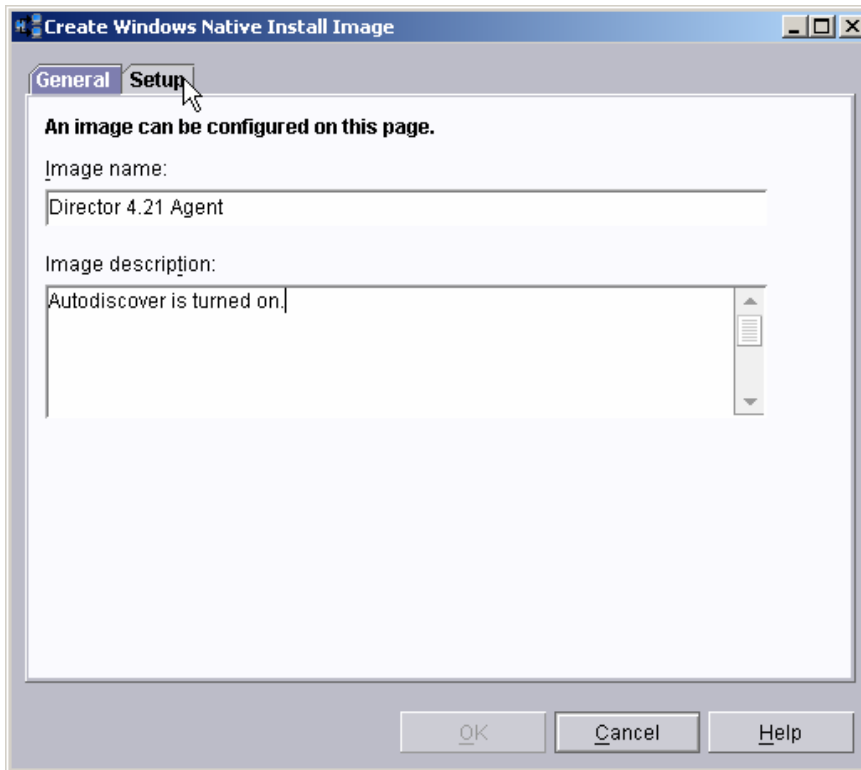
```
AddKnownServerAddress=TCPIP::10.2.0.5
```

This line identifies the IBM Director Server, so that systems are automatically discovered by IBM Director. We also changed this line, in our version 5.1 file:

```
WakeOnLan=1
```

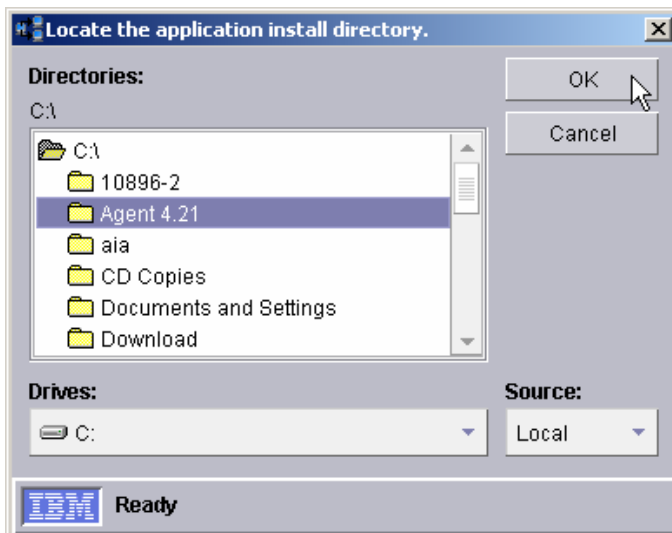
3.2.5.4 Create the RDM Windows Native Install application image

6. Open the RDM Image Management window, using the *Tasks*→*Remote Deployment Manager*→*Image Management*→*Create and Modify Images...* menu.
7. Select the *Create* button. Then select *Windows Native Install* on the dropdown menu, and select the *OK* button. This displays the *Create Windows Native Install Image* window.

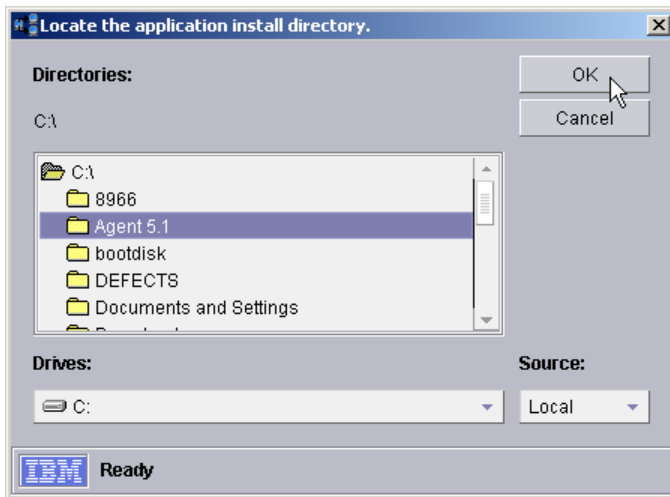


8. Enter an image name and an optional description on the *General* page, and then select the *Setup* page.
9. Select *Application* as the image type, from the dropdown menu.
10. Select the *Browse...* button, navigate to the directory (the one we created in section 3.2.5.2 above), and select the *OK* button.

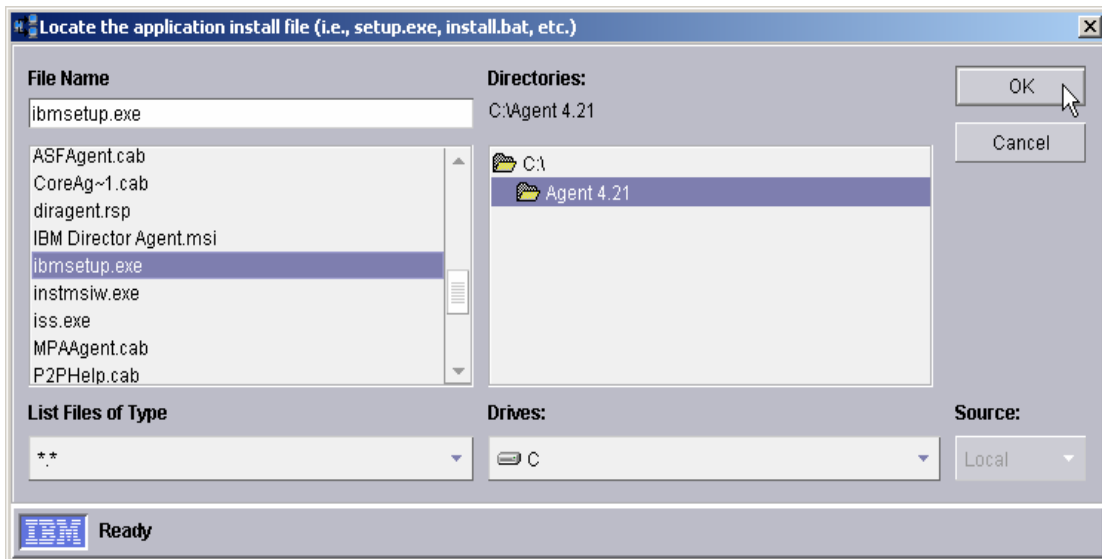
If you are using version 4.22 or earlier:



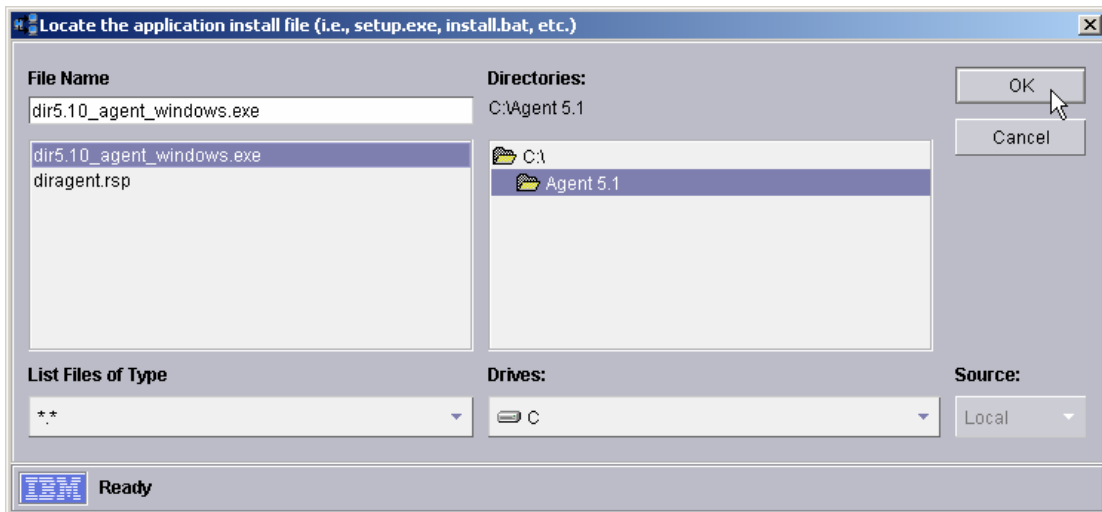
If you are using version 5.1 or later:



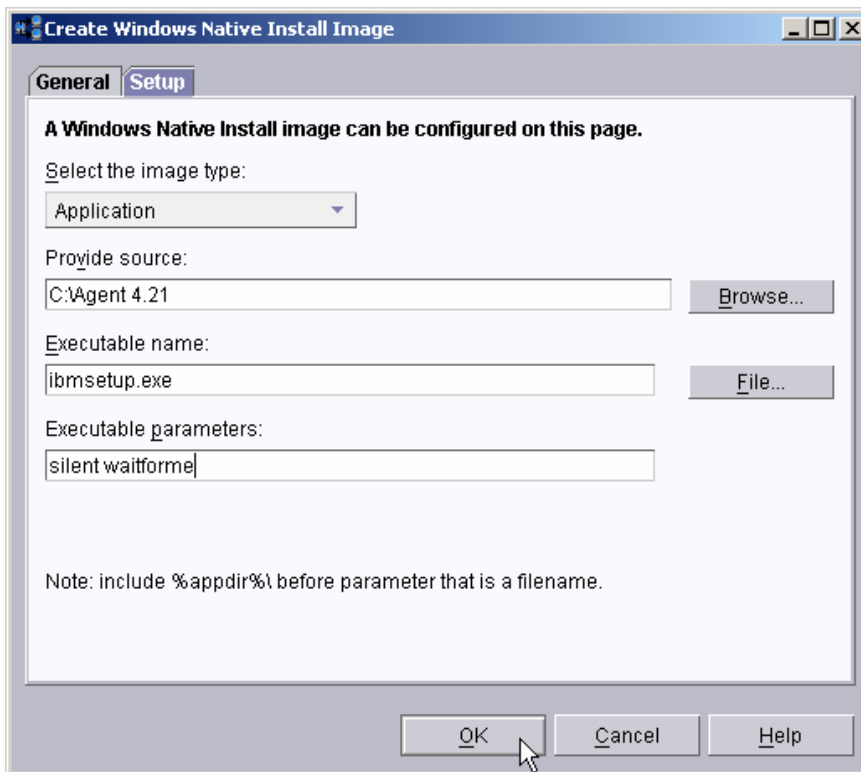
11. Select the *File...* button, and then select the appropriate executable. Then select the *OK* button. If you are using version 4.22 or earlier, select IBMSETUP.EXE:



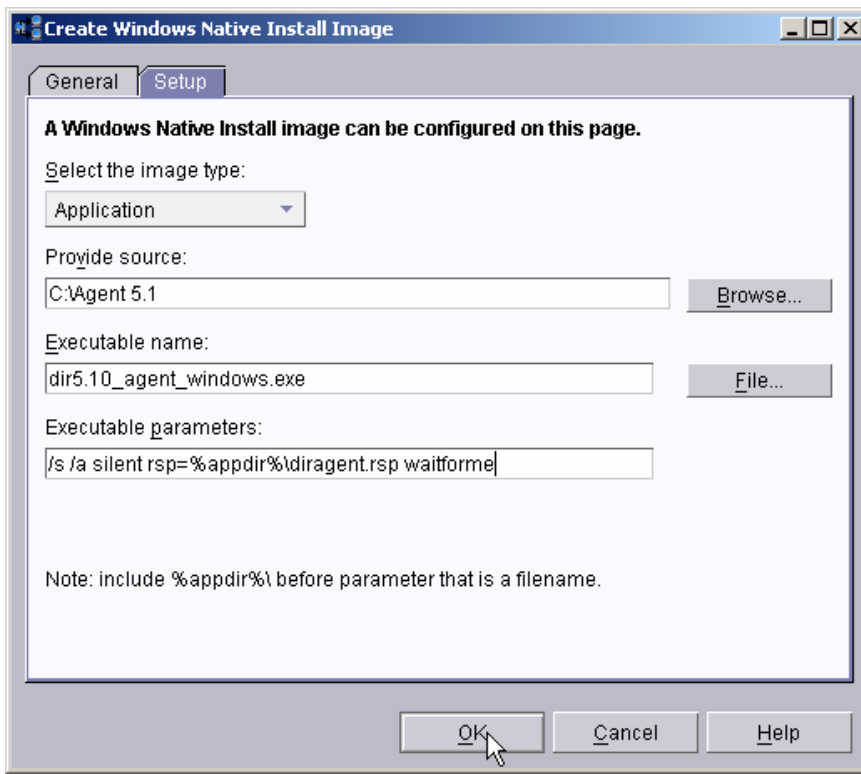
- If you are using version 5.1 or later, select DIR5.10_AGENT_WINDOWS.EXE:



12. Enter the executable parameter as shown below.
if you are using version 4.22 or earlier:

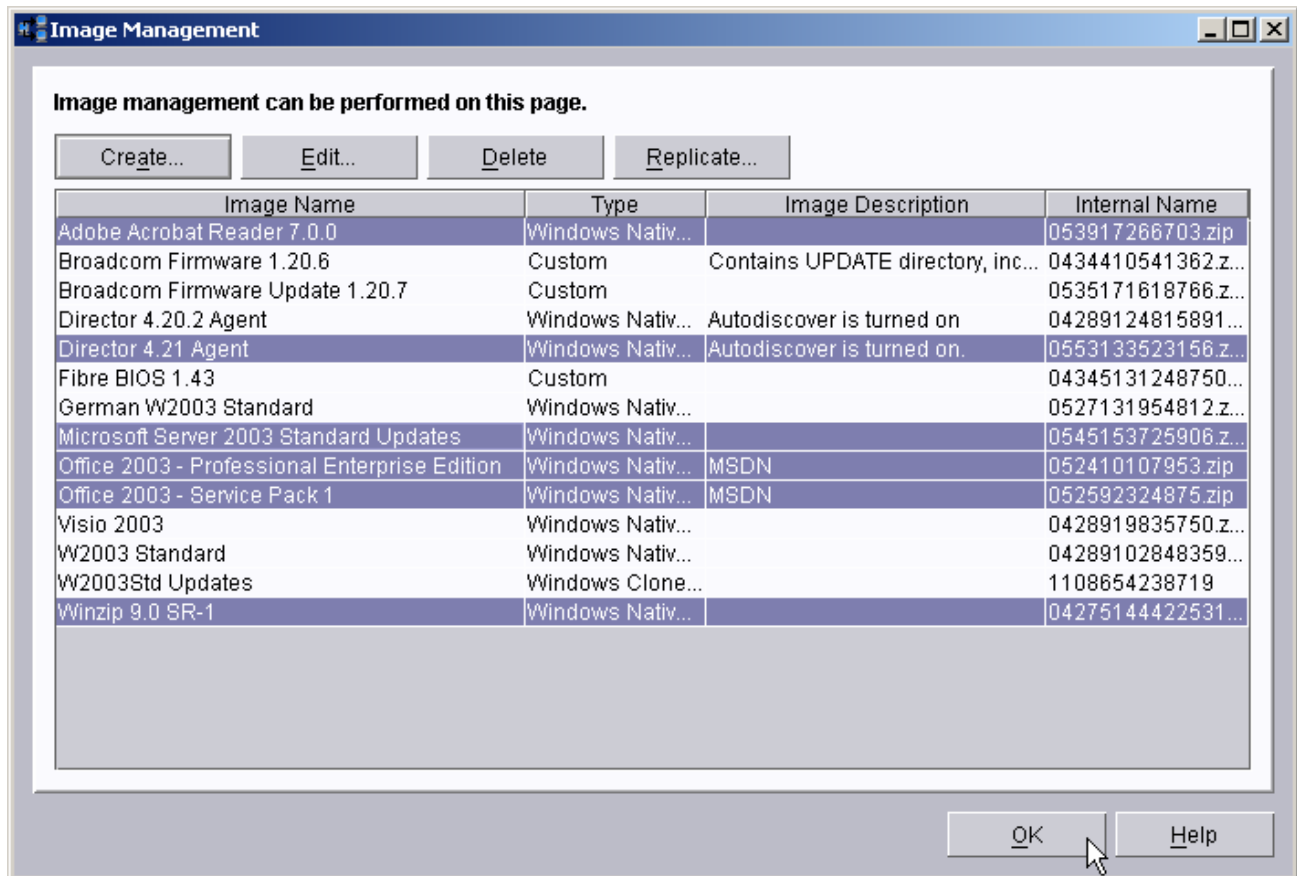


If you are using version 5.1 or later:



13. Select the *OK* button to create the image.

In the above sections, we created 6 images. Here is the Image Management window, with those images selected:



3.3 Installing applications

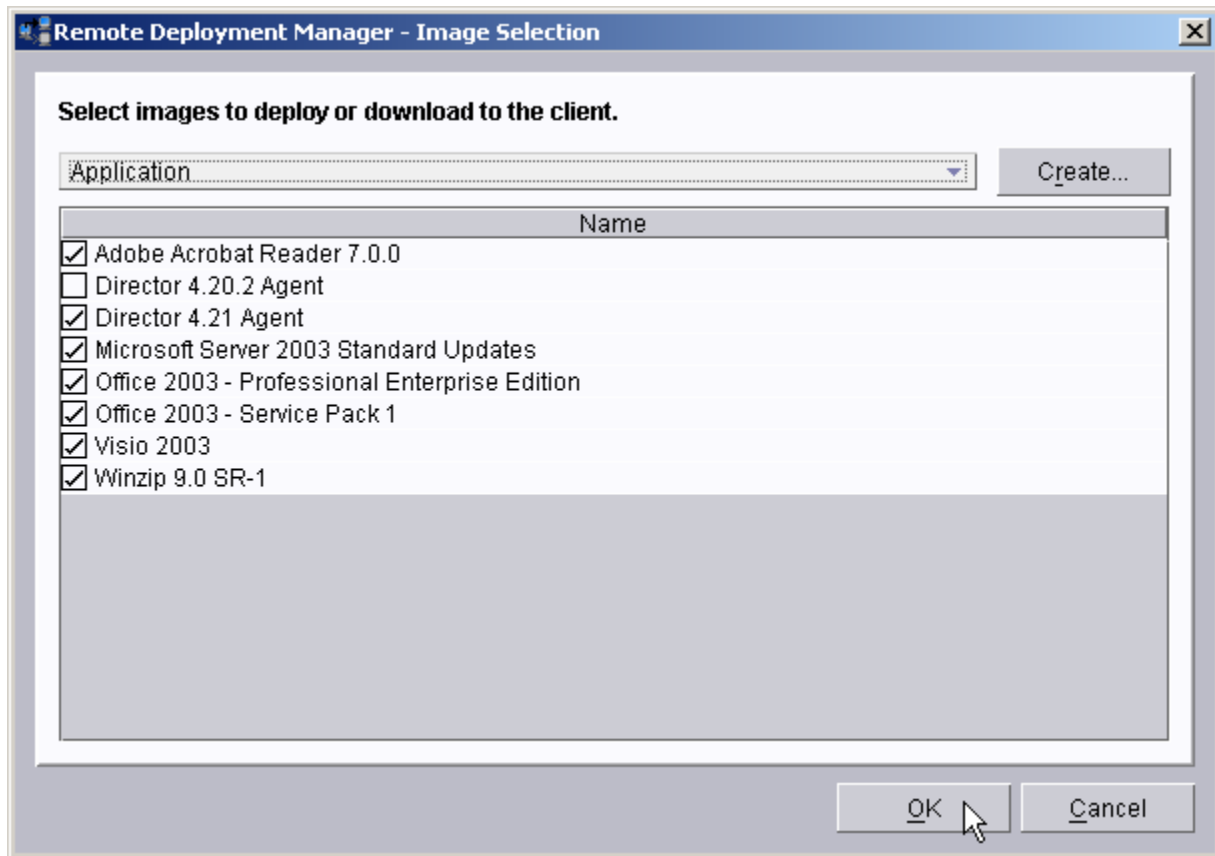
There are several ways to install applications in RDM.

- RDM's built-in application-install capability
- Customized RDM's built-in application-install capability
- Via the task's command list
- Via CMDLINES.TXT

This section uses several examples to show how to do the above.

3.3.1 Using RDM's built-in application-install capability

The easiest way to install applications is to use RDM's built-in capabilities. The procedure is the following: when you create (or modify) a *Windows Native Install* task, just select the applications that you want to install in the appropriate wizard or properties window.



The task will automatically install all the selected applications, in alphabetical order.

Important requirements for this method include:

- None of the applications can reboot the system during its installation procedure.
- You have to ensure that each application's prerequisites are installed prior to installing the application itself.

If an application does not meet these requirements, you must install it in another way.

3.3.2 Customizing RDM's built-in application-install capability

For this discussion, assume that we have to install an application plus its hotfixes, and that the application requires that the system reboot before you install its hotfixes. Certain versions of Citrix Metaframe Client could be one example of such an application. The built-in RDM application-install capability installs all applications associated with the task, in order, without rebooting between any of the application installs. So we must customize RDM to handle the required-reboot situation.

We will illustrate the procedure by creating a *Windows Native Install* task that installs Adobe Acrobat Reader, reboots the target system, and then installs WinZip.

The general procedure is the following:

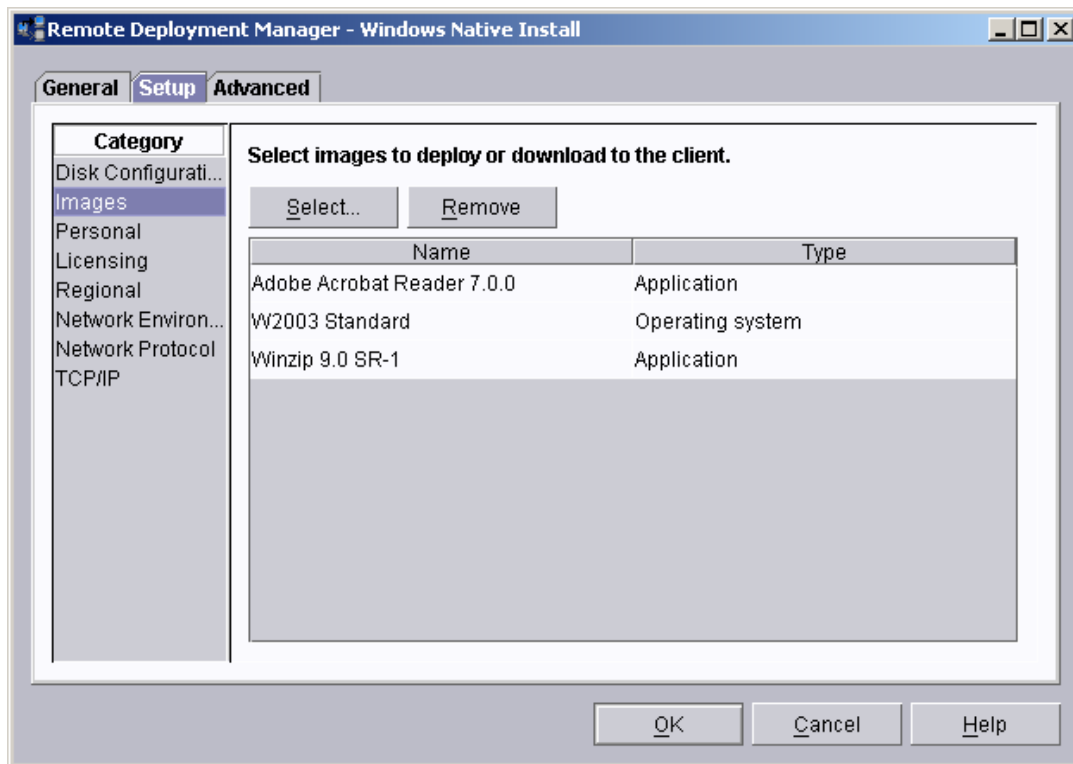
1. Create a *Windows Native Install* task that contains *both* applications. The hotfixes application (we will use WinZip as this application) should be the last application (in alphabetic order).
2. Copy the task folder and its contents to a temporary directory.
3. Edit the task, and deselect the RDM image of the hotfixes application.
4. Create a customized version of the task's TASKWORK.BAT file, called MYTASKWK.BAT:
 - Add a statement that reboots right after installing the built-in applications.

- Add a block of statements that download and install the hotfixes application (using the files from the temporary directory as a guide).
5. Create customized versions of several other files that are used to install applications.
 6. Create a new DOS 7.1 system environment that contains an INSTALL.BAT file that is modified to handle 2 sets of applications.
 7. Test the task.

We will illustrate the procedure, below, using Adobe Reader and WinZip as our applications.

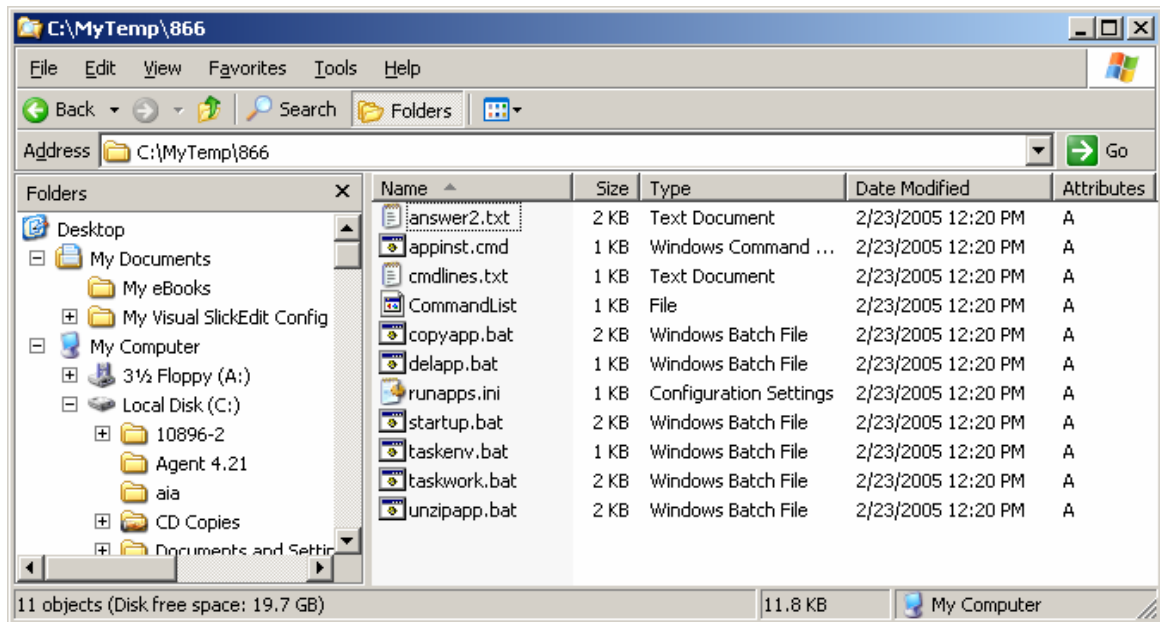
3.3.2.1 Create the task

1. Create a *Windows Native Install* task (named *W2003Std Adobe WinZip*) that contains both applications. If you then edit the task, you will see the images that the task will use in this window:



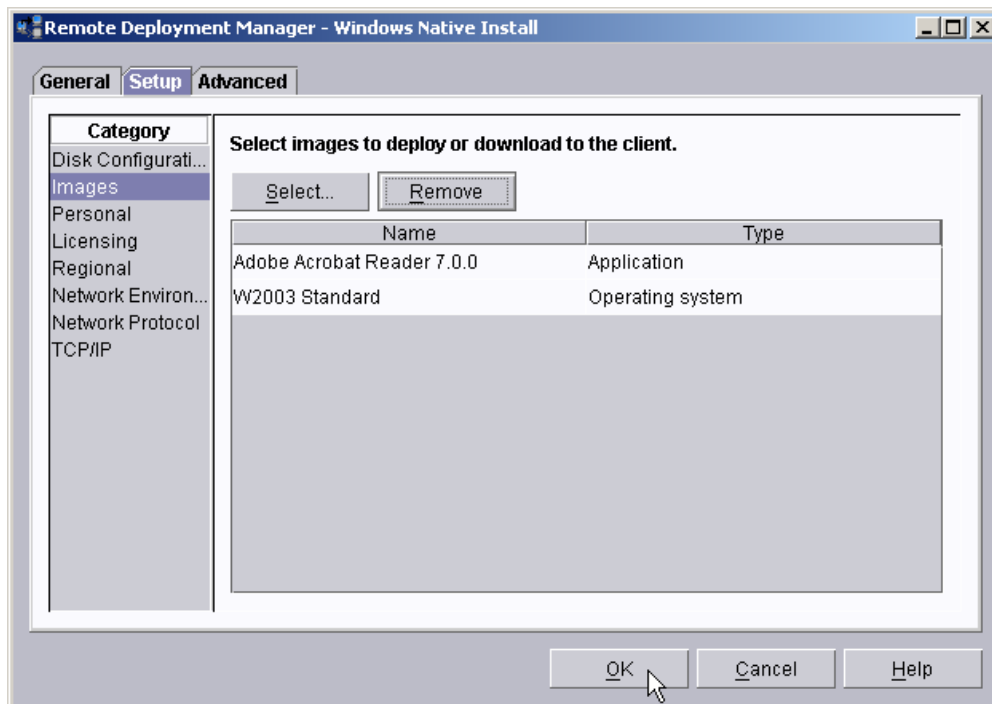
3.3.2.2 Copy the task folder

2. Edit the *W2003Std Adobe WinZip* task. Got to the *Advanced* page to find its task folder name (ours was *C:\Program Files\IBM\RDM\repository\template\14\866*).
3. Copy the task folder and its contents to a temporary directory (we created *C:\MyTemp\866*).



3.3.2.3 Copy the task

4. In the Director console, right click on the task and select the *Copy task...* menu.
5. Give the task a new name (we used *W2003Std Adobe WinZip 2*).
6. Go to the *Setup* page and select *Images*.
7. Select the *WinZip 9.0 SR-1* image, and then select the *Remove* button. You will then have only 2 images in the task:

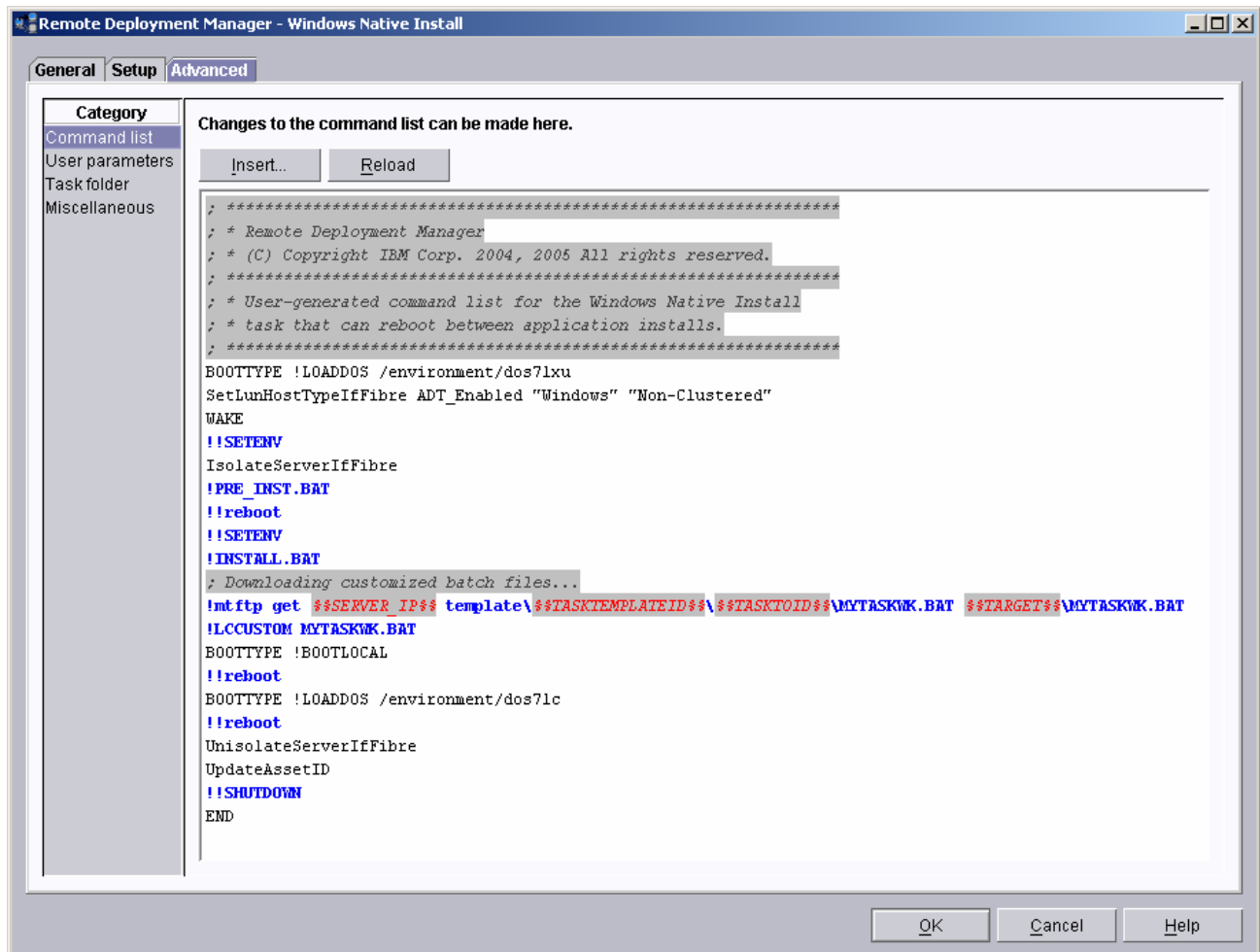


8. Go to the *Advanced* page and edit the command list. Make the changes as shown in the picture below:

- Add a block of comments at the top of the command list. It's a good practice always to document your customization with comments (in the file you are customizing).
- Change the first BOOTTYPE command to use system environment dos71xu (instead of dos71x).
- Add 3 statements to the MYTASKWK.BAT file:
 - A comment: ; Downloading customized batch files...
 - A command to download the file: !mtftp GET %%server_ip%% template\%%TASKTEMPLATEID%%\%%TASKTOID%%\MYTASKWK.BAT %%TARGET%%\MYTASKWK.BAT
 - A command to substitute parameter values: !LCCUSTOM MYTASKWK ,BAT

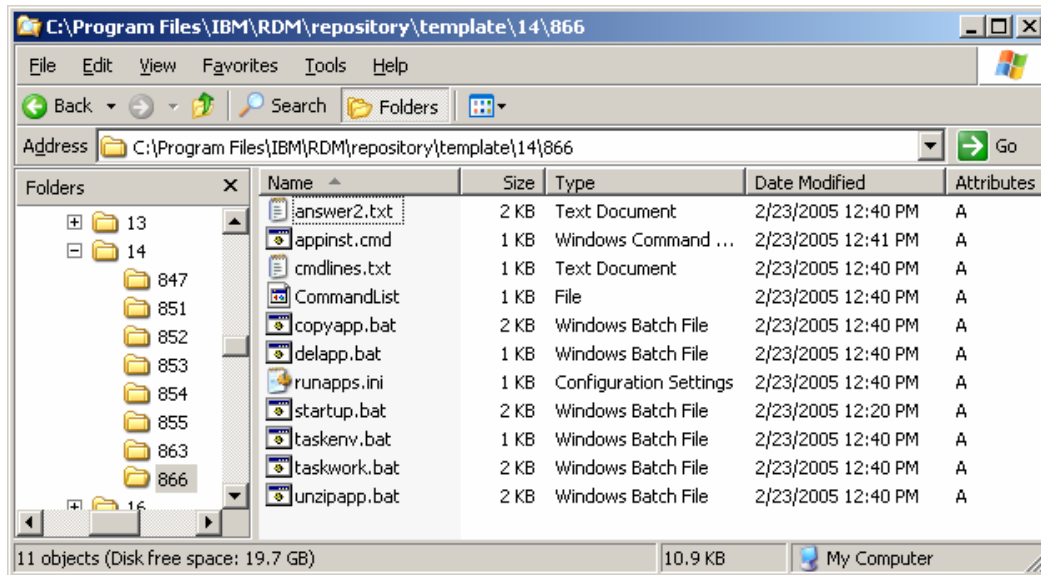
Note the use of variables in the command list.

- The variable names are preceded and followed by %%. This is required for all variables that are used in the command list.
- %%SERVER_IP%% is the IP address of the RDM Deployment Server.
- %%TASKTEMPLATEID%% is the template folder name.
- %%TASKTOID%% is the task folder name.
- %%TARGET%% is the path into which to download the files. The INSTALL.BAT file sets the value of this variable (usually D:, since the RAM drive is usually C:).



9. Then select the *OK* button to save the changes.
10. Look at the task folder with Windows Explorer. Our task folder name was C:\Program Files\IBM\RDM\repository\template\14\867.

Important: Notice (from the Date Modified) that editing the task and saving the changes modified all of the files in the task folder except *STARTUP.BAT*.



3.3.2.4 Create the MYTASKWK.BAT file

11. Create a copy of *TASKWORK.BAT*, named *MYTASKWK.BAT*, in the task folder.
12. Edit *MYTASKWK.BAT*.
13. Modify the block of comments at the top of the file. It's a good practice always to document your customization with comments (in the file you are customizing).
14. Add a line that reboots the system, and a line to force the system to wait until the reboot starts.

```

MYTASKWK.BAT - Notepad
File Edit Format View Help
@ECHO off
REM *****
REM * Remote Deployment Manager
REM * (C) Copyright IBM Corp. 2005 All rights reserved.
REM *****
REM * This file was originally created as TASKWORK.BAT by the
REM * RDM (w2003Std Adobe winzip) task. We modified it so
REM * that it installs Adobe Acrobat Reader 7.0.0, reboots,
REM * and then installs winZip 9.0 SR-1.
REM *****
TITLE "MYTASKWK.BAT running..."

ECHO Start processing MYTASKWK.bat...
c:\RDAGENT /L "Start processing MYTASKWK.bat..."
SET RDRASLEVEL=0
SET RDSTATUS=""
SET taskRC=0
Echo RDM_ADMIN=%USERNAME% >> c:\client.ini

if exist c:\doneapp.txt goto SKIP_APP
ECHO Preparing to unzip application images...
c:\RDAGENT /L "Preparing to unzip application images..."
call %TARGET%\task\unzipapp.bat
if %RDRASLEVEL%==1 goto FAIL

ECHO Preparing to install other images...
c:\RDAGENT /L "Preparing to install other images..."
call c:\rdmbin\appsinst.exe "%TARGET%\task\runapps.ini"
ECHO Preparing to delete temporary folders of images...
c:\RDAGENT /L "Preparing to delete temporary folders of images..."
C:\sleep.exe 20
call %TARGET%\task\delapp.bat
ECHO y > c:\doneapp.txt

c:\rdmbin\lcreboot.exe
C:\sleep.exe 60

:SKIP_APP

```

15. Add a block of lines that installs WinZip in a similar manner to the way the built-in task logic installs Adobe Reader. The lines are nearly identical, the only changes being the customized file names (doneapp2.txt, myunzapp.bat, myrunapp.ini, and mydelapp.bat) and the label (SKIP_APP2).

```

MYTASKWK.BAT - Notepad
File Edit Format View Help

c:\rdmbin\lcreboot.exe
C:\sleep.exe 60

:SKIP_APP

if exist c:\doneapp2.txt goto SKIP_APP2

ECHO Preparing to unzip application images...
c:\RDAGENT /L "Preparing to unzip application images..."
call %TARGET%\task\myunzapp.bat
if %RDRASLEVEL%==1 goto FAIL

ECHO Preparing to install other images...
c:\RDAGENT /L "Preparing to install other images..."
call c:\rdmbin\appsinst.exe "%TARGET%\task\myrunapp.ini"
ECHO Preparing to delete temporary folders of images...
c:\RDAGENT /L "Preparing to delete temporary folders of images..."
C:\sleep.exe 20
call %TARGET%\task\mydelapp.bat
ECHO y > c:\doneapp2.txt

:SKIP_APP2

if exist c:\donedrv.txt goto SKIP_DRV

```

16. Save your changes.

3.3.2.5 Create the USERAPP.BAT file

17. Create a copy of C:\MyTemp\866\COPYAPP.BAT, named USERAPP.BAT, in the task folder.
18. Modify the block of comments at the top of the file. It's a good practice always to document your customization with comments (in the file you are customizing).
19. Delete the first block of statements. These are the ones that download the Adobe image files.
20. Change "app1" to "uapp1" everywhere.
21. In the messages near the bottom of the file, change "copyApp.bat" to "USERAPP.bat" everywhere.
22. If appropriate, add more LCCUSTOM statements to replace variable names with their proper values. For example, you might add a statement like the following

```
LCCUSTOM C:\UAPP1\MYFILE.INI
```

to replace each occurrence of %appdir% in MYFILE.INI with its actual value c:\uapp1.

```

USERAPP.BAT - Notepad
File Edit Format View Help
@ECHO off
REM *****
REM * Remote Deployment Manager
REM * (C) Copyright IBM Corp. 2005 All rights reserved.
REM *****
REM * Copies a second block of applications that will be
REM * installed after a reboot done by the first block of
REM * applications.
REM *****

ECHO Starting USERAPP.bat processing to get application install files...
RDAGENT /L "Starting USERAPP.bat processing to get application install files..."

SET STATUS="MTFTP application file (04275144422531.zip)"
ECHO MTFTP application file (04275144422531.zip)
RDAGENT /L "MTFTP application file (04275144422531.zip)"
mtftp get %SERVER_IP% image\04275144422531.zip %TARGET%\temp\uapp1 -M
if errorlevel 1 goto FAIL
SET STATUS="MTFTP application launcher file (04275144422531.bat)"
if not exist %TARGET%\uapp1 md %TARGET%\uapp1
mtftp get %SERVER_IP% image\04275144422531.bat %TARGET%\uapp1\uapp1.bat
if errorlevel 1 goto FAIL
set appdir=c:\uapp1
LCCUSTOM c:\uapp1\uapp1.bat

ECHO Processing of USERAPP.bat completed
RDAGENT /L "Processing of USERAPP.bat completed"
goto END

:FAIL
ECHO Processing of USERAPP.bat failed
RDAGENT /L "Processing of USERAPP.bat failed"
SET RDRASLEVEL=1
goto END

:End

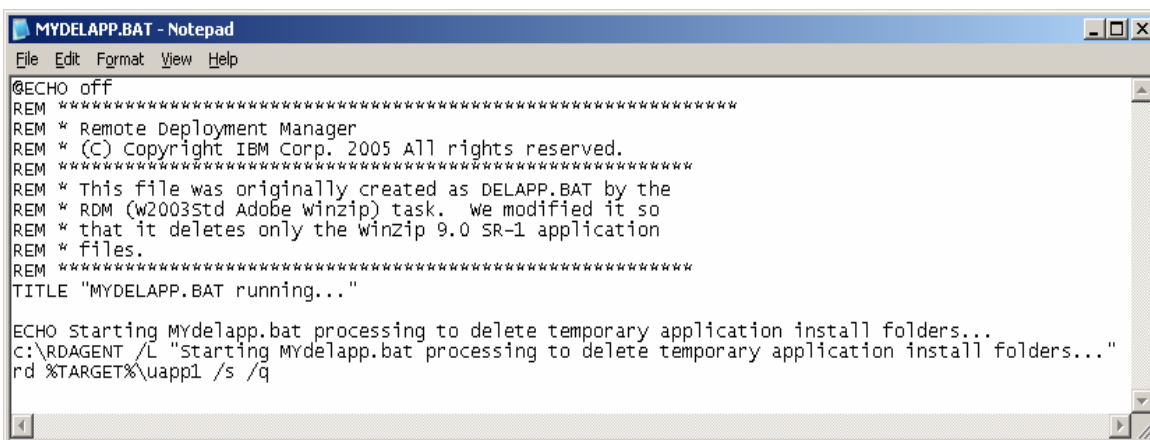
```

23. Save your changes.

3.3.2.6 Create the MYDELAPP.BAT file

24. Create a copy of C:\MyTemp\866\DELAPP.BAT, named MYDELAPP.BAT, in the task folder.
25. Modify the block of comments at the top of the file. It's a good practice always to document your customization with comments (in the file you are customizing).

26. Change "delapp.bat" to "MYdelapp.bat" everywhere.
27. Change "app1" to "uapp1" everywhere.



```

MYDELAPP.BAT - Notepad
File Edit Format View Help
@ECHO off
REM *****
REM * Remote Deployment Manager
REM * (C) Copyright IBM Corp. 2005 All rights reserved.
REM *****
REM * This file was originally created as DELAPP.BAT by the
REM * RDM (w2003Std Adobe winzip) task. We modified it so
REM * that it deletes only the Winzip 9.0 SR-1 application
REM * files.
REM *****
TITLE "MYDELAPP.BAT running..."

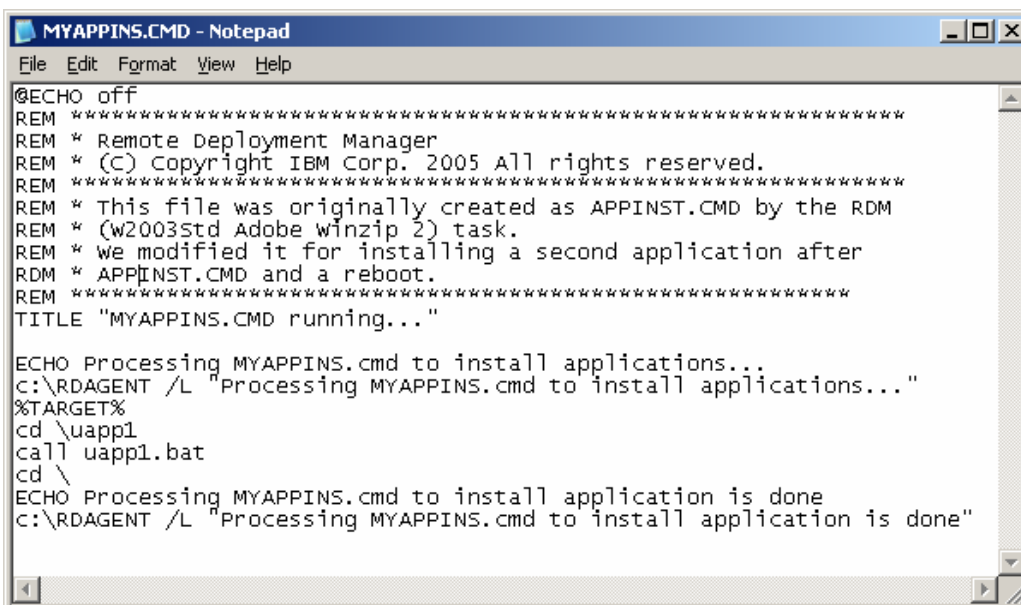
ECHO Starting MYdelapp.bat processing to delete temporary application install folders...
c:\RDAGENT /L "Starting MYdelapp.bat processing to delete temporary application install folders..."
rd %TARGET%\uapp1 /s /q

```

28. Save your changes.

3.3.2.7 Create the MYAPPINS.CMD file

29. Create a copy of C:\MyTemp\866\APPINST.CMD, named MYAPPINS.CMD, in the task folder.
30. Modify the block of comments at the top of the file. It's a good practice always to document your customization with comments (in the file you are customizing).
31. Delete the first block of statements (the ones referring to "app0"). These are the ones that download the Adobe image files.
32. Change "appinst.cmd" to "MYAPPINS.cmd" everywhere.
33. Change "app1" to "uapp1" everywhere.



```

MYAPPINS.CMD - Notepad
File Edit Format View Help
@ECHO off
REM *****
REM * Remote Deployment Manager
REM * (C) Copyright IBM Corp. 2005 All rights reserved.
REM *****
REM * This file was originally created as APPINST.CMD by the RDM
REM * (w2003Std Adobe winzip 2) task.
REM * We modified it for installing a second application after
RDM * APPINST.CMD and a reboot.
REM *****
TITLE "MYAPPINS.CMD running..."

ECHO Processing MYAPPINS.cmd to install applications...
c:\RDAGENT /L "Processing MYAPPINS.cmd to install applications..."
%TARGET%
cd \uapp1
call uapp1.bat
cd \
ECHO Processing MYAPPINS.cmd to install application is done
c:\RDAGENT /L "Processing MYAPPINS.cmd to install application is done"

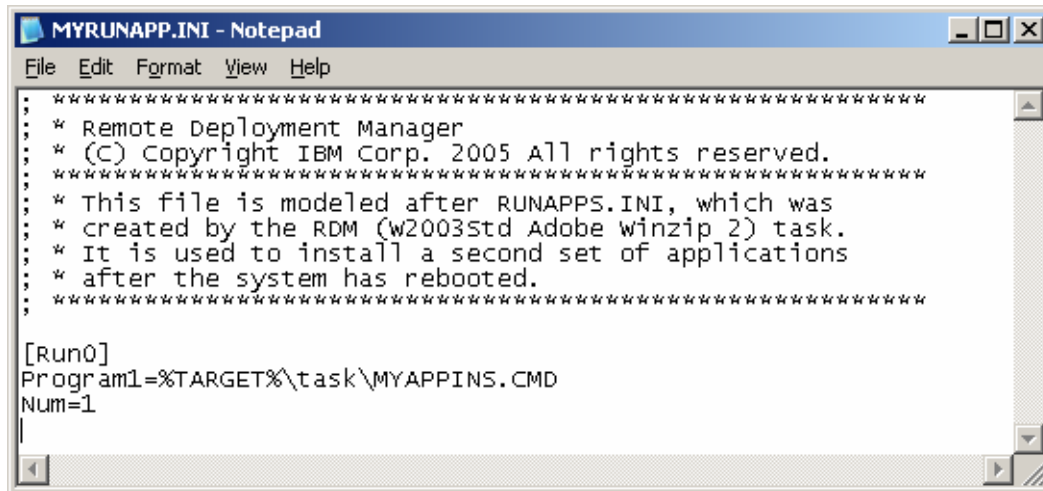
```

34. Save your changes.

3.3.2.8 Create the MYRUNAPP.INI file

35. Create a copy of C:\MyTemp\866\RUNAPPS.INI, named MYRUNAPP.INI, in the task folder.

36. Modify the block of comments at the top of the file. It's a good practice always to document your customization with comments (in the file you are customizing).
37. Change "appinst.cmd" to "MYAPPINS.CMD".

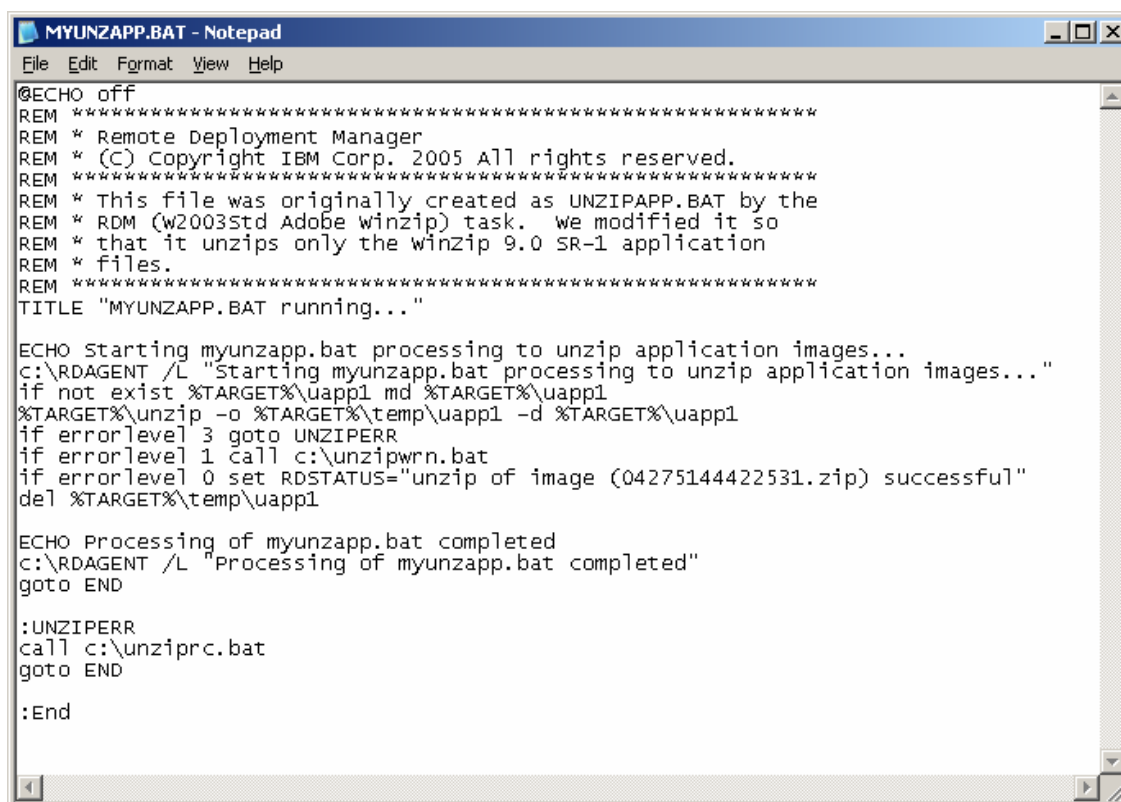


```
MYRUNAPP.INI - Notepad
File Edit Format View Help
:
: *****
: * Remote Deployment Manager
: * (C) Copyright IBM Corp. 2005 All rights reserved.
: *****
: * This file is modeled after RUNAPPS.INI, which was
: * created by the RDM (w2003std Adobe winzip 2) task.
: * It is used to install a second set of applications
: * after the system has rebooted.
: *****
:
: [Run0]
: Program1=%TARGET%\task\MYAPPINS.CMD
: Num=1
:
```

38. Save your changes.

3.3.2.9 Create the MYUNZAPP.BAT file

39. Create a copy of C:\MyTemp\866\UNZIPAPP.BAT, named MYUNZAPP.BAT, in the task folder.
40. Modify the block of comments at the top of the file. It's a good practice always to document your customization with comments (in the file you are customizing).
41. Delete the first block of statements (the ones referring to "app0"). These are the ones that unzip the Adobe image files.
42. Change "UNZIPAPP.BAT" to "MYUNZAPP.BAT" everywhere.
43. Change "app1" to "uapp1" everywhere.



```
MYUNZAPP.BAT - Notepad
File Edit Format View Help
@ECHO off
REM *****
REM * Remote Deployment Manager
REM * (C) Copyright IBM Corp. 2005 All rights reserved.
REM *****
REM * This file was originally created as UNZIPAPP.BAT by the
REM * RDM (w2003Std Adobe winzip) task. we modified it so
REM * that it unzips only the winzip 9.0 SR-1 application
REM * files.
REM *****
TITLE "MYUNZAPP.BAT running..."

ECHO Starting myunzapp.bat processing to unzip application images...
c:\RDAGENT /L "Starting myunzapp.bat processing to unzip application images..."
if not exist %TARGET%\uapp1 md %TARGET%\uapp1
%TARGET%\unzip -o %TARGET%\temp\uapp1 -d %TARGET%\uapp1
if errorlevel 3 goto UNZIPERR
if errorlevel 1 call c:\unzipwrn.bat
if errorlevel 0 set RDSTATUS="unzip of image (04275144422531.zip) successful"
del %TARGET%\temp\uapp1

ECHO Processing of myunzapp.bat completed
c:\RDAGENT /L "Processing of myunzapp.bat completed"
goto END

:UNZIPERR
call c:\unziprc.bat
goto END

:End
```

44. Save your changes.

3.3.2.10 Modify the STARTUP.BAT file

In the above sections, we created modified, renamed versions of several batch files. We changed their file names to prevent RDM from overwriting our changes (which would happen, if we used the original files, whenever you edit the task and then select the *OK* button). Now we have to tie the renamed files into the task logic. We do that in the STARTUP.BAT file.

45. Edit the STARTUP.BAT file in the task folder.
46. Add a comment to describe your changes. It's a good practice always to document your customization with comments (in the file you are customizing).
47. Change "taskwork.bat" to "MYtaskwk.bat".

```

startup.bat - Notepad
File Edit Format View Help
@ECHO off
REM *****
REM * Remote Deployment Manager
REM * (C) Copyright IBM Corp. 2005 All rights reserved.
REM *****
REM * This file is used immediately after windows completed the
REM * unattended operating-system install. The task folder contains
REM * this file, which is copied from the template folder when RDM
RDM * creates the task. Subsequent editing of the task will not
REM * overwrite this file.
REM * Feb 25, 2005 - changed to run MYtaskkwk.bat, which contains
REM * task-logic modifications.
REM *****

ECHO Start processing startup.bat...
c:\RDAGENT /L "RDAGEN000I start processing startup.bat..."
TITLE "RDM STARTUP.BAT - processing, please wait..."
C:\sleep.exe 40
call c:\setpath.bat

if exist c:\donentfs.txt goto SKIP_NTFS
ECHO Converting drive(s) to NTFS...
c:\RDAGENT /L "RDAGEN000I Converting drive(s) to NTFS..."
call c:\rdmbin\setNTFS.bat
ECHO y > c:\donentfs.txt

:SKIP_NTFS

REM *****
REM * This block of instruction should not be moved or removed
call %TARGET%\MYtaskkwk.bat
if %taskRC%==1 goto FAIL
REM *****

ECHO Running post-install batch file that might install driver e.g. ASM...
c:\RDAGENT /L "RDAGEN000I Running post-install batch file that might install driver e.g. ASM..."
call c:\postinst.bat 1
ECHO Cleaning files and folder
c:\RDAGENT /L "RDAGEN000I Cleaning files and folder"
del c:\postinst.bat
c:\cleanup.exe
ECHO startup.bat processing is done
c:\RDAGENT /L "RDAGEN000I startup.bat processing is done"
ECHO Calling RDMAGENT.bat for possible custom work
c:\RDAGENT /L "RDAGEN000I Calling RDMAGENT.bat for possible custom work"
call c:\RDMAGENT.bat
goto END

:FAIL
C:\RDAGENT /L "FATAL ERROR ENCOUNTERED - HALT PROCESSING OF STARTUP.BAT"
SET RDRASLEVEL = 1
SET RDSTATUS = "RDAGEN099E windows startup encountered fatal error."
C:\RDAGENT.EXE /fs

:END

```

48. Save your changes.

3.3.2.11 Create the DOS71XU system environment

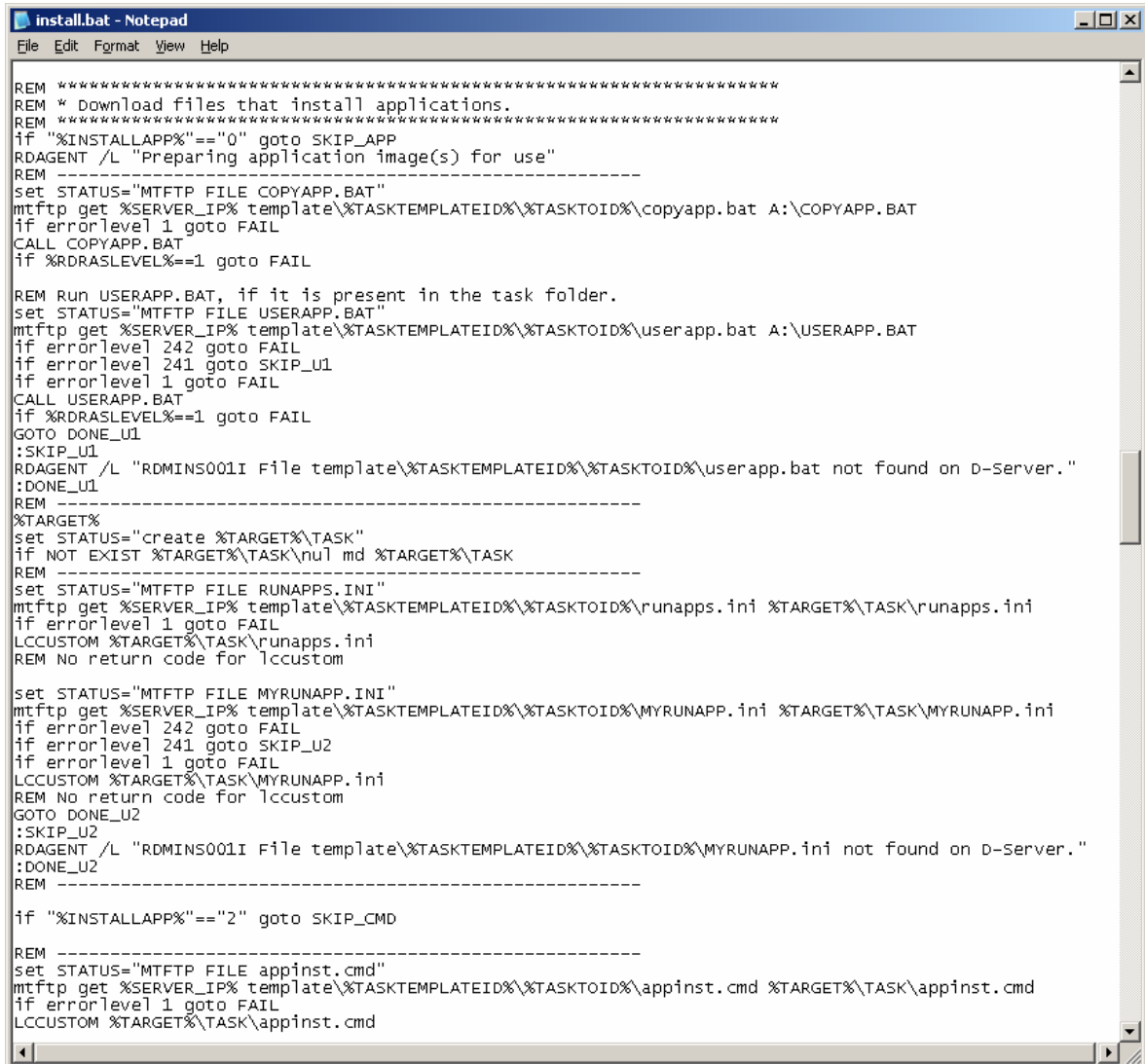
The built-in RDM logic downloads all application install files under DOS 7.1, in the INSTALL.BAT file. We will mimic that logic to download the install files for the applications that RDM will install after a reboot. Since INSTALL.BAT is part of the DOS71X system environment, we will create a similar system environment that contains a modified version of INSTALL.BAT.

We create a new system environment in order to protect our tasks from having our customizations erased during a future RDM upgrade to the DOS71X system environment.

49. Using Windows Explorer, drag the \Program Files\IBM\RDM\local\env\71x directory to make a new directory named \Program Files\IBM\RDM\local\env\Copy of 71x.
50. Rename that directory to \Program Files\IBM\RDM\local\env\71xu.
51. Right click on the paperclip icon below, and save the imbedded file to \Program Files\IBM\RDM\local\env\71xu\INSTALL.BAT (or use Notepad or your favorite editor to modify the file).

This file contains additions for the second set of applications that mimic the existing logic used for the built-in applications. For example, we added a block of code to process USERAPP.BAT in a similar way to the existing logic for COPYAPP.BAT. We did this for all the application-related files that INSTALL.BAT handles.

Notice that these changes will also work for *Windows Native Install* tasks whose application install uses only the built-in RDM logic, and it will work for *Windows Native Install* tasks that do not install applications. If the user files (USERAPP.BAT, MYRUNAPP.INI, etc.) are not present, the processing continues normally.



```

install.bat - Notepad
File Edit Format View Help
REM *****
REM * Download files that install applications.
REM *****
if "%INSTALLAPP%"=="0" goto SKIP_APP
RDAGENT /L "Preparing application image(s) for use"
REM -----
set STATUS="MTFTP FILE COPYAPP.BAT"
mtftp get %SERVER_IP% template\%TASKTEMPLATEID%\%TASKTOID%\copyapp.bat A:\COPYAPP.BAT
if errorlevel 1 goto FAIL
CALL COPYAPP.BAT
if %RDRASLEVEL%==1 goto FAIL

REM Run USERAPP.BAT, if it is present in the task folder.
set STATUS="MTFTP FILE USERAPP.BAT"
mtftp get %SERVER_IP% template\%TASKTEMPLATEID%\%TASKTOID%\userapp.bat A:\USERAPP.BAT
if errorlevel 242 goto FAIL
if errorlevel 241 goto SKIP_U1
if errorlevel 1 goto FAIL
CALL USERAPP.BAT
if %RDRASLEVEL%==1 goto FAIL
GOTO DONE_U1
:SKIP_U1
RDAGENT /L "RDMINS001I File template\%TASKTEMPLATEID%\%TASKTOID%\userapp.bat not found on D-Server."
:DONE_U1
REM -----
%TARGET%
set STATUS="create %TARGET%\TASK"
if NOT EXIST %TARGET%\TASK\nul md %TARGET%\TASK
REM -----
set STATUS="MTFTP FILE RUNAPPS.INI"
mtftp get %SERVER_IP% template\%TASKTEMPLATEID%\%TASKTOID%\runapps.ini %TARGET%\TASK\runapps.ini
if errorlevel 1 goto FAIL
LCCUSTOM %TARGET%\TASK\runapps.ini
REM No return code for lccustom

set STATUS="MTFTP FILE MYRUNAPP.INI"
mtftp get %SERVER_IP% template\%TASKTEMPLATEID%\%TASKTOID%\MYRUNAPP.ini %TARGET%\TASK\MYRUNAPP.ini
if errorlevel 242 goto FAIL
if errorlevel 241 goto SKIP_U2
if errorlevel 1 goto FAIL
LCCUSTOM %TARGET%\TASK\MYRUNAPP.ini
REM No return code for lccustom
GOTO DONE_U2
:SKIP_U2
RDAGENT /L "RDMINS001I File template\%TASKTEMPLATEID%\%TASKTOID%\MYRUNAPP.ini not found on D-Server."
:DONE_U2
REM -----

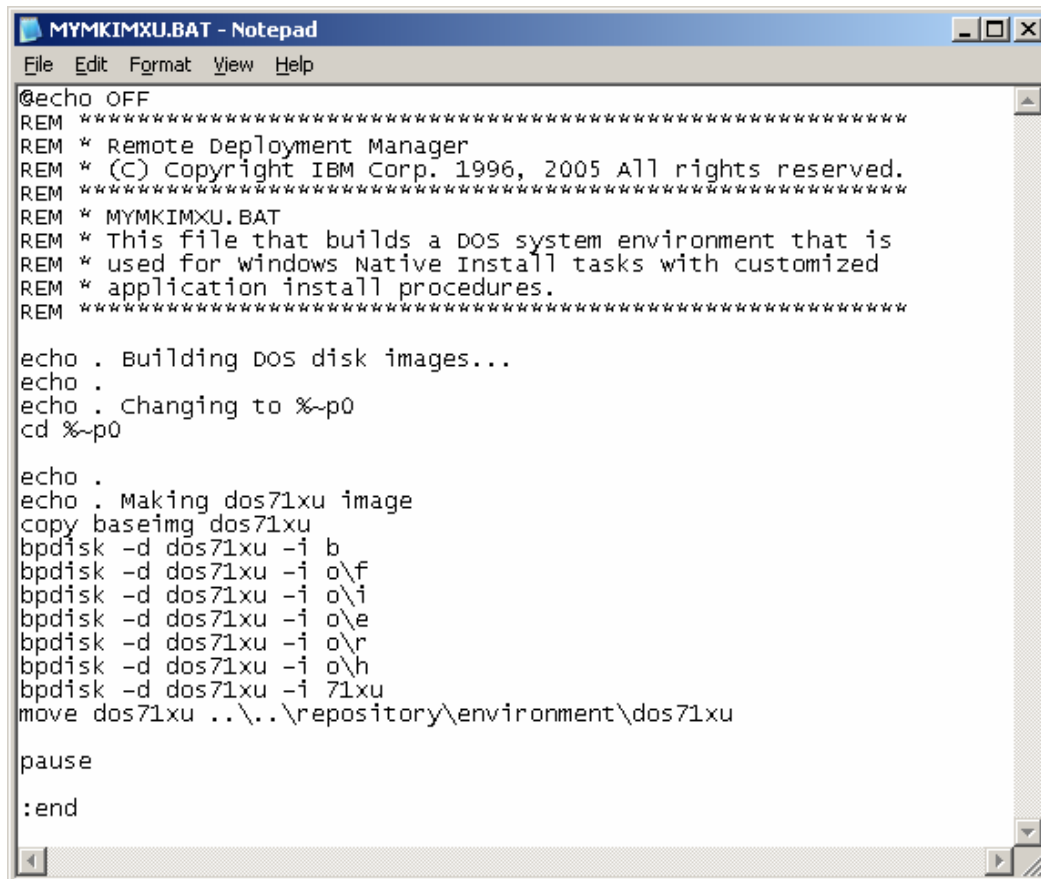
if "%INSTALLAPP%"=="2" goto SKIP_CMD

REM -----
set STATUS="MTFTP FILE appinst.cmd"
mtftp get %SERVER_IP% template\%TASKTEMPLATEID%\%TASKTOID%\appinst.cmd %TARGET%\TASK\appinst.cmd
if errorlevel 1 goto FAIL
LCCUSTOM %TARGET%\TASK\appinst.cmd

```

52. Right click on the paperclip icon below, and save the imbedded file to \Program Files\IBM\RDM\local\env\MYMKIMXU.BAT (or use Notepad or your favorite editor to create the file).

This file contains the following statements:



```

MYMKIMXU.BAT - Notepad
File Edit Format View Help
@echo OFF
REM *****
REM * Remote Deployment Manager
REM * (C) Copyright IBM Corp. 1996, 2005 All rights reserved.
REM *****
REM * MYMKIMXU.BAT
REM * This file that builds a DOS system environment that is
REM * used for windows Native Install tasks with customized
REM * application install procedures.
REM *****

echo . Building DOS disk images...
echo .
echo . Changing to %~p0
cd %~p0

echo .
echo . Making dos71xu image
copy baseimg dos71xu
bpdisk -d dos71xu -i b
bpdisk -d dos71xu -i o\f
bpdisk -d dos71xu -i o\i
bpdisk -d dos71xu -i o\e
bpdisk -d dos71xu -i o\r
bpdisk -d dos71xu -i o\h
bpdisk -d dos71xu -i 71xu
move dos71xu ..\..\repository\environment\dos71xu

pause

:end

```

53. This file creates the new DOS system environment. It first copies the DOS 7.1 kernel files (file name baseimg) into DOS71XU (the file name of the new DOS system environment). Then it uses BPDISK.EXE to add other files into DOS71XU. Finally, it moves the finished DOS71XU file to the RDM master repository.
54. Note that DOS71XU is the name that is used in the task's command list (see step 8 above).
55. Execute the \Program Files\IBM\RDM\local\env\MYMKIMXU.BAT file. After the batch file pauses, make sure that there are no error messages, before closing the window.

3.3.3 Using RDM's command list

You can explicitly add an application install to an RDM *Windows Native Install* CommandList file. One reason for doing this might be to encapsulate all of the application install logic into a single place.

Note: We can use a similar procedure in an RDM *Windows Clone Install* task (see section 4.2.1 on page 64 for a complete description of the procedure).

Here is the general procedure:

1. Create a RDM *Windows Native Install* application image for each application, using the procedures described in section 3.2 on page 26.
2. Edit the task, and use the Command List Editor Wizard to add statements that download and unzip the images.
3. Add a statement to the command list that installs the application.
4. Add a statement to erase the directory from which you installed the application, if appropriate.
5. Add a statement that reboots the system after installing the application.

3.3.4 Using *CMDLINES.TXT*

You can explicitly add an application install to any unattended Windows install using the *CMDLINES.TXT* file. This is a standard user procedure for Windows install, and you can incorporate it into an RDM *Windows Native Install* task. One reason for doing this might be to reuse application-install logic that you had already prepared prior to starting to use RDM.

You can get Microsoft documentation that describes the use of the *CMDLINES.TXT* file. All of the appropriate files are available in RDM (see section 6 above for details). You just need to modify the files as needed, in a way that will preclude RDM from overwriting your modifications and in a way that will prevent an RDM update from overwriting your modifications.

The details are left as an exercise for the reader.

3.3.5 Integrating updates or hotfixes into your operating-system image

This technique, also called “slipstreaming”, involves installing the updates into a copy of the Microsoft Windows CD, and then using that updated copy to create the RDM *Windows Native Install* operating-system image. The detailed procedure for Windows 2003 is available on this web page:

<http://www.microsoft.com/technet/security/topics/patchmanagement/hfdeploy.mspx>

Here is a high-level summary of how to do this with RDM:

1. On your RDM console computer, make a copy of the I386 directory from your Windows CD.
2. Modify that I386 copy using the detailed procedure from the above web page.
3. Use that modified I386 copy as input when you create the RDM *Windows Native Install* operating-system image.

This procedure is a bit cumbersome to set up, but it makes the RDM *Windows Native Install* task run faster, because it installs the updates as part of the operating-system install (instead of doing it after the operating-system install completes).

Note that you can use a similar integrating procedure for adding a Windows service pack as part of the RDM *Windows Native Install* operating-system image.

4. Windows Clone Install

4.1 Internal task logic

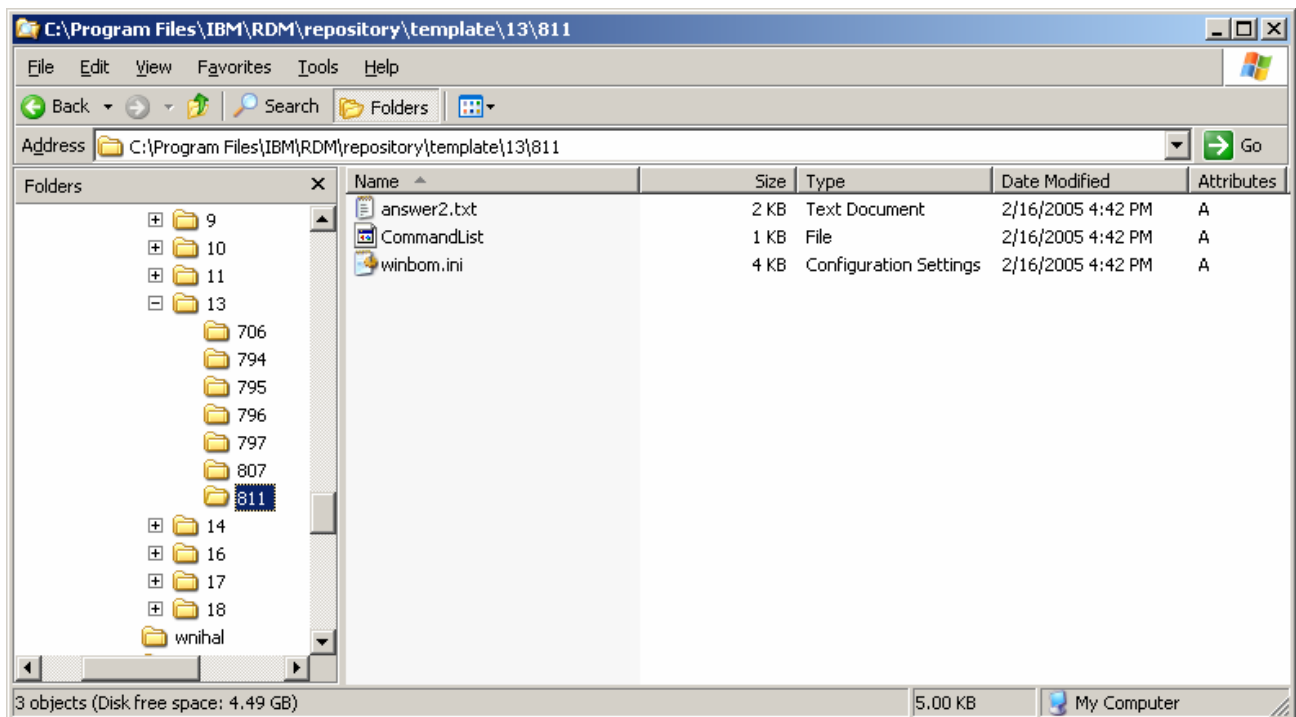
To customize a *Windows Clone Install* task application install, it will be helpful to understand how this task works. In this section, we will explore a typical *Windows Clone Install* task that installs Windows Server 2003 Standard. Assume that we have completed the first procedure outlined in section 2.3 above.

4.1.1 Find the task folder

Use the technique from section 3.1.3 on page 10.

4.1.2 Explore the task logic

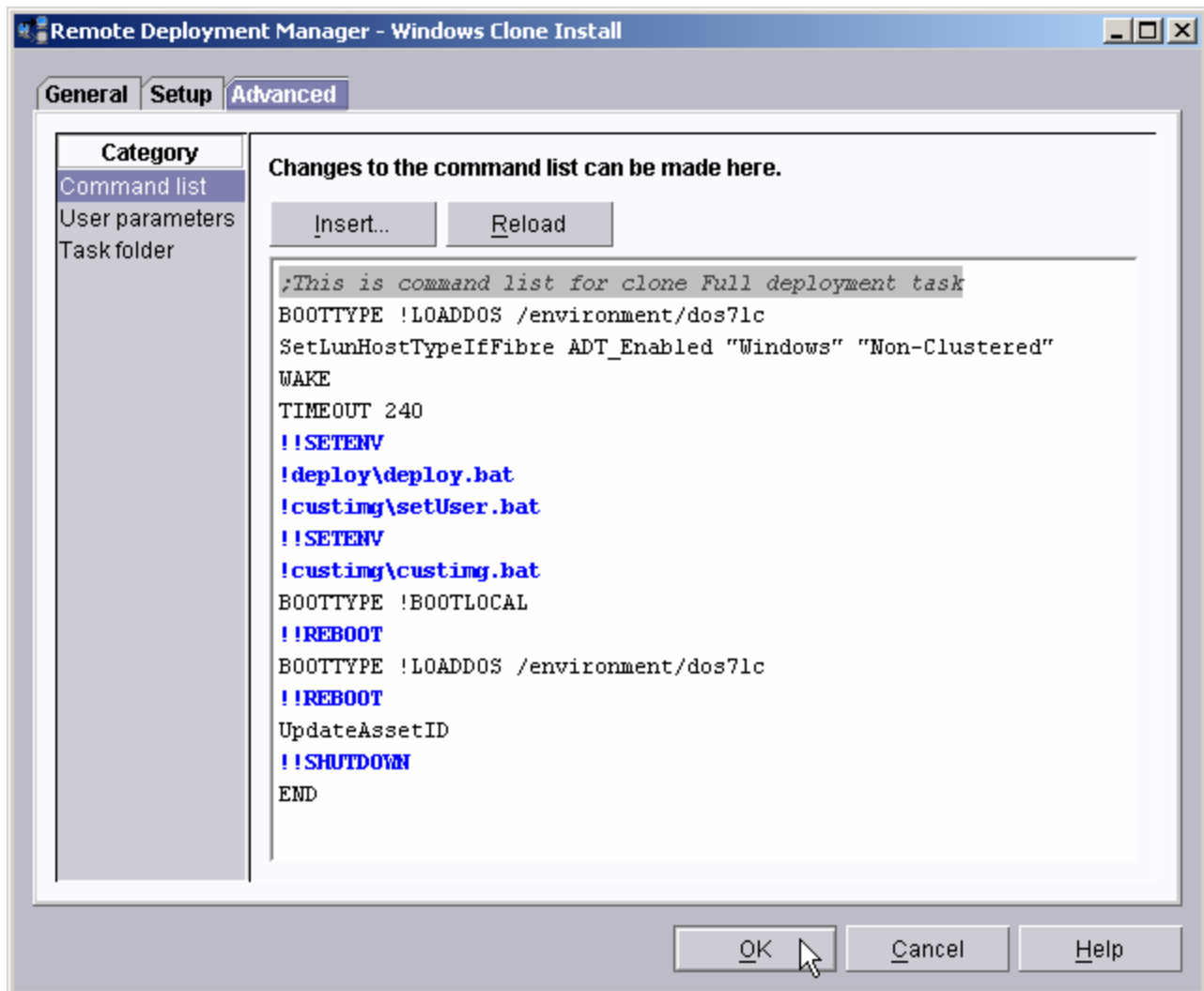
Open Windows Explorer to view the files in the task folder.



We'll briefly describe and view the contents of each file.

4.1.2.1 CommandList

See section 3.1.4.1 above for a generic description of a CommandList file.



Now we'll consider each command, in the context of this task.

1. **BOOTTYPE !LOADDOS /environment/dos71c** – The RDM server will force the target system to boot the DOS71C system environment the next time it does a PXE network boot.
2. **SetLunHostTypeIfFibre ADT_Enabled "Windows" "Non-Clustered"** – If Windows is being deployed to a FASTt fibre boot drive and RDM remote storage has been enabled via storage/switch entries in the RDM Network Storage tool, this command will set the host type of the FASTt fibre boot drive to Windows Non-Clustered with Automatic Data Transfer (ADT) enabled.
3. **WAKE** – The RDM server will tell the RDM Deployment Server (D-Server) to power on the target system. The target system will download and run the RDM Bootstrap Loader program, and it will eventually boot the DOS71C system environment.
4. **TIMEOUT 240** – This command sets the maximum run time for this task to 240 minutes. The standard default value is 120 minutes.
 Note that a typical *Windows Clone Install* task takes much less time (depending on the size of the image and network speed, perhaps 15 to 20 minutes).
5. **!!SETENV** – The RDAGENT.EXE program will initialize the task's environment variables on the target system. That is, it will run several statements of the form SET NAME=VALUE under DOS 7.1 on the target system.

6. **!deploy\deploy.bat** – The RDAGENT.EXE program will run the DEPLOY.BAT file on the target system. This batch file removes all existing partitions the hard disk drive, and it uses the DeployCenter imaging tool to download the operating-system image (see section 2.3.1 above).
7. **!custimg\setUser.bat** – This batch file sets default values for certain parameters, because in some cases they may not be set by the task logic.
8. **!!SETENV** – See step 5 above.
9. **!custimg\custimg.bat** – This batch file prepares the target system to run the Microsoft mini setup program. It copies several files to the C: drive. These files contain the information needed by mini setup.
10. **BOOTTYPE !BOOTLOCAL** – The RDM server will force the target system to boot the local hard drive the next time it does a PXE network boot.
11. **!!reboot** – This reboots the system. Because it is a warm reboot, it will be the same kind of boot that it did in step 3 above (i.e., a PXE network boot). Since the BOOTTYPE from step 10 above is now in effect, the target system will download and run the RDM Bootstrap Loader program, and it will eventually boot the local hard drive.

The target system will automatically run Microsoft mini setup, in unattended mode, to personalize the system. It uses the information in the ANSWER2.TXT file as input. Mini setup forces the system to reboot.

Since the BOOTTYPE from step 10 above is still in effect, the target system will download and run the RDM Bootstrap Loader program, and it will eventually boot the local hard drive again.

Now, because of the earlier setup done in step 9 above, the target system runs the PQAGENT.BAT file. This file contains an infinite loop in which it contacts the RDM server asking for another command to run. In a typical Windows Clone Install task, the next command (see step 13 below) will cause the system to reboot.

Note: This is the place where we will put our customized application install logic.

12. **BOOTTYPE !LOADDOS /environment/dos71c** – The RDM server will force the target system to boot the DOS71C system environment the next time it does a PXE network boot.
13. **!!reboot** – This reboots the system. Because it is a warm reboot, it will be the same kind of boot that it did in step 3 above (i.e., a PXE network boot). Since the BOOTTYPE from step 12 above is now in effect, the target system will download and run the RDM Bootstrap Loader program, and it will eventually boot the DOS71C system environment.

The purpose of this reboot is so that the target system can do its final handshake with the RDM server.
14. **UpdateAssetID** – This causes the RDM Server to initiate an update of 2 fields on the Asset ID EEPROM management chip, for systems (i.e., some IBM NetVista, ThinkCentre, and ThinkPad systems) that have this chip. It writes the first 16 characters of the RDM task name in the IMAGE field, and it writes the current date in the IMAGEDATE field.
15. **!!SHUTDOWN** – This powers off the system.
16. **END** – This tells the RDM server that the task is complete.

4.1.2.2 ANSWER2.TXT

This file is used by Microsoft mini setup to personalize the target system. In other contexts, this file is often named SYSPREP.INI.

It is possible for you to modify this file. For example, you might want to change the resolution in the [Display] section to 1024 by 768.

Note the use of environment variables in this file. RDM replaces these with the appropriate values for each target system.



```
answer2.txt - Notepad
File Edit Format View Help

[Unattended]
ExtendOEMPartition=0
InstallFilesPath=C:\sysprep\i386
OemSkipEula=yes
;OemPnpDriversPath = drv\video; drv\net

[UserData]
OrgName="%CompanyName%"
ProductKey="%CDKey%"
ComputerName="%ComputerName%"
FullName="%UserName%"

[GuiUnattended]
EncryptedAdminPassword=No
OEMSkipRegional=1
OemSkipwelcome=1
TimeZone="%TimeZone%"
AdminPassword=*
AutoLogon=Yes
AutoLogonCount=1

[RegionalSettings]
Language="%LocaleLanguage%"
LanguageGroup="%LocaleLanguageGroup%"

[Display]
BitsPerPel=16
VRefresh=70
XResolution=800
YResolution=600

[LicenseFilePrintData]
AutoUsers=%LicenseCount%
AutoMode=PERSERVER
```

4.1.2.3 WINBOM.INI

This file is used only when you ran SYSPREP.EXE and selected the Factory button (see step 4 in section 2.3.1 on page 8).

A screenshot of a Notepad window titled "winbom.ini - Notepad". The window contains the following text:

```
;winbom.ini
[Factory]
AutoDetectNetwork=Yes
DoDeviceIDScanOnError=Yes
FactoryComputerName="%ComputerName%"
Logfile=C:\winbom.log
Logging=Yes
LogLevel=2
LogPerf=Yes
OptimizeShell=Yes
Password=*
RebootAfterComputerName=No
Reseat=Reboot
ReseatMode=Mini
;winbomType = Factory
;UserName = myDomain\myUser

[ComputerSettings]
AutoLogon = Yes
;AuditAdminAutoLogon = Yes
;DisplayRefresh = 75
DisplayResolution="1024x768x32"
ExtendPartition = 0
FontSmoothing=Standard
SourcePath=C:\i386

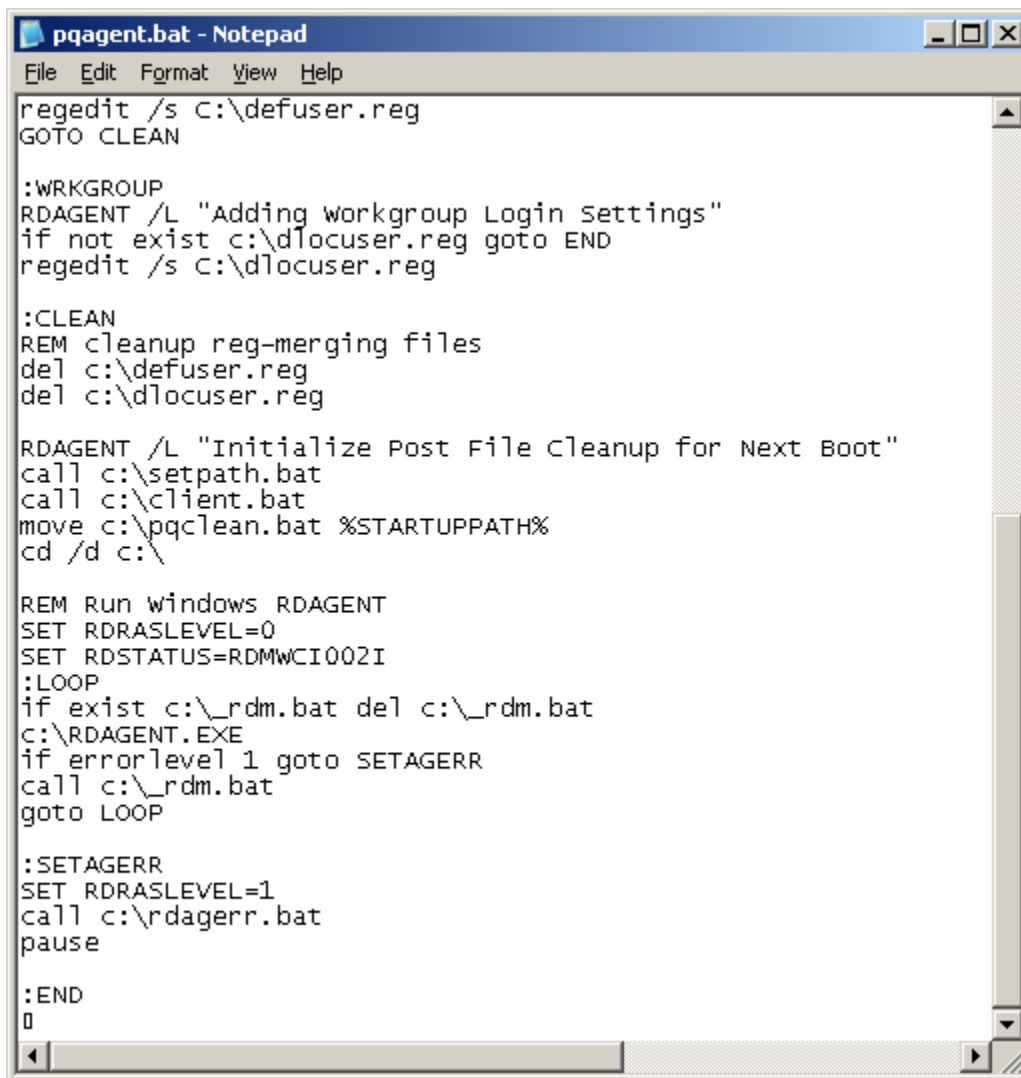
[Shell]
;DefaultClientStartMenuInternet =
;DefaultClientMail =
;DefaultClientMedia =
;DefaultClientIM =
;DefaultClientJavaVM =

;runs synchronously
[OEMRunOnce]
```

4.1.2.4 PQAGENT.BAT

This is the file that contains the loop that continually asks the RDM server for the next command.

The C:\RDAGENT.EXE line results in a download of a batch file *_rdm.bat* into the current directory (which is currently assumed to be C:\). The next line, call c:_rdm.bat, runs the next command from the task's CommandList file.



```
pqagent.bat - Notepad
File Edit Format View Help
regedit /s C:\defuser.reg
GOTO CLEAN

:WRKGROUP
RDAGENT /L "Adding workgroup Login Settings"
if not exist c:\dlocuser.reg goto END
regedit /s c:\dlocuser.reg

:CLEAN
REM cleanup reg-merging files
del c:\defuser.reg
del c:\dlocuser.reg

RDAGENT /L "Initialize Post File Cleanup for Next Boot"
call c:\setpath.bat
call c:\client.bat
move c:\pqclean.bat %STARTUPPATH%
cd /d c:\

REM Run windows RDAGENT
SET RDRASLEVEL=0
SET RDSTATUS=RDMwCI002I
:LOOP
if exist c:\_rdm.bat del c:\_rdm.bat
c:\RDAGENT.EXE
if errorlevel 1 goto SETAGERR
call c:\_rdm.bat
goto LOOP

:SETAGERR
SET RDRASLEVEL=1
call c:\rdagerr.bat
pause

:END
□
```

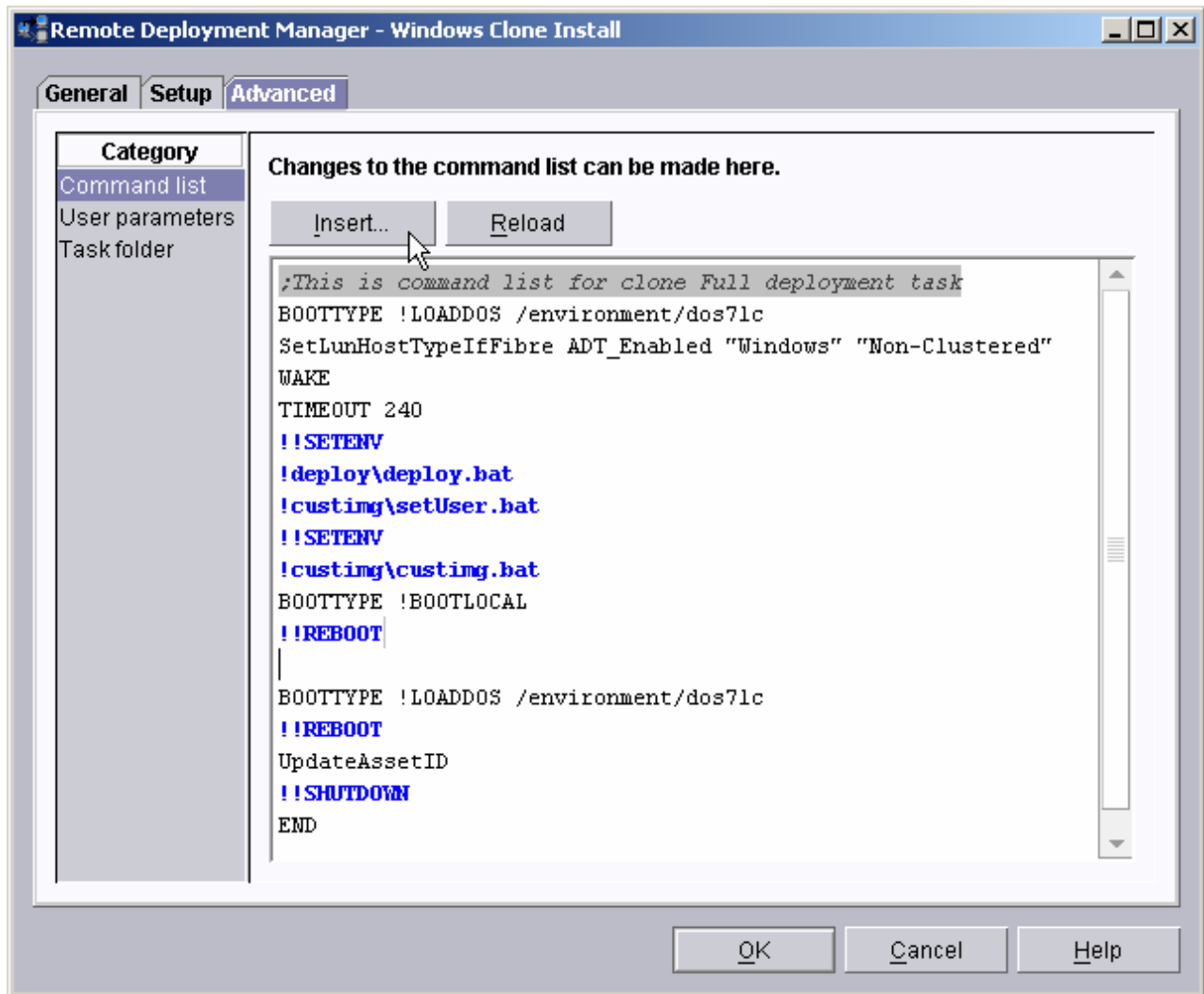
4.2 Installing applications

It is possible to use *Windows Native Install* application images in *Windows Clone Install* tasks. We will show how to modify a standard *Windows Clone Install* task to add application installs. We will take advantage of the RDAGENT loop in the PQAGENT.BAT file to add several commands that will install the applications.

A scenario where this is desirable is when your current *Windows Clone Install* task installs Windows plus a set of applications, but you have some new applications that you want to add to the task. Instead of rebuilding the task from scratch, using the procedure in section 2.3.1 on page 8, you can just add the applications to your existing task. This will save quite a bit of preparation work.

4.2.1 Procedure

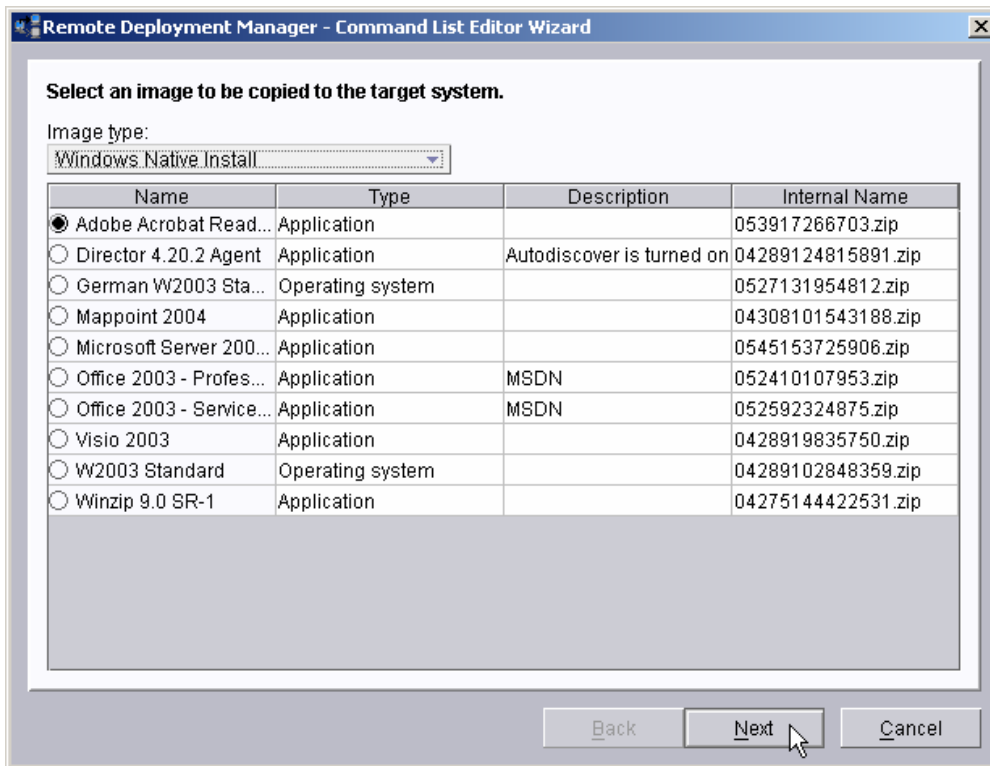
Our technique will be to add the application installs to the CommandList. We will insert the commands that do those application installs where the blank line is shown in this picture. This is the point in the task processing at which PQAGENT.BAT runs (see item 11 on page 61 in section 4.1.2.1 above).



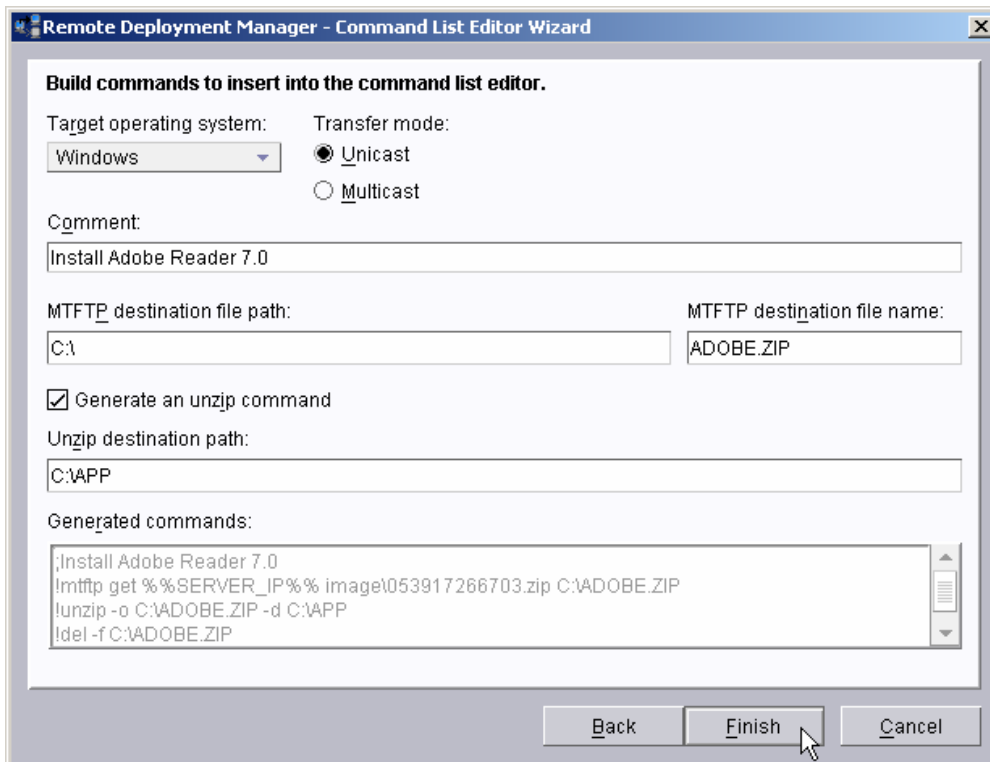
Important: One caveat to consider here is that PQAGENT.BAT only runs once. So we have to be able to install all of our extra applications at this point, before the system reboots again. If you have 2 applications whose installs must be separated by a reboot, you cannot install them as described herein.

Here is the general procedure:

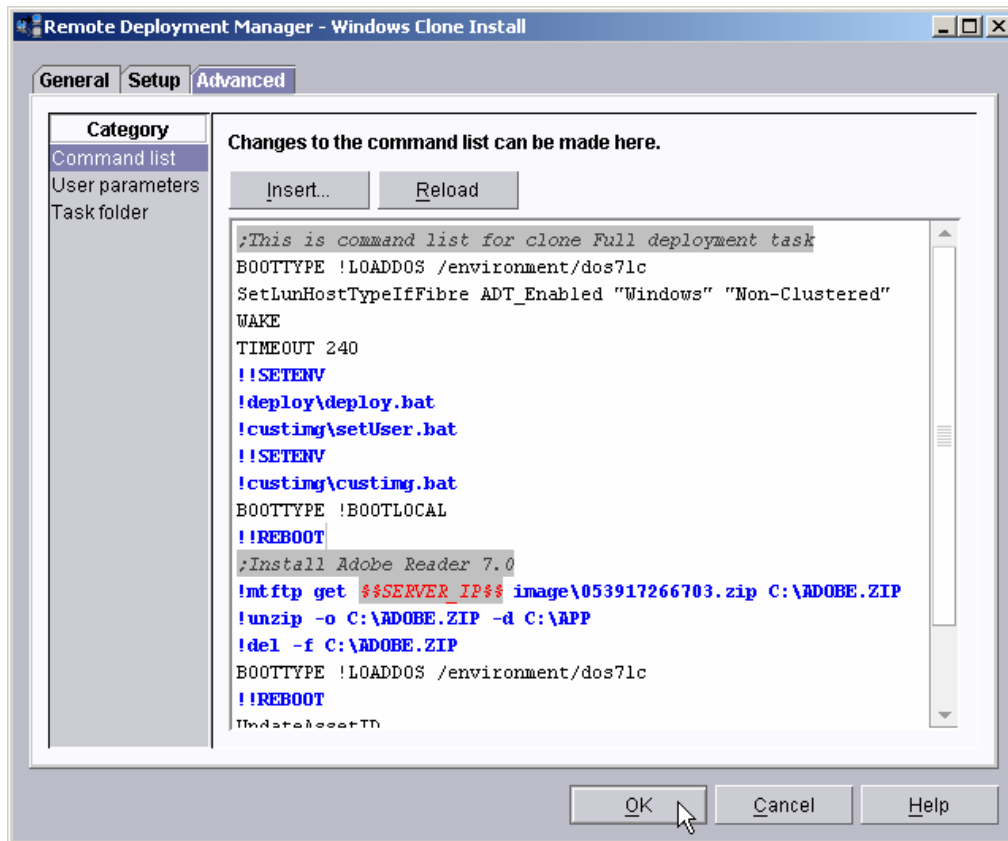
1. Edit your existing *Windows Clone Install* task. Select the *Advanced* page to display the CommandList.
2. Insert a blank line at the position shown in the above picture. Then press the *Insert* button to display the *Command List Editor Wizard* window.



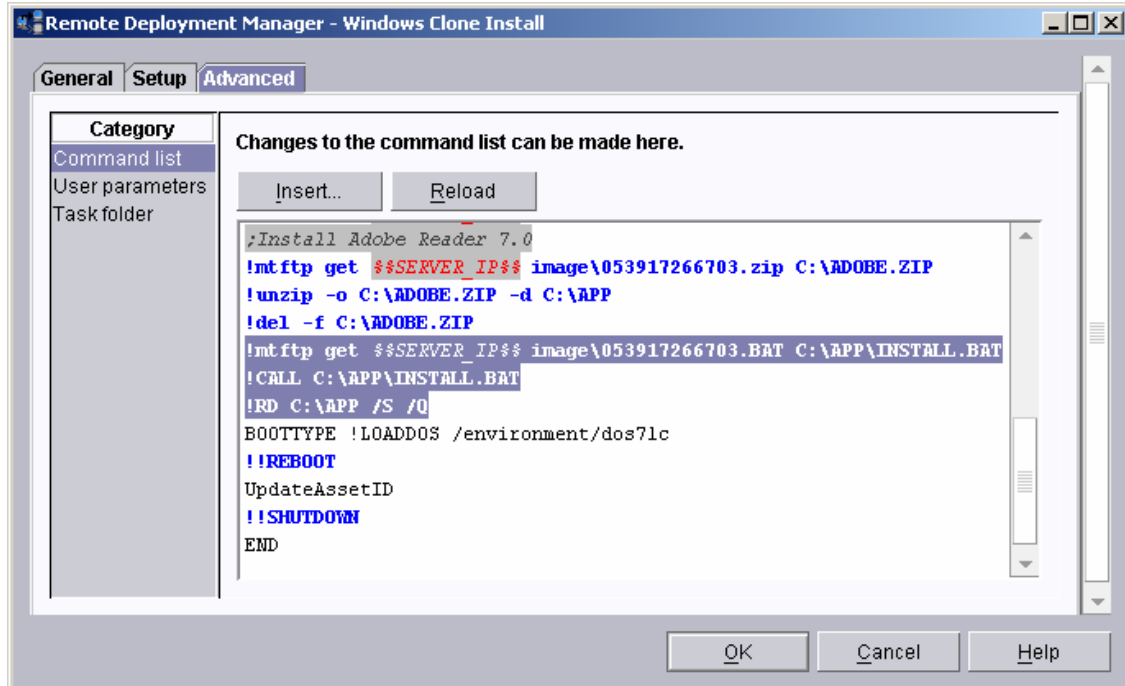
3. Select *Windows Native Install* in the drop-down list, and then select the application that you want to install. Then select the *Next* button to display the wizard's second page.



4. Enter data as shown above. Then select the *Finish* button. This will insert the generated commands from the wizard window into the CommandList.



5. Now add three more commands, as shown (selected) in the picture below.

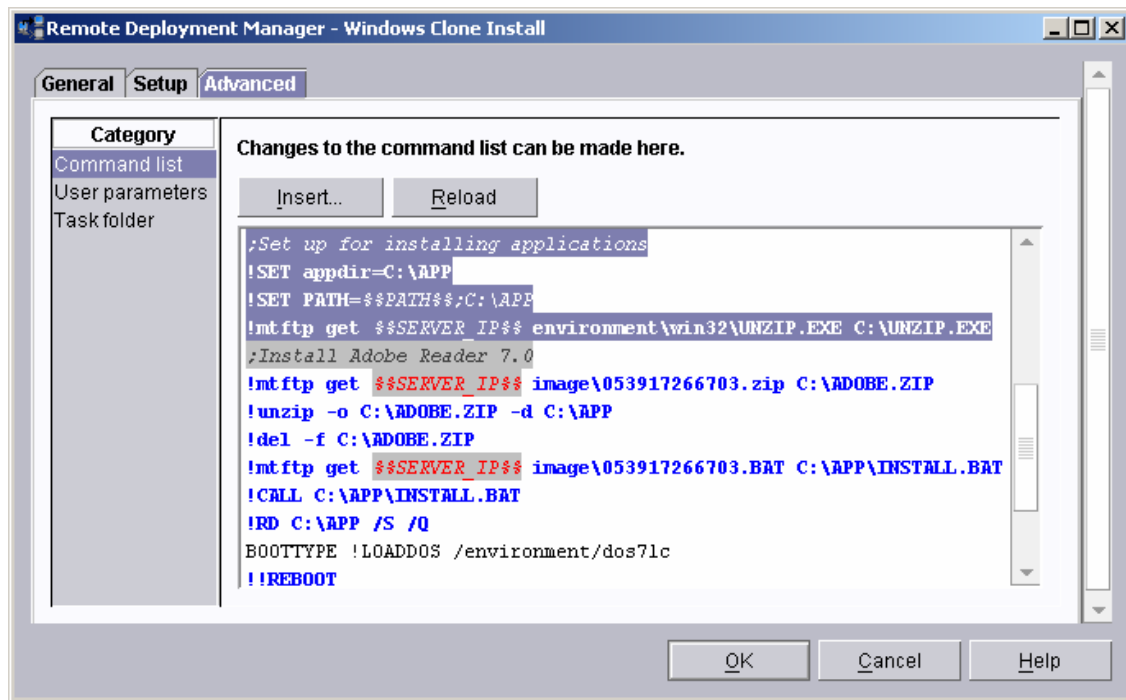


The `!mtftp get %%SERVER_IP%% image\<<name>.BAT C:\APP\INSTALL.BAT` command will download the batch file that is used to install the application. It has the same file name as the application's zip file, but with an extension of BAT.

The `!CALL C:\APP\INSTALL.BAT` will run the batch file and install the application.

The `!RD C:\APP /S /Q` command will remove the application's install directory, which is no longer needed.

- As written, the application install commands will not work properly. So we add 3 more commands, as shown (selected) in the picture below.



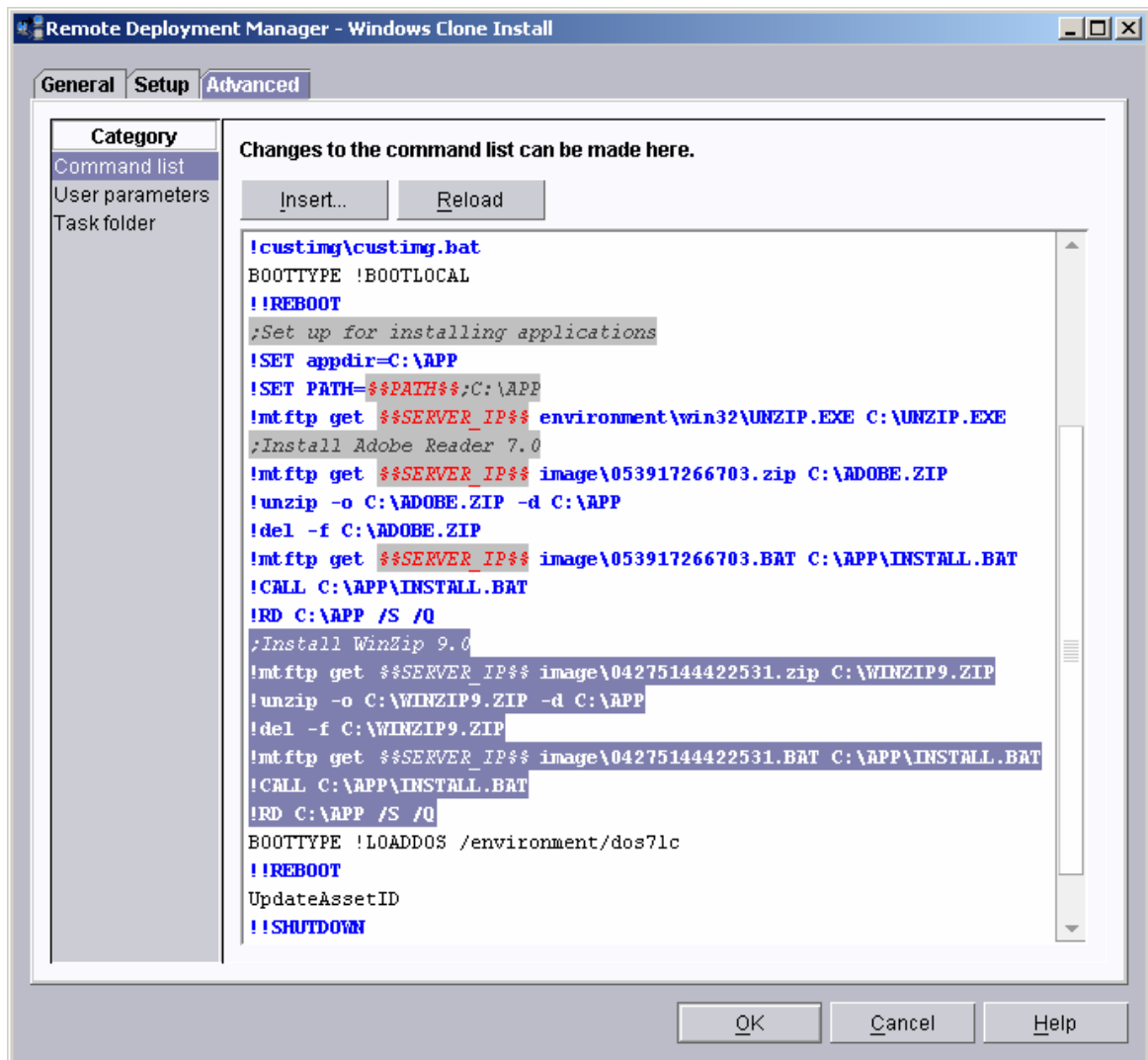
The `!SET appdir=C:\APP` command sets an environment variable that may be needed in some applications' batch files.

The `!SET PATH=%%PATH%%;C:\APP` command is needed to ensure that the executable inside `INSTALL.BAT` can be found.

The `!mtftp get %%SERVER_IP%% environment\win32\UNZIP.EXE C:\UNZIP.EXE` command downloads the `UNZIP.EXE` program which is needed to unzip the application image file.

At this point, if we ran the task, it would install the task's Windows image and then install Adobe Reader. However, we will add a second application.

- Insert a blank line right before the `BOOTTYPE` command, and repeat steps 1 through 6 above, selecting the WinZip 9.0 SR-1 application. The resulting command list will look like this:



The selected lines will install WinZip.

At this point, you can run the task, and it will install Windows (plus all the applications that are in the task's Windows image) plus Adobe Reader and WinZip.

4.2.2 Install logic

There are other ways to design the application install logic for *Windows Clone Install* tasks. For example, we might have encapsulated into a batch file many or all of the statements that we inserted into the command list.

We could have mirrored the logic used in the *Windows Native Install* task. For example, we could have done the following:

1. Use a file similar to COPYAPP.BAT (see section 3.1.4.8 above) immediately following the !custing\custing.bat command to download the image files.
2. Use a file similar to APPSINST.CMD (see section 3.1.4.6 above) in the same location that we used in section 4.2.1 above to install the applications.
3. Follow that with a file similar to DELAPP.BAT (see section 3.1.4.11 above) to delete the application install directories.

That technique would have made the command list look less cluttered.

However, the advantage to our technique is that the application images are tied to the *Windows Clone Install* task (because we inserted them with the Command List Editor Wizard. RDM will prevent us from accidentally deleting the images while they are used by the task.

5. Notices

This information was developed for products and services offered in the U.S.A.

IBM might not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service might be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right might be used instead. However, it is the user responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM might have patents or pending patent applications covering subject matter described 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:

*IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.*

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement might not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM might make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product, and use of those Web sites is at your own risk.

IBM might use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Some software might differ from its retail version (if available) and might not include all user manuals or all program functionality.

IBM makes no representations or warranties regarding third-party products or services.

5.1 Edition notice

© COPYRIGHT INTERNATIONAL BUSINESS MACHINES CORPORATION 2005. All rights reserved.

Note to U.S. Government Users Restricted Rights — Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

5.2 Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM

IBM (logo)

Asset ID

IntelliStation

LANClient Control Manager

Netfinity

ServeRAID

ThinkPad

Wake on LAN

xSeries

Adaptec is a trademark of Adaptec Inc. in the United States, other countries, or both.

Broadcom is a trademark of Broadcom Corporation in the United States, other countries, or both.

Microsoft, Windows, and Windows NT are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, or service names might be trademarks or service marks of others.

6. Glossary

BAT file. A file that contains a batch program (that is, a set of commands).

bind. Associating one or more systems to a task. This causes all information to be verified (by one of the STC modules) and a resulting job to be scheduled to run.

console, or RDM Console. The group of programs that make up the user interface to RDM. RDM is client/server in nature so that the Console might run on any computer and not necessarily be running on the same computer as the RDM server or other RDM components. The RDM Console is actually an IBM Director Console on which the RDM Console component is installed.

image. An image is the software stored on a deployment server that is downloaded to a system during an operation. Images vary in size and in the type of software they provide to the system. The purpose and content of each image depends on the task to be accomplished, as well as the method used to download the image from the deployment server to the system. A *native* image is built off a product installation CD. A *clone* image is copied from a donor system.

job. An object managed by the scheduler and created by STC. A job is a binding of one task and one or more systems. A job can be scheduled to run once or to recur. Sometimes a job is called by a different name (Scheduled Task, Running Task), to emphasize some aspect of the job.

managed system. The IBM Director term for its system. Mentioned here only for clarity; the term *system* is preferred when referring to an RDM system.

preboot DOS agent. The preboot DOS agent is a DOS operating system with a communications stack that is booted from the network by the bootstrap agent. The preboot DOS agent performs actions on a system as directed by the RDM server.

Preboot Execution Environment (PXE). PXE is an industry standard client/server interface that allows networked computers that are not yet loaded with an operating system to be configured and booted remotely. PXE is based on Dynamic Host Configuration Protocol (DHCP). Using the PXE protocol, clients can request configuration parameter values and startable images from the server.

The PXE process consists of the system initiating the protocol by broadcasting a DHCPREQUEST containing an extension that identifies the request as coming from a client that uses PXE. The server sends the client a list of boot servers that contain the operating systems available. The client then selects and discovers a boot server and receives the name of the executable file on the chosen boot server. The client downloads the file using Trivial File Transfer Protocol (TFTP) and executes it, which loads the operating system.

Redundant Array of Independent Disks (RAID). RAID is way of storing the same data in different places (thus, redundantly) on multiple hard disks. By placing data on multiple disks, I/O operations can overlap in a balanced way, improving performance. Multiple disks increase the mean time between failure (MTBF) and storing data redundantly increases fault-tolerance.

system. An individual, target system being deployed or managed by RDM. In IBM Director terminology, an RDM system is always a platform managed object. These can represent any of the supported-by-RDM systems. They cannot represent an IBM Director object that RDM does not process, such as a chassis or an SNMP object.

system environment. This is the RDM term for a preboot operating system, one that contains a communications stack and is booted from the network by the bootstrap loader program.

task. An already defined and configured unit of work that is available to be applied to a system or a group (of systems). You create a task by clicking on the applicable task template from the RDM main

console. RDM is installed with predefined tasks, such as data disposal and scan.

task template. A prototype of a specific kind of RDM task. This is a term used to describe the different kinds of tasks shown on the task pane in the main window of the RDM console. Each task template has its own characteristics and attributes. RDM comes with a set of task templates.

Wake on LAN. Technology developed by IBM that allows LAN administrators to remotely power up systems. The following components are essential for the Wake on LAN setup:

- Wake on LAN-enabled network interface card (NIC).
- Power supply that is Wake on LAN-enabled.
- Cable which connects NIC and power supply.
- Software that can send a magic packet to the system.

If the system has the first three of the previous components, the system is called a Wake on LAN-enabled system. Even though a system might be powered off, the NIC keeps receiving power from the system power supply to keep it alive. A network administrator sends a magic packet to the system through some software, for example, RDM or Netfinity IBM Director. The NIC on the system detects the magic packet and sends a signal to the power supply to turn it on. This process is also called *waking up the system*. Using RDM, this process can be scheduled for individual systems. The Wake on LAN feature and RDM together make it very easy for you to deploy software on individual systems on a scheduled basis.