



# **Using RDM to Deploy Applications and Windows**

**A White Paper**

**May 5, 2006**

**Notes:**

Visit [www.ibm.com/pc/safecomputing](http://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 57.

© Copyright International Business Machines Corporation 2005, 2006. 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

<b>1. PREFACE</b>	<b>5</b>
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
<b>2. OVERVIEW</b>	<b>7</b>
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
<b>3. WINDOWS NATIVE INSTALL</b>	<b>10</b>
3.1 INTERNAL TASK LOGIC	10
3.1.1 <i>Overview of task logic</i>	10
3.1.2 <i>Overview of application-install logic</i>	10
3.1.3 <i>Task folder</i>	11
3.1.4 <i>Explore the task logic</i>	12
3.2 APPLICATION IMAGE EXAMPLES	23
3.2.1 <i>Standard applications</i>	23
3.2.2 <i>Irregular application</i>	26
3.2.3 <i>MSI application</i>	29
3.2.4 <i>Collection of applications</i>	31
3.2.5 <i>IBM Director Agent</i>	34
3.3 INSTALLING APPLICATIONS	38
3.3.1 <i>Using RDM's built-in application-install capability</i>	38
3.3.2 <i>Customizing RDM's built-in application-install capability</i>	39
3.3.3 <i>Using RDM's command list</i>	42
3.3.4 <i>Using CMDLINES.TXT</i>	44
3.3.5 <i>Integrating updates or hotfixes into your operating-system image</i>	44
<b>4. WINDOWS CLONE INSTALL</b>	<b>46</b>
4.1 INTERNAL TASK LOGIC	46
4.1.1 <i>Find the task folder</i>	46
4.1.2 <i>Explore the task logic</i>	46
4.2 INSTALLING APPLICATIONS	51
4.2.1 <i>Procedure</i>	52
4.2.2 <i>Install logic</i>	55
<b>5. NOTICES</b>	<b>57</b>
5.1 EDITION NOTICE	57
5.2 TRADEMARKS	57
<b>6. GLOSSARY</b>	<b>59</b>



# 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) 4.30, 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 4.30 User's Reference* – The main reference manual for RDM

- *Remote Deployment Manager 4.30 Installation and Configuration Guide* – Describes the complete installation process of RDM
- *Remote Deployment Manager 4.30 Compatibility 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
- Broadcom network teaming configuration
- RSA-II firmware update

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 each application (described in section 3.2 below).
3. Create a *Windows Native Install* task (including its corresponding operating-system image) that uses the application images (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 Windows and the applications 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 Windows and 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 each application (described in section 3.2 on page 23).
4. Add logic to the *Windows Clone Install* task's command list to install the application images (described in section 4 on page 46).
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 Windows and the applications installed correctly.

Reasons for installing some applications with their unattended install procedure (after downloading the clone image) include:

- The application is difficult to clone, because of system-unique configuration requirements. IBM Director Agent is an example.



- The application is impossible to clone, because it updates firmware. RSA-II firmware update is an example.

## 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.

**Note:** The task logic in RDM 4.30 is significantly different than it was in RDM 4.20.

#### 3.1.1 Overview of task logic

In this section, we describe the *Windows Native Install* task logic for 32-bit Windows 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. The command list runs DISKCFG.BAT, which does the remaining disk partitioning and formatting.
7. The command list installs a Windows service pack and any applications that it contains.
8. The command list runs POSTINST.BAT, which installs service-processor device drivers.
9. The system powers off.

#### 3.1.2 Overview of application-install logic

In this section, we summarize how RDM 4.30 (or later) installs applications as part of a Windows Native Install task. This process is substantially different from earlier RDM releases, and it is straightforward to understand. The steps are the following, which occur immediately after the Windows-install part of the task is complete:

1. STARTUP.BAT runs (under Windows) from the startup folder.
2. RDMAGENT.BAT, which is run by STARTUP.BAT under Windows, is a loop that runs statements from the command list.
3. The command list contains 4 statements, like the following, for each application:

```
;Installing application Adobe Acrobat Reader 7.0.0
!MTFTP get %%SERVER_IP%% image\053917266703.bat c:\app.bat
```

```
!c:\app.bat
!del c:\app.bat
```

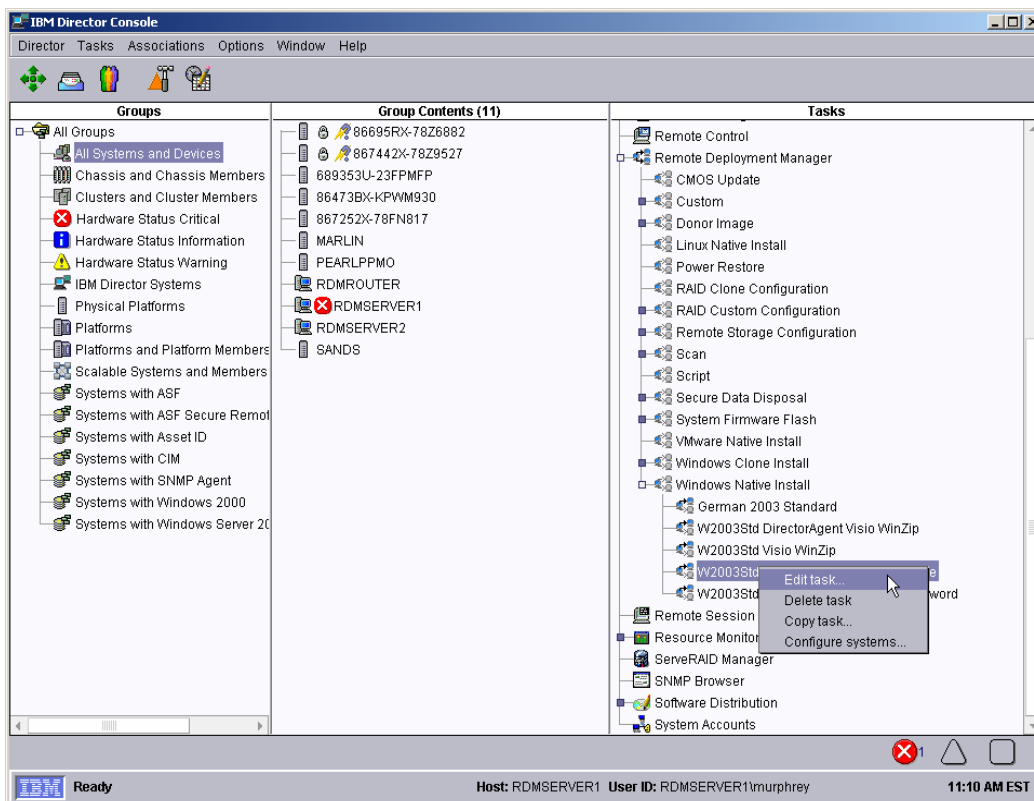
4. The first statement, which starts with a semicolon (;), is a comment.
5. MTFTP.EXE, which is run by RDMAGENT.BAT under Windows, downloads the application-image's batch file (naming it APP.BAT).
6. APP.BAT, which is run by RDMAGENT.BAT under Windows, installs the application using the following logic:
  - a. Deletes an old C:\APP directory and an old APP.ZIP file, if either exists.
  - b. MTFTP.EXE, which is run by APP.BAT under Windows, downloads the application-image's ZIP file (naming it APP.ZIP).
  - c. UNZIP.EXE, which is run by APP.BAT under Windows, downloads the application-image's batch file (naming it APP.BAT).
  - d. Deletes APP.ZIP (which is no longer needed).
  - e. RDMRUNAPP.BAT, which is run by APP.BAT under Windows, installs the application, using the application-image's command and parameters.
7. The fourth statement, which is run by RDMAGENT.BAT under Windows, deletes APP.BAT (which is no longer needed).

All of the application's install logic is encapsulated in the application-image's batch file.

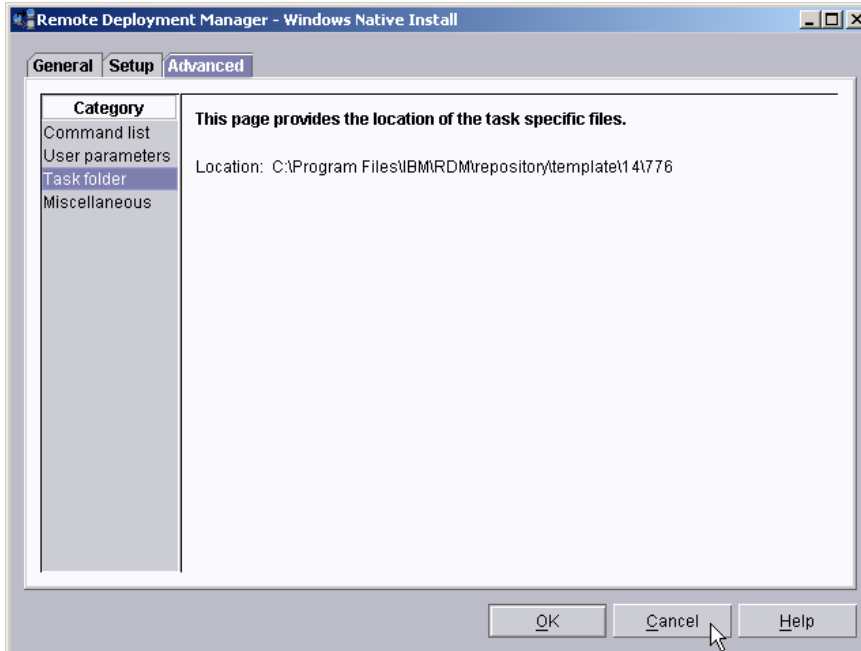
### 3.1.3 Task folder

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.



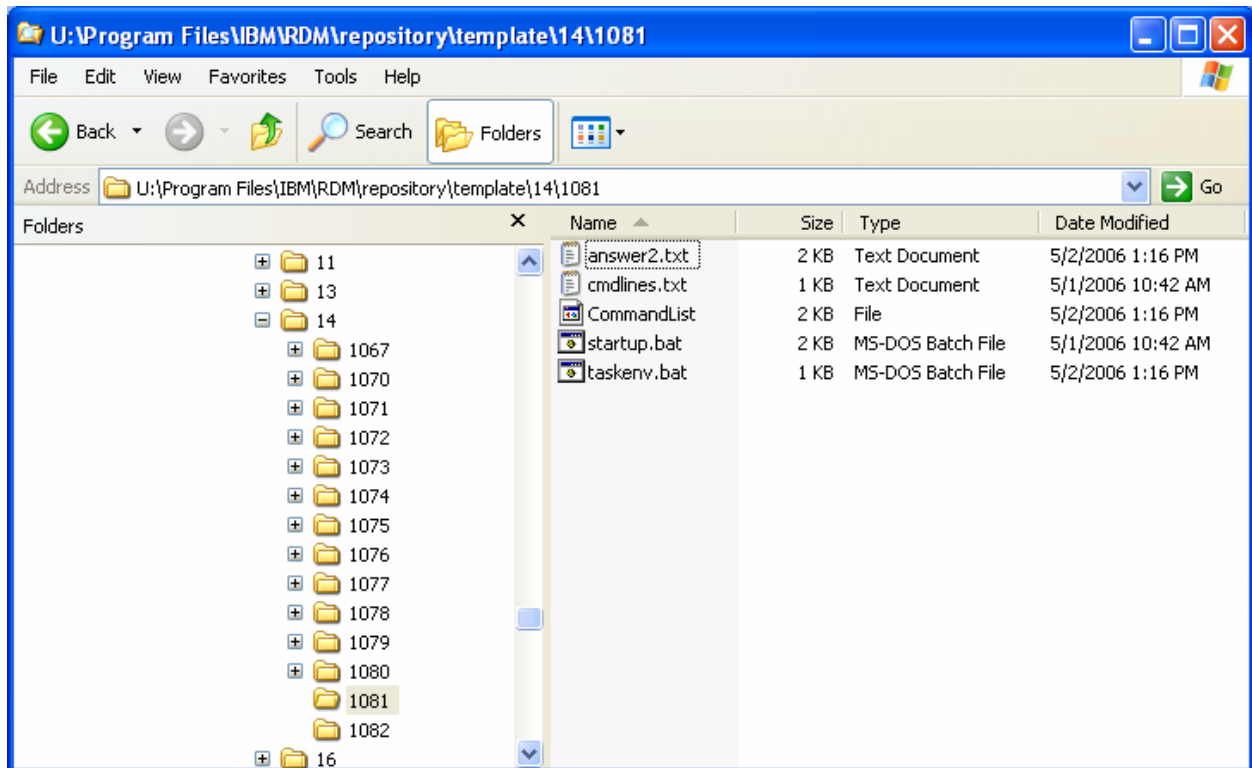
2. Select the *Advanced* page, and then select the *Task folder* category.



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.

**Note:** Some task folders may contain several files with “bak” or “obsolete” in their names. This will happen for tasks that were created in RDM 4.20 and later modified during the upgrade to RDM 4.30. Those files are saved for reference, in case you had customized the task and now need to incorporate your customization in the upgraded task.

### 3.1.4.1 CommandList

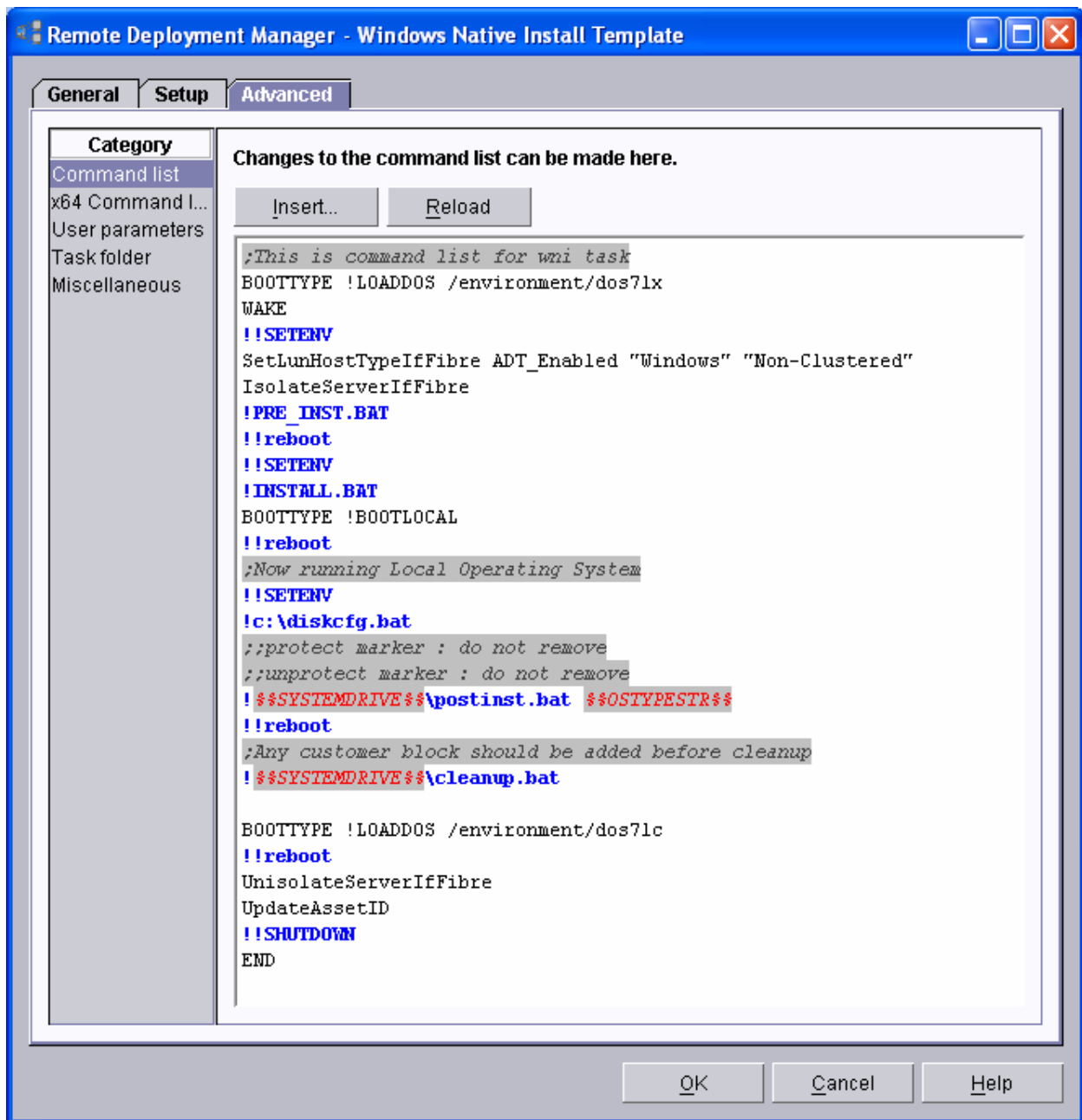
This file contains the overall task logic. It is the key file for any RDM task. RDM runs the commands in top-to-bottom order, and then (typically) powers off the system. Some commands spawn quite a bit of task logic that is encapsulated in batch files. So you need to understand that logic, in addition 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 3.1.3 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. **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).
3. **!!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.
4. **SetLunHostTypeIfFibre ADT\_Enabled "Windows" "Non-Clustered"** – 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 set the host type of the FASTT fibre boot drive to Windows Non-Clustered with Automatic Data Transfer (ADT) enabled.

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. **!PRE\_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 2 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. **!INSTALL.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 2 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 the running of the RDAGENT.EXE program in a loop (via the RDAGENT.BAT file).

This is where the command list initiates all of the application installs that are part of the task. This also allows the user to add statements to the command list (after the !!reboot statement) for customization.

12. **!!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.

13. **DISKCFG.BAT** – This creates and formats all partitions other than the boot (i.e., C:) partition.

The drive letters of these partitions depend on the target system's hardware configuration. For example, if the system has one hard drive and one CD drive, a task that creates 2 partitions will result in the following drive-letter assignments:

- C: = the boot partition (on the hard drive)
- D: = the CD drive

- E: = the second partition (on the hard drive)

If you prefer a different drive-letter scheme (e.g., D: as the second partition and E: as the CD drive), you need to customize the task to change this. An easy way to do this is to create an RDM application image that changes the drive-letter assignments, and to install it as the first application. Then the new drive letters will be in force when the task installs the remaining RDM application images.

14. **;;protect marker : do not remove** – This comment is a place holder. The statements that do the application installs (none is present in the picture above) will immediately follow this statement.

When you create an RDM *Windows Native Install* task, RDM automatically adds those statements here.

When you later edit the task, if you make any changes in the *Image* category of the *Setup* page, RDM replaces all statements between the protect/unprotect place holders (based on the current state of that page). If you then select the *OK* button, RDM saves those changes. (If you select the *Cancel* button, RDM does not change the task in any way.)

For example, if one of your applications requires an immediate reboot after installation, you should add the following statements after the lines that install that application:

```
!!REBOOT
!!SETENV
```

**Important:** If you later make changes in the *Image* category of the *Setup* page, RDM's rebuilding of this section of the command list will remove those statements. If this happens, you will have to put them back.

15. **;;unprotect marker : do not remove** – This comment is a place holder. It marks the end of the application-install area of the command list.
16. **POSTINST.BAT** – This installs service-processor drivers, including Remote Supervisor Adapter, Remote Supervisor Adapter II, and Automatic Server Restart drivers.
17. **!!reboot** – This reboots the system. Because it is a warm reboot, it will be the same kind of boot that it did in step 11 above (i.e., a Windows 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 Windows). STARTUP.BAT then initiates the running of the RDAGENT.EXE program in a loop (via the RDAGENT.BAT file).
- The reason for this reboot is because it is required after the device-driver install done by POSTINST.BAT. You may also add statements to the command list (after the !!reboot statement) for customization.
18. **CLEANUP.BAT** – This removes all of the RDM-related files that the task has used and that are no longer needed.
19. **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.
20. **!!reboot** – This reboots the system. Because it is a warm reboot, it will be the same kind of boot that it did in step 2 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.
21. **UnisolateServerIfFibre** – 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 restore multiple paths between the fibre HBA on the target and the FASTT storage controller.



22. **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.
23. **!!SHUTDOWN** – This powers off the system.
24. **END** – This tells the RDM server that the task is complete.

### 3.1.4.2 PRE\_INST.BAT

This is the first of several batch files run from the command list. Much of the encapsulated task logic is contained in the batch files. The PRE\_INST.BAT file is part of the DOS71X system environment. It is in RDM's localenv\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

if "%OSTYPESTR%"=="win2003serverx64" goto CHECKHW
goto NOTCHCK

:CHECKHW
SET STATUS="CHECKING FOR SUPPORTED HARDWARE"
RDAGENT /L "Checking for system manufacturer."
hwdetect.exe /s
if errorlevel 1 goto HwFAIL

:NOTCHCK
fdisk32 /status > fdiskout.txt

REM GENERATE RDMFDISK.BAT, RDMFORMAT.BAT, FORMATX.BAT AND SETNTFS.BAT
PREPDSKS.EXE
if errorlevel 1 goto FAIL

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

:HwFAIL
@ECHO Failed to %STATUS%
SET RDRASLEVEL=1
SET RDSTATUS="RDINST000E Non-IBM system is detected. This task does not support non-IBM system."
goto END

:END
```

PRE\_INST.BAT creates and formats the boot partition. It does this in the following steps:

1. It downloads and runs the TASKENV.BAT file, which sets some environment variables.
2. If the task deploys 64-bit Windows, it runs HWDetect.EXE to confirm that the target system is an IBM system.

**Important:** The version of 64-bit WinPE that is included in RDM is licensed only for IBM systems. To deploy non-IBM systems, you must obtain WinPE from Microsoft and install the appropriate device drivers.

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

In order to view the generated-at-run-time 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

INSTALL.BAT contains the high-level encapsulated task logic by which RDM installs Windows. The file is part of the DOS71X system environment. It is in RDM's local\env\o\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 for the larger files) and unzips device driver repositories and various utilities and RDM batch files. It then deletes the zip files.
6. It downloads the ANSWER2.TXT file, and sets up some other files used for RDM processing.
7. It downloads the wallpaper image (if one exists), and it creates the other files related to wallpaper install.
8. It runs RAIDCFG.EXE to obtain information about the existing RAID configuration.
9. It runs SCAN.EXE to obtain information about the current hardware configuration.
10. It modifies the ANSWER2.TXT file with customized hardware information.
11. It creates the textmode drivers specific to the target system's machine type and to the specific Windows version that the task installs.
12. It creates the CLIENT.INI and CLIENT.BAT files, which are used under Windows to set environment variables and parameter values.

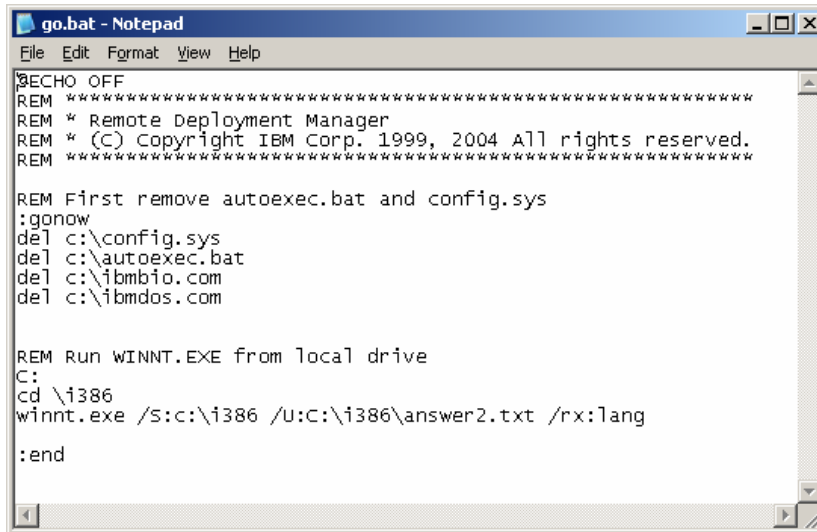
After INSTALL.BAT runs, the command list reboots the target system to its local hard drive. Because of step 3 above, the system boots IBM DOS 7.1, and its AUTOEXEC.BAT file runs GO.BAT, which performs the Windows install using Microsoft's WINNT.EXE program.

### 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.
- RDM will handle the value of AutoLogonCount as part of the procedures described in sections below.
- You can add statements that install other Windows components
- Notice how RDM now handles AdminPassword with a variable.

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

[windowsFirewall]
Profiles=windowsFirewall.TurnoffFirewall

[windowsFirewall.TurnoffFirewall]
Mode=0

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

[GuiUnattended]
OEMSkipRegional=1
TimeZone="%Timezone%"
AdminPassword="%AdminPassword%"
AutoLogon=Yes
AutoLogonCount=100
OEMSkipwelcome=1

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

[Display]
BitsPerPel=16
VRefresh=60
XResolution=800
YResolution=600
```

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.

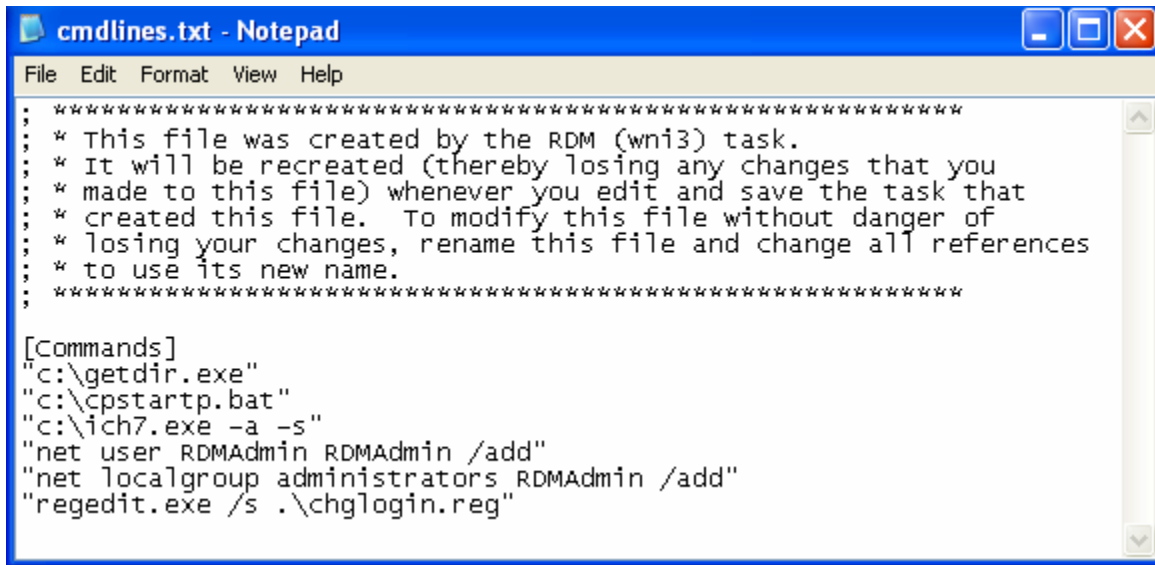
### 3.1.4.6 CMDLINES.TXT

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.

This file creates the RDMAdmin user. RDM uses this local administrator user account to do all of its work under Windows. RDM deletes the RDMAdmin account at the end of the task.



```

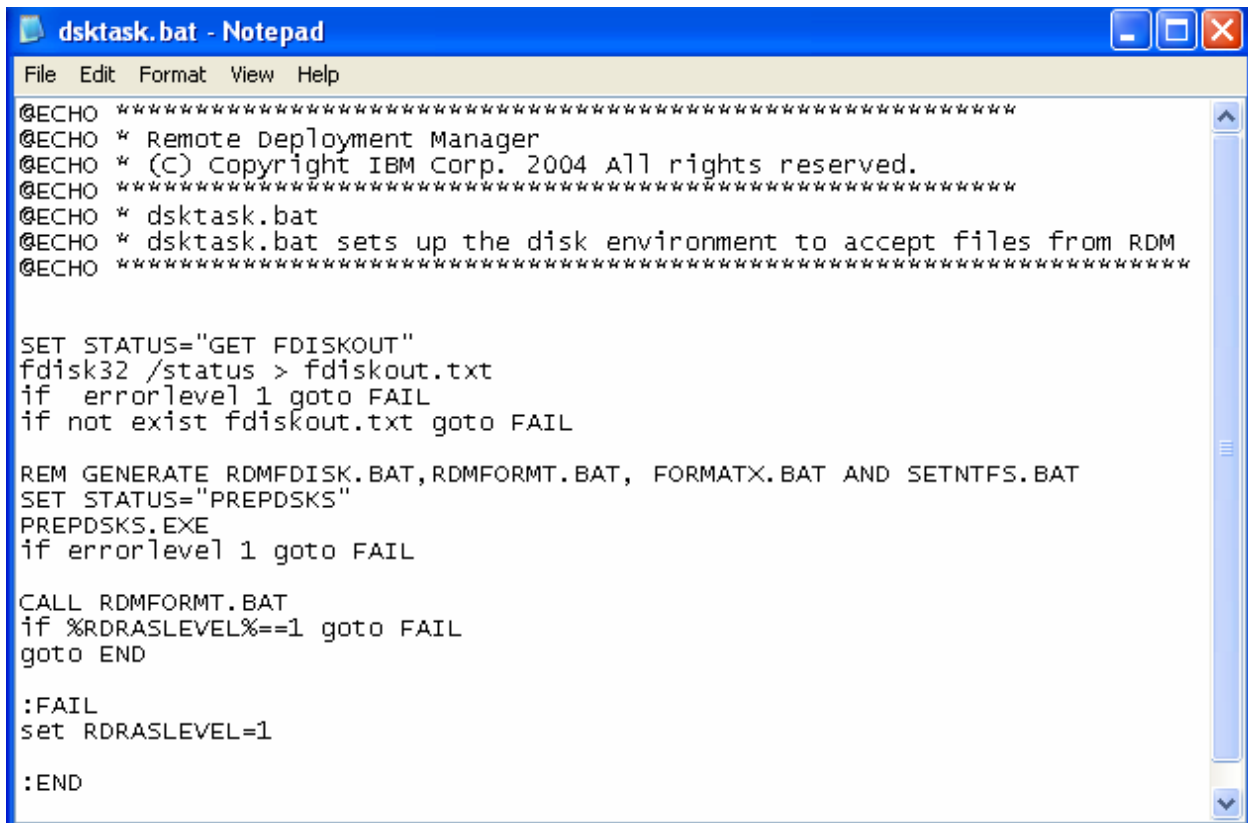
cmdlines.txt - Notepad
File Edit Format View Help
: *****
: * This file was created by the RDM (wni3) task.
: * It will be recreated (thereby losing any changes that you
: * made to this file) whenever you edit and save the task that
: * created this file. To modify this file without danger of
: * losing your changes, rename this file and change all references
: * to use its new name.
: *****

[Commands]
"c:\getdir.exe"
"c:\cpstartp.bat"
"c:\ich7.exe -a -s"
"net user RDMAdmin RDMAdmin /add"
"net localgroup administrators RDMAdmin /add"
"regedit.exe /s .\chglogin.reg"

```

### 3.1.4.7 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, RDMFORMAT.BAT, FORMATX.BAT AND SETNTFS.BAT
SET STATUS="PREPDSKS"
PREPDSKS.EXE
if errorlevel 1 goto FAIL

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

:FAIL
set RDRASLEVEL=1

:END

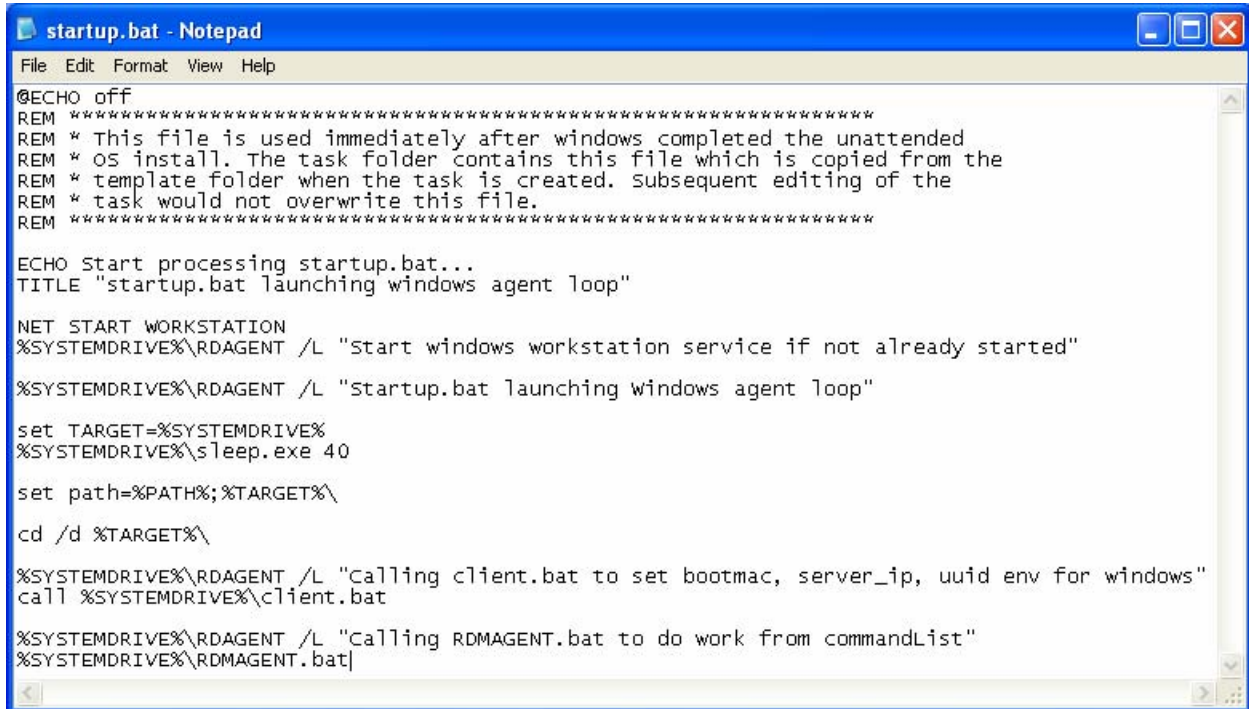
```

### 3.1.4.8 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:

- Issues a command to start the Windows Workstation Service, and then waits 40 seconds. Typically, this service is already started, but these statements ensure that network communication is configured and working before the task continues with the rest of its work.
- Sets up some environment variables needed by RDM (via file CLIENT.BAT, below).
- 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.



```

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...
TITLE "startup.bat launching windows agent loop"

NET START WORKSTATION
%SYSTEMDRIVE%\RDAGENT /L "start windows workstation service if not already started"

%SYSTEMDRIVE%\RDAGENT /L "startup.bat launching windows agent loop"

set TARGET=%SYSTEMDRIVE%
%SYSTEMDRIVE%\sleep.exe 40

set path=%PATH%;%TARGET%\
cd /d %TARGET%\

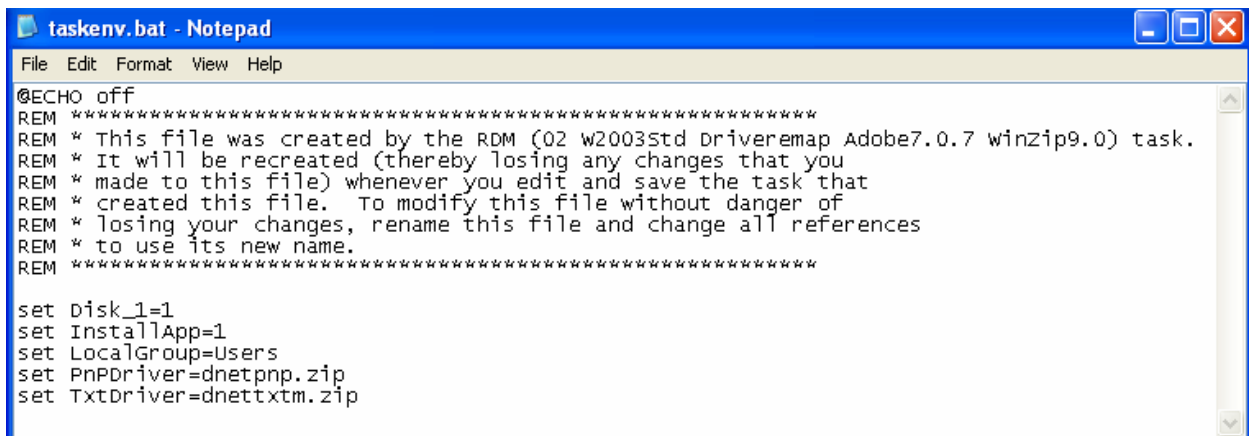
%SYSTEMDRIVE%\RDAGENT /L "calling client.bat to set bootmac, server_ip, uuid env for windows"
call %SYSTEMDRIVE%\client.bat

%SYSTEMDRIVE%\RDAGENT /L "calling RDMAGENT.bat to do work from commandList"
%SYSTEMDRIVE%\RDMAGENT.bat

```

### 3.1.4.9 TASKENV.BAT

This file sets some task-related environment variables. 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 (02 w2003std Driveremap Adobe7.0.7 winZip9.0) 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 Disk_1=1
set InstallApp=1
set LocalGroup=Users
set PnPDriver=dnetpnp.zip
set TxtDriver=dnettxtm.zip

```

### 3.1.4.10 CLIENT.BAT

RDM generates this file. It contains some environment variables that are needed by RDM when running in Windows to ensure that it communicates over the correct network adapter.

## 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 2.

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.

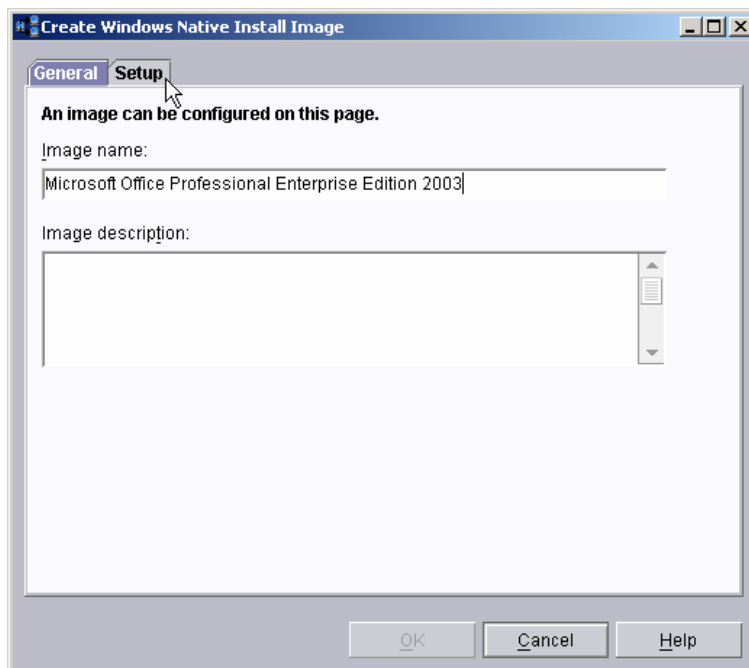
Here is the procedure:

#### 3.2.1.1 Obtain the Microsoft Office install media

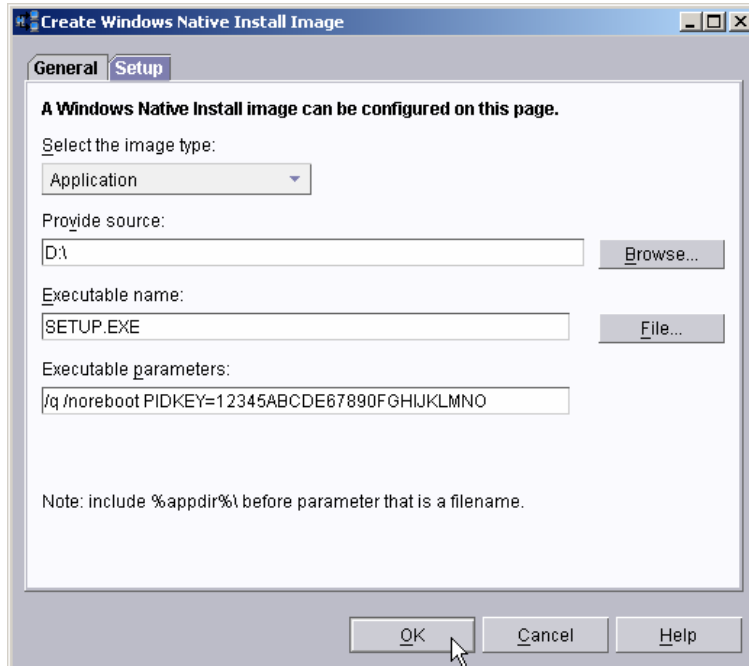
1. We used the Microsoft Office Professional Enterprise Edition 2003 CD from MSDN.

#### 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. 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.



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

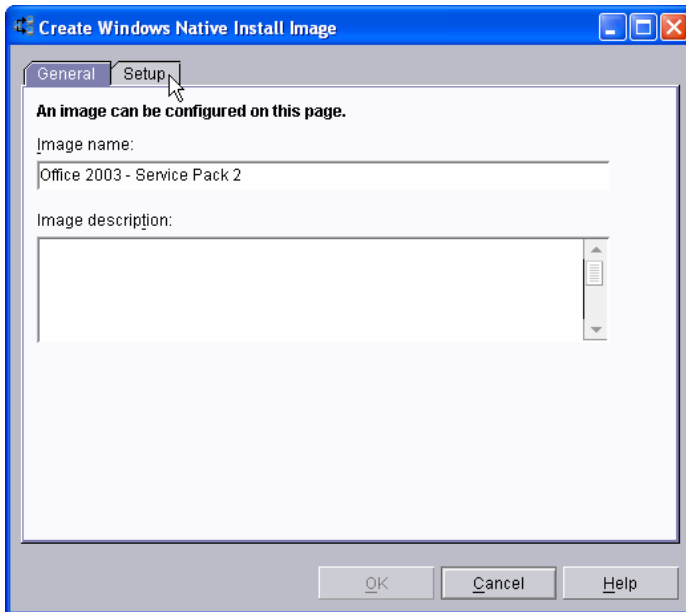
### 3.2.1.3 Obtain the Microsoft Office service-pack install media

10. We used the Office2003SP2-KB887616-FullFile-ENU.exe file from MSDN. We created a directory that contains only this file.

### 3.2.1.4 Create the RDM Windows Native Install application image

11. Open the RDM Image Management window, using the *Tasks* → *Remote Deployment Manager* → *Image Management* → *Create and Modify Images...* menu.
12. 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.





13. 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 2

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:

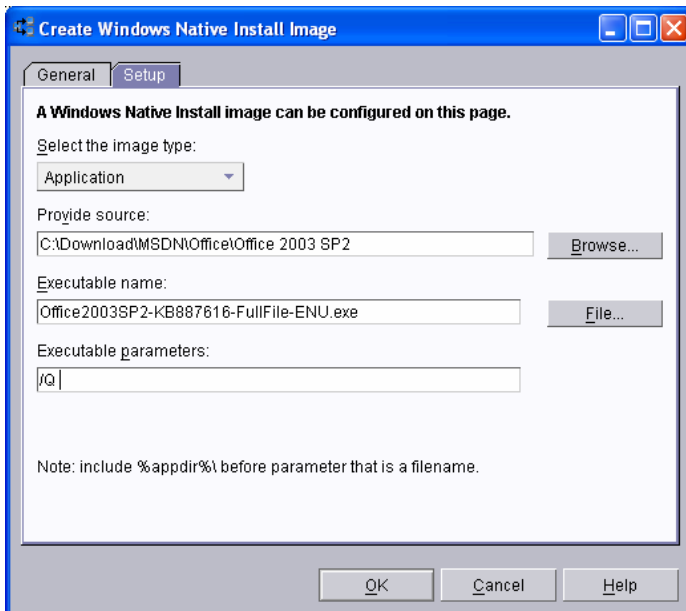
01 Microsoft Office 2003

02 Microsoft Office SP 2

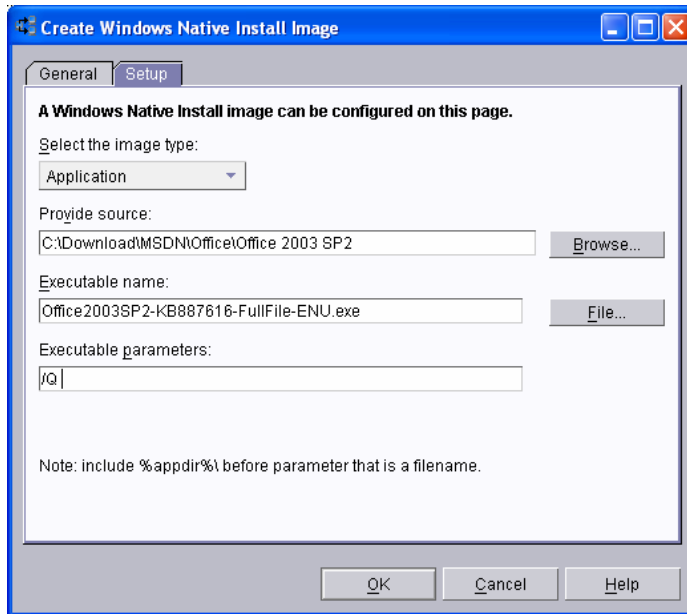
03 WinZip 9.0

04 Adobe Reader 7.0.7

05 IBM Director Agent 5.20.2



14. Then select the *Setup* page.
15. Select *Application* as the image type, from the dropdown menu.
16. Select the *Browse...* button, and navigate to the directory that contains the Office2003SP2-KB887616-FullFile-ENU.exe file.
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

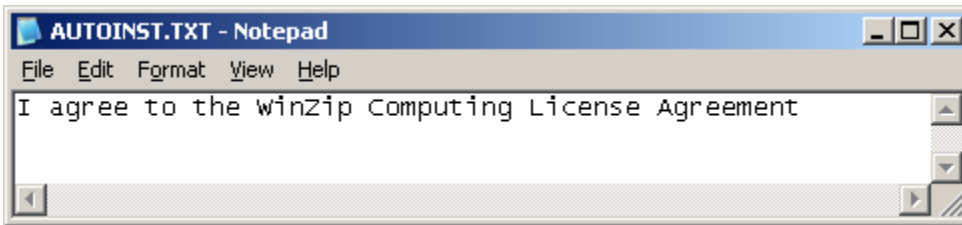
1. Download the install file from the WinZip web site. Our file is `winzip9.0.exe`, whose size is 2,366 KB.

#### 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.

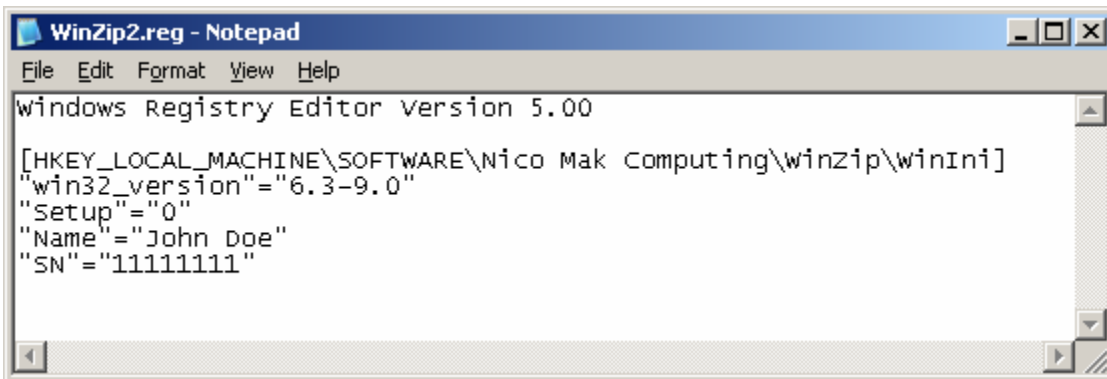
### 3.2.2.3 Customize the install directory

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



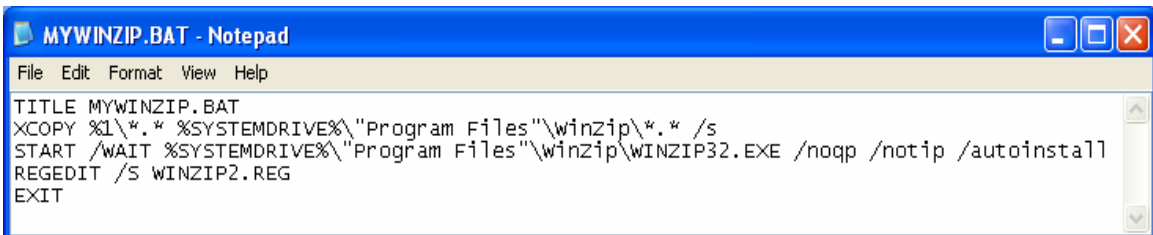
This prevents the license screen from appearing each time you run WinZip.

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 %SYSTEMDRIVE%\APP, and the WINZIP32.EXE file needs to run from its final install directory, %SYSTEMDRIVE%\Program Files\WinZip. So RDM will run a command like

```
MYWINZIP.BAT "%SYSTEMDRIVE%\Program Files\WinZip"
```

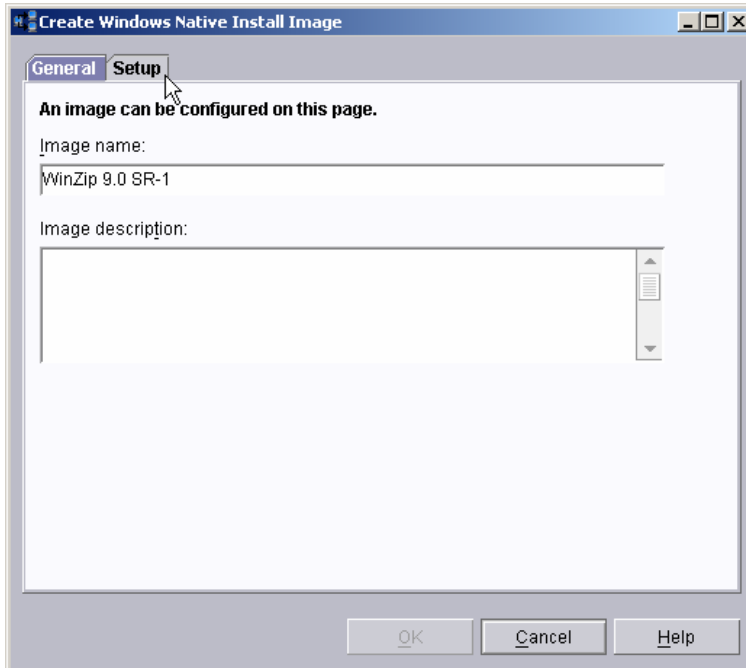
to install WinZip.

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.

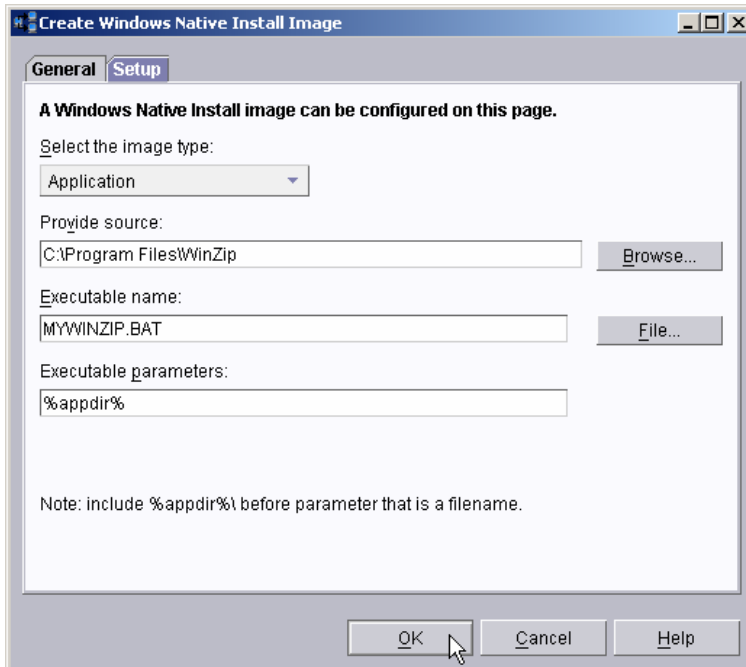
Note also the use of the environment variable %SYSTEMDRIVE%, here. Doing this gives us the flexibility to have the application install not depend on the drive letter of the boot drive. This is useful in a case where you remap the drive letters (e.g., when installing Citrix Metaframe servers) prior to installing the application.

### 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 download the image and install the application.
- A zip file that contains the application install directory.

### 3.2.2.5 RDM install logic

When RDM installs this application, via the image's batch file, the actual install command looks like this:

```
start /WAIT "Installing Winzip 9.0 SR-1" "MYWINZIP.BAT" %SYSTEMDRIVE%\app
```

The variable `%appdir%` that you entered as the executable parameters becomes `%SYSTEMDRIVE%\app` in that command.

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 above), 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 could be to download LCCUSTOM.EXE and to use it to change `%appdir%` to its correct run-time value. An alternate logic could be simply to use `%SYSTEMDRIVE%\app` in your INI file (since in RDM 4.30, all applications install from that same directory).

### 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 (but you could substitute another version, such as Adobe Reader 7.0.7, if you account for the different install file name). 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.

**Note:** If you had earlier installed or extracted any version of Adobe Reader, your directory might be named slightly differently, as in these examples:

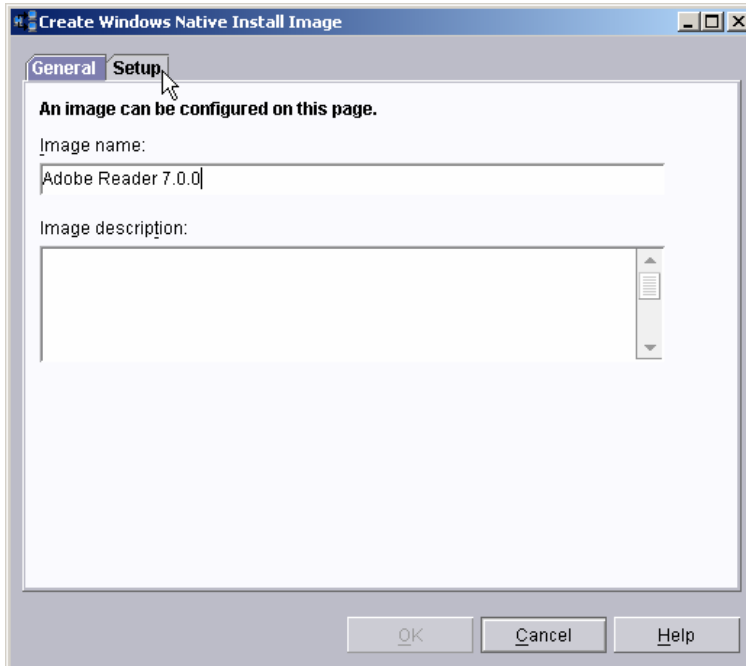
```
C:\Program Files\Adobe\Acrobat 7.0\Setup Files\RdrBig707\ENU
```

```
C:\Program Files\Adobe\Acrobat 7.0\Setup Files\RdrBig707\ENU_
```

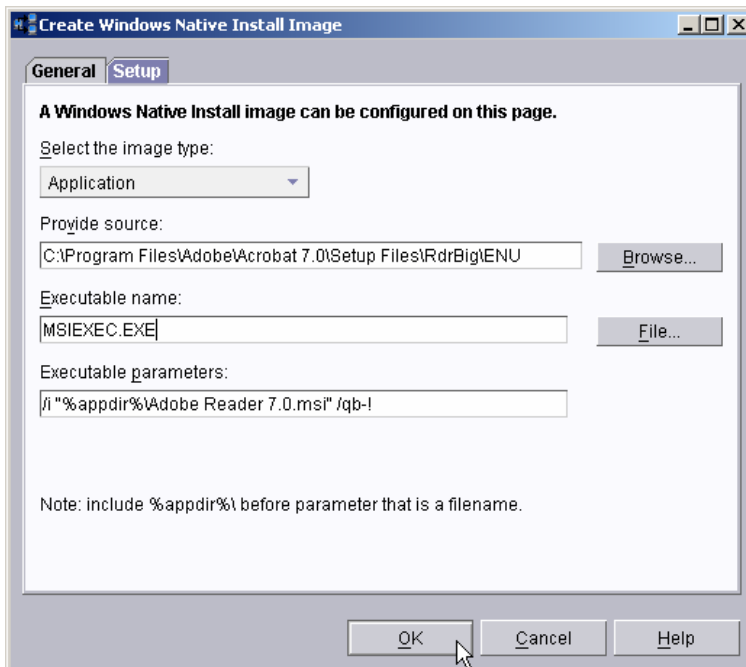
Just pick the appropriate directory.

### 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...* menu.
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).

Note that if you are installing Adobe Reader 7.0.7, the executable parameters would be slightly different:

```
/i "%appdir%\Adobe Reader 7.0.7.msi" /qb-
```

**Important:** Make sure that you use the correct version of the double quote ( " ) character when you enter the executable parameters. Do not use the “double opening quote” or “double closing quote” characters, because these will be misinterpreted by the Windows command processor.

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

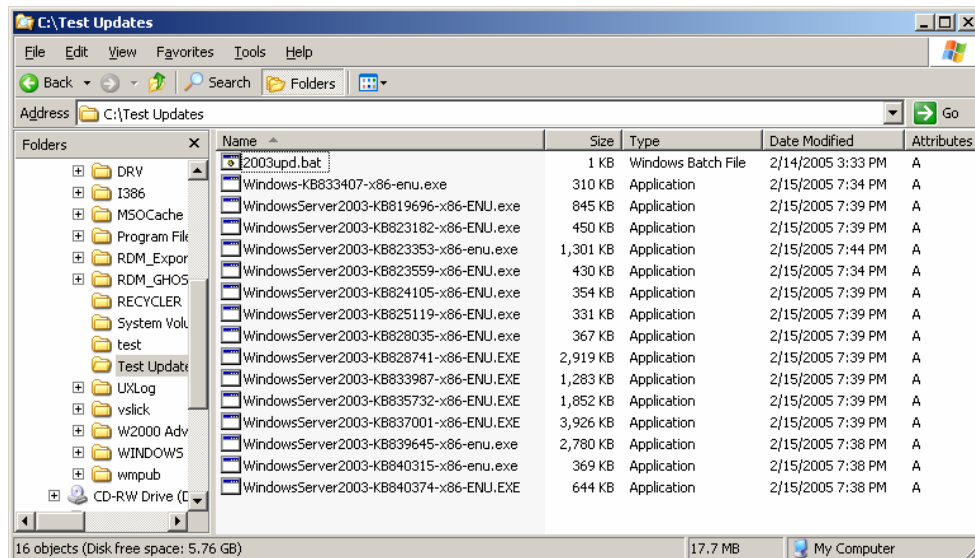
1. Download the updates or hotfixes from the Microsoft web site.

#### 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:

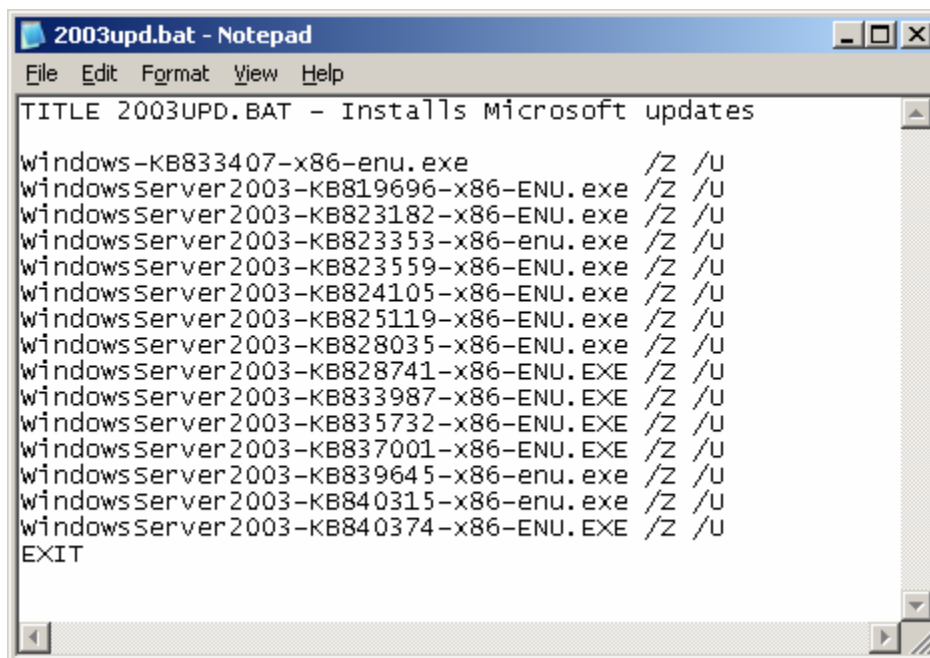
C:\Test Updates

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.



```
2003upd.bat - Notepad
File Edit Format View Help
TITLE 2003UPD.BAT - Installs Microsoft updates

windows-KB833407-x86-enu.exe /Z /U
windowsServer2003-KB819696-x86-ENU.exe /Z /U
windowsServer2003-KB823182-x86-ENU.exe /Z /U
windowsServer2003-KB823353-x86-enu.exe /Z /U
windowsServer2003-KB823559-x86-ENU.exe /Z /U
windowsServer2003-KB824105-x86-ENU.exe /Z /U
windowsServer2003-KB825119-x86-ENU.exe /Z /U
windowsServer2003-KB828035-x86-ENU.exe /Z /U
windowsServer2003-KB828741-x86-ENU.EXE /Z /U
windowsServer2003-KB833987-x86-ENU.EXE /Z /U
windowsServer2003-KB835732-x86-ENU.EXE /Z /U
windowsServer2003-KB837001-x86-ENU.EXE /Z /U
windowsServer2003-KB839645-x86-enu.exe /Z /U
windowsServer2003-KB840315-x86-enu.exe /Z /U
windowsServer2003-KB840374-x86-ENU.EXE /Z /U
EXIT
```

Note that these 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.

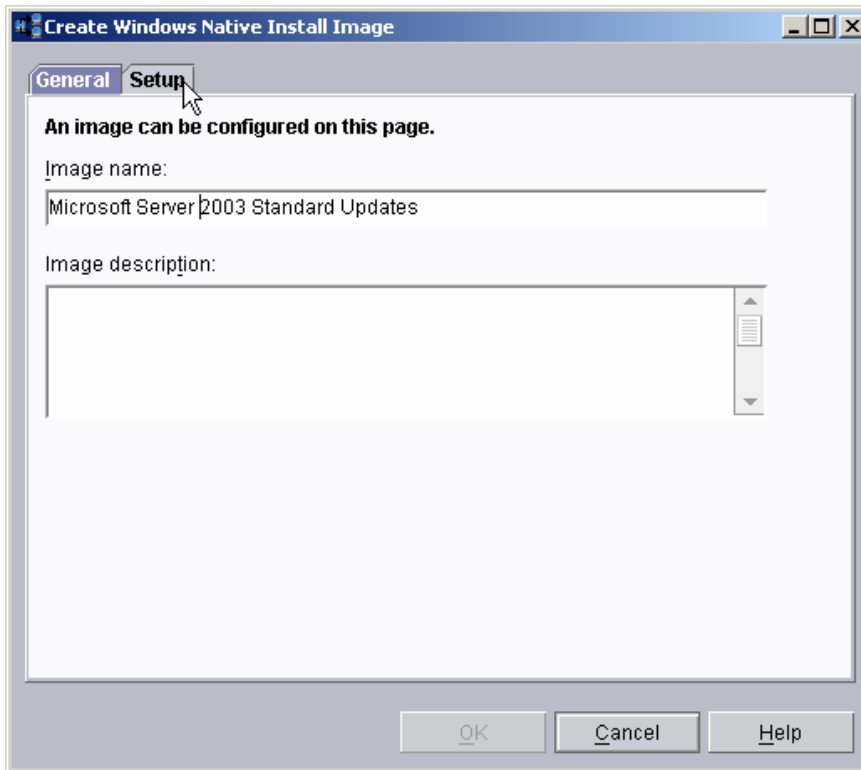
Note the use of the EXIT statement in the batch file. You must do this for any batch file that is used as an application's install program. The TITLE statement is optional, but it is useful when debugging.

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.

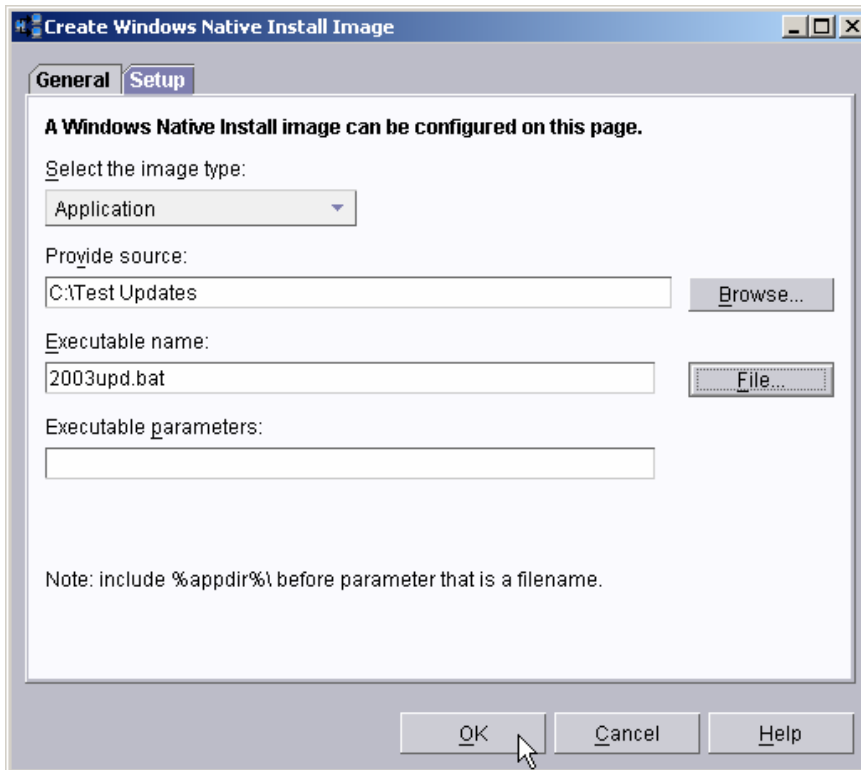
### 3.2.4.4 Create the RDM Windows Native Install application image

5. Open the RDM Image Management window, using the *Tasks*→*Remote Deployment Manager*→*Image Management*→*Create and Modify Images...* menu.
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 5.10, versus the procedure used in versions 4.22 and earlier. We will describe the procedure using version 5.10.2 in this section.

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 an MSI application, instead.)

#### 3.2.5.1 Obtain the install files

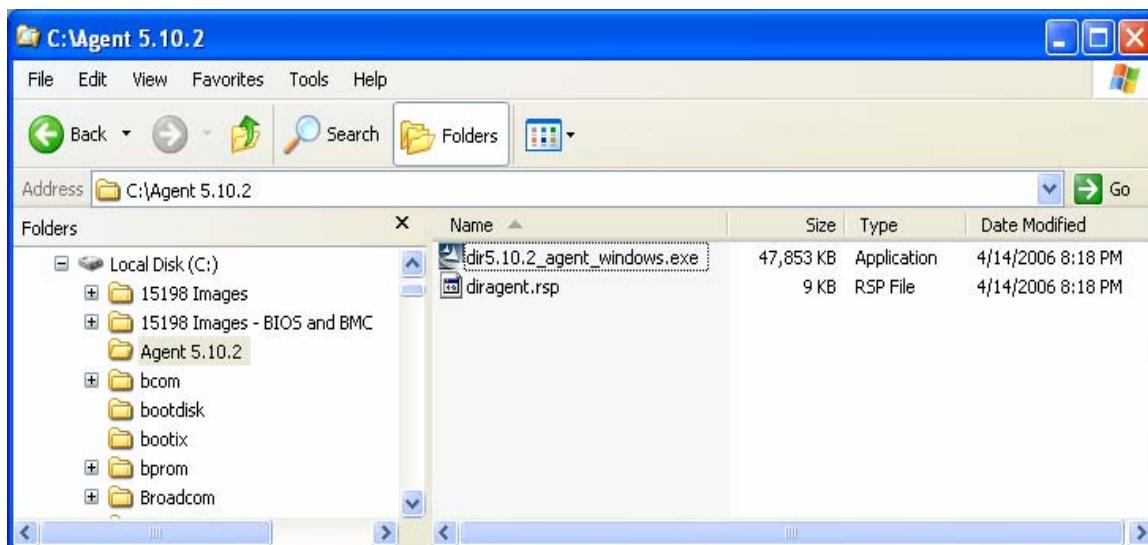
1. You can get the IBM Director Agent install files from an IBM Director CD or from the public IBM web site:
  - a. You can find the IBM Director download matrix on this web site:  
<http://www-307.ibm.com/pc/support/site.wss/document.do?Indocid=SERV-DIRECT>
  - b. You can then navigate to this page  
<http://www-307.ibm.com/pc/support/site.wss/document.do?Indocid=MIGR-63786>  
which contains the e2cd1.iso file, from which you can make a CD. On the CD, these files are in this directory: D:\director\agent\windows\i386\FILES
  - c. Or you can then navigate to this page  
<http://www-307.ibm.com/pc/support/site.wss/document.do?Indocid=MIGR-63780>  
which contains the dir5.20.2\_agent\_windows.zip file.

#### 3.2.5.2 Create an install directory

The reason for this step is so that you can modify the response file before you create the application image. An alternate procedure would be to create the image from the CD, to open the resulting ZIP file, and to modify (and to save) the response file there.

2. On your RDM console system, create a directory for the downloaded executables. We used this directory:  
C:\Agent 5.1
3. Copy the install files to the above directory:
  - From the CD, copy the files from the appropriate directory:  
D:\director\agent\windows\i386\FILES
  - From the web download, unzip the files

The result will be the following:



### 3.2.5.3 Modify the response file

You will have to modify the response file, DIRAGENT.RSP, in order to install IBM Director Agent with RDM. Make whatever changes that your needs dictate. Here is how we modified our response file:

- 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.

- Important:** Make sure that you configure encryption to match the way you installed the IBM Director Server. We chose not to use encryption:

```
EncryptCommunication = N
```

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

```
AddKnownServerAddress=TCPIP::10.2.0.6
```

This line identifies the IBM Director Server, so that systems are automatically discovered by IBM Director.

- We also changed this line, in our version 5.10.2 file:

```
TargetDrive = M
```

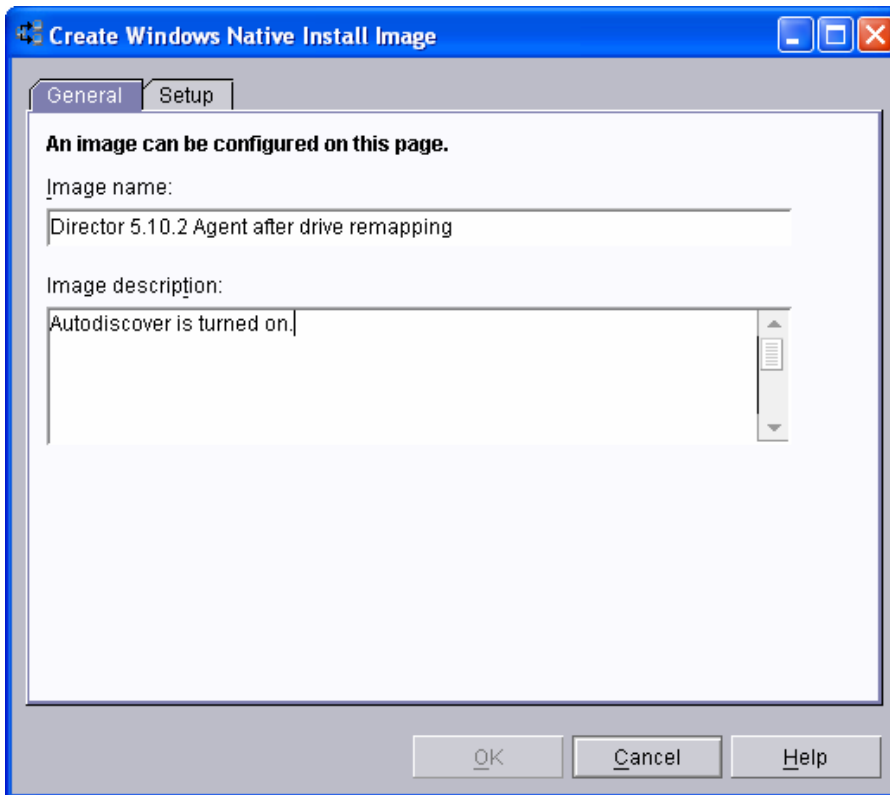
This line changes the drive on which RDM installs Director Agent. You would change it this way if you are building a Citrix Metaframe server with remapped drives.

- We also changed this line, in our version 5.10.2 file:

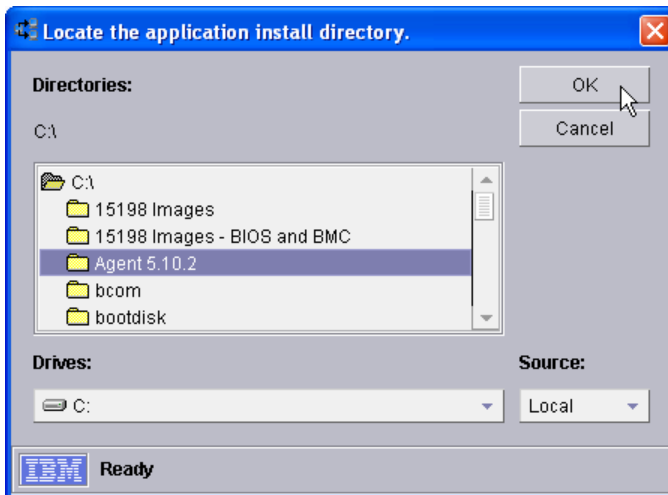
```
WakeOnLan=1
```

### 3.2.5.4 Create the RDM Windows Native Install application image

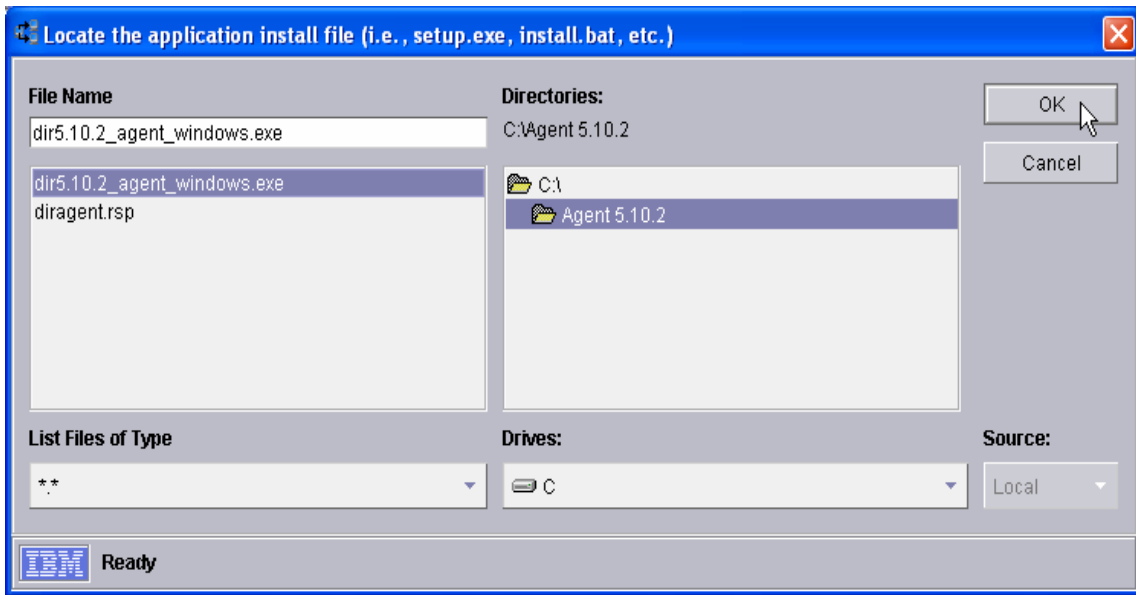
- Open the RDM Image Management window, using the *Tasks*→*Remote Deployment Manager*→*Image Management*→*Create and Modify Images...* menu.
- Select the *Create* button. Then select *Windows Native Install* on the dropdown menu, and select the *OK* button. This displays the *Create Windows Native Install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



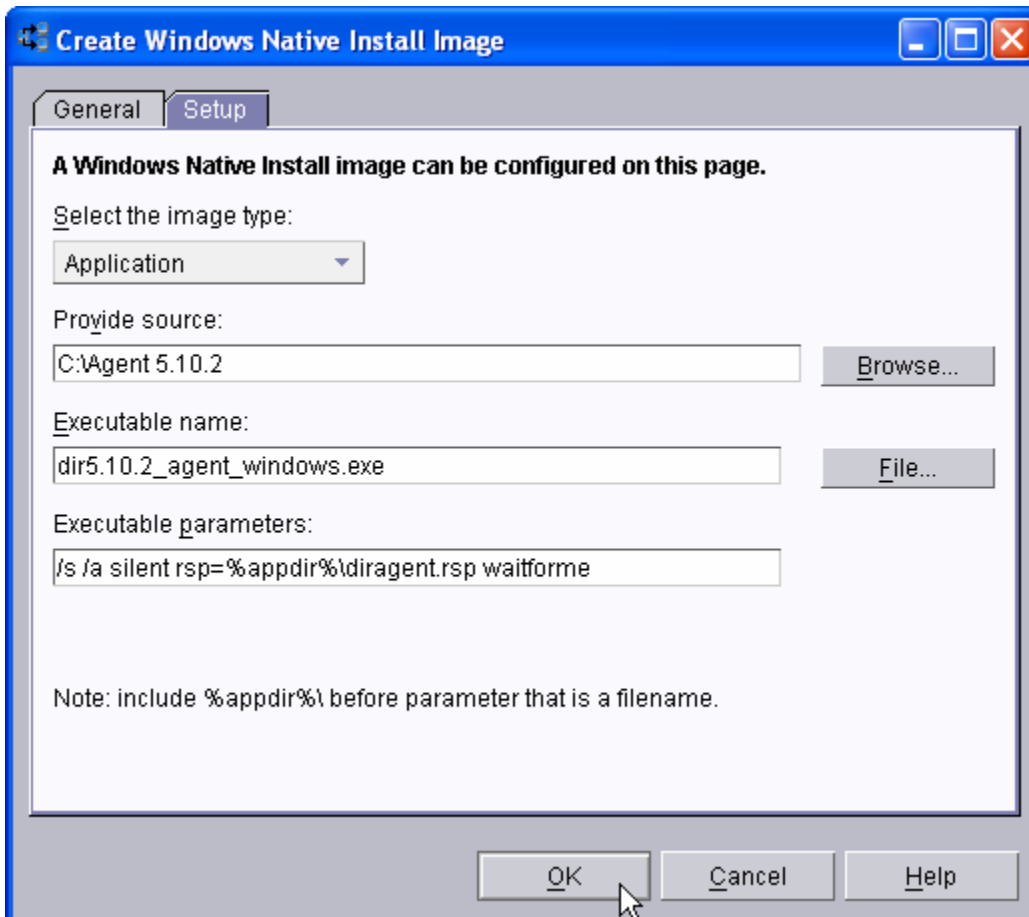
11. Enter an image name and an optional description on the *General* page, and then select the *Setup* page.
12. Select *Application* as the image type, from the dropdown menu.
13. Select the *Browse...* button, navigate to the directory (the one we created in section 3.2.5.2 above), and select the *OK*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



14. Select the *File...* button, and then select the appropriate executable (e.g., DIR5.10.2\_AGENT\_WINDOWS.EXE). Then select the *OK* button.

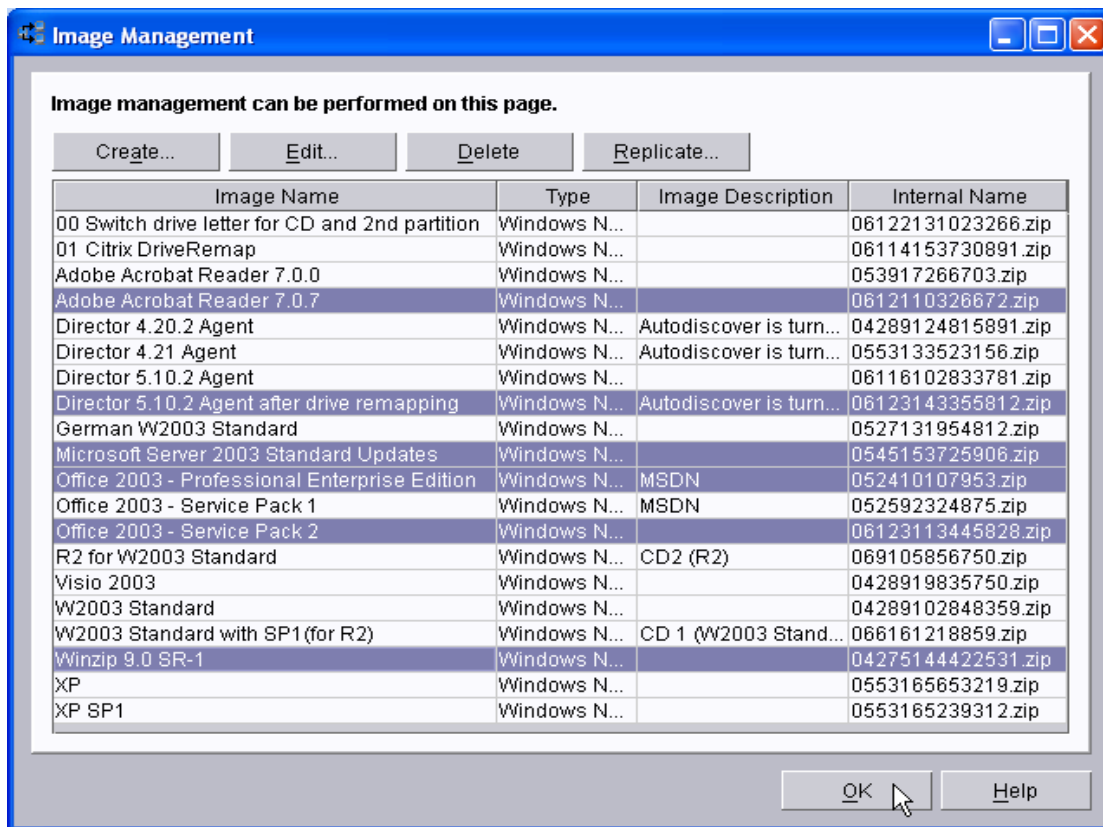


15. Enter the executable parameter as shown below.



16. 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

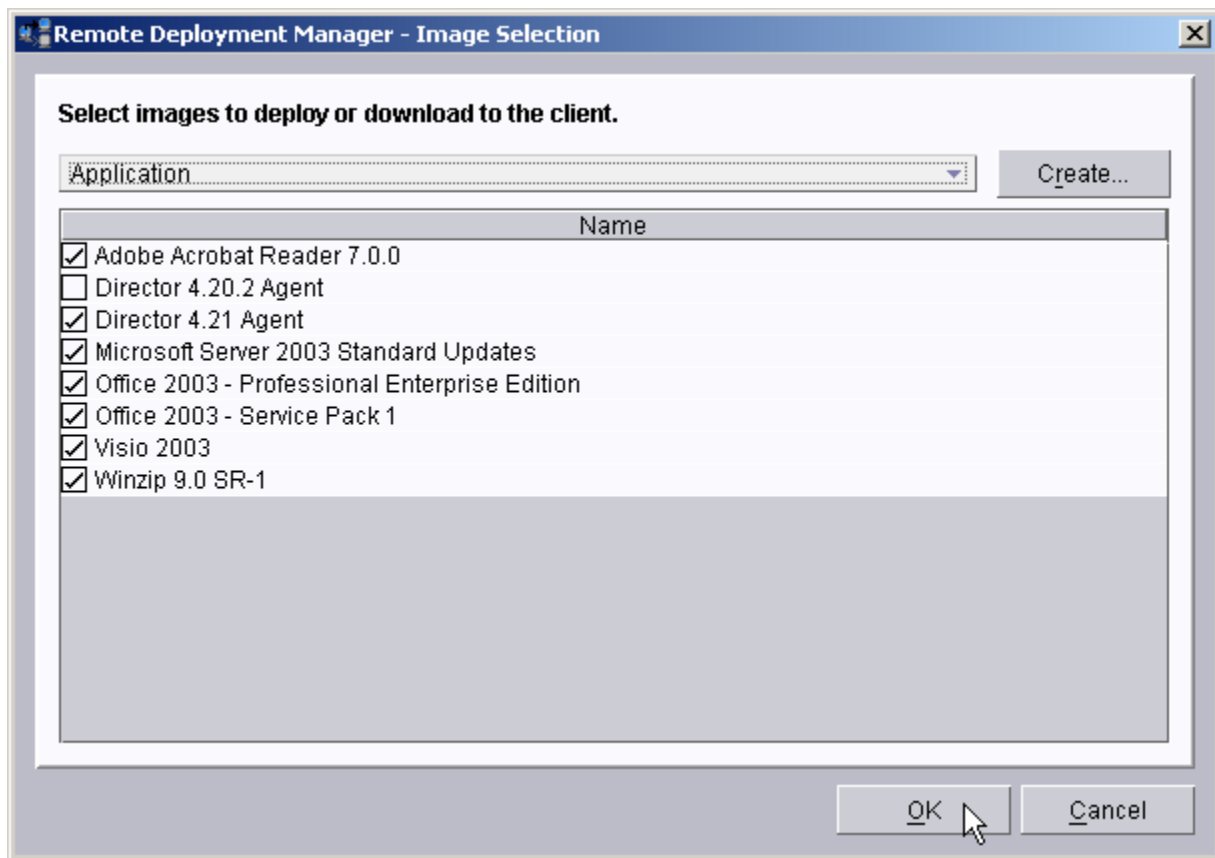
There are several ways to install applications in RDM.

- 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.  
If the application requires a reboot at the end of its installation (e.g., an application that runs the Citrix DRIVEREMAP.EXE), you can easily modify the task's command list to do that reboot. Just add the following commands at the appropriate place:

```
!!REBOOT
!!SETENV
```

The important point is that RDM **must** be in control of the reboots.

- You have to ensure that each application's prerequisites are installed prior to installing the application itself.

This is usually straightforward to do.

- If the prerequisites are Windows components, you can modify the task's ANSWER2.TXT file so that it installs those components.
- If the prerequisites are other applications, you can modify the application names so that the prerequisite applications install first. (RDM installs applications in alphabetical order by their image names.)

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 several applications, and one of them requires that the system reboot before you do anything else with the system. Our example task runs Citrix

DRIVEREMAP.EXE, which is an example of such an application. The basic requirements in this example are:

- Remap the drives immediately after installing the operating system.
- Reboot before installing any other applications.

Other examples (not illustrated here) could be:

- Citrix Metaframe Client requires a reboot before you install its hotfixes.
- Windows hotfixes may require a reboot.

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.

The general procedure (which much easier in RDM 4.30 than it was in prior RDM releases) is the following:

1. Create a *Windows Native Install* task that contains *both* applications.
2. Edit the task, and add the following commands between the commands that install the 2 applications:

```
!!REBOOT
!!SETENV
```

3. Test the task.

We will illustrate the procedure, below, using a task with several applications.

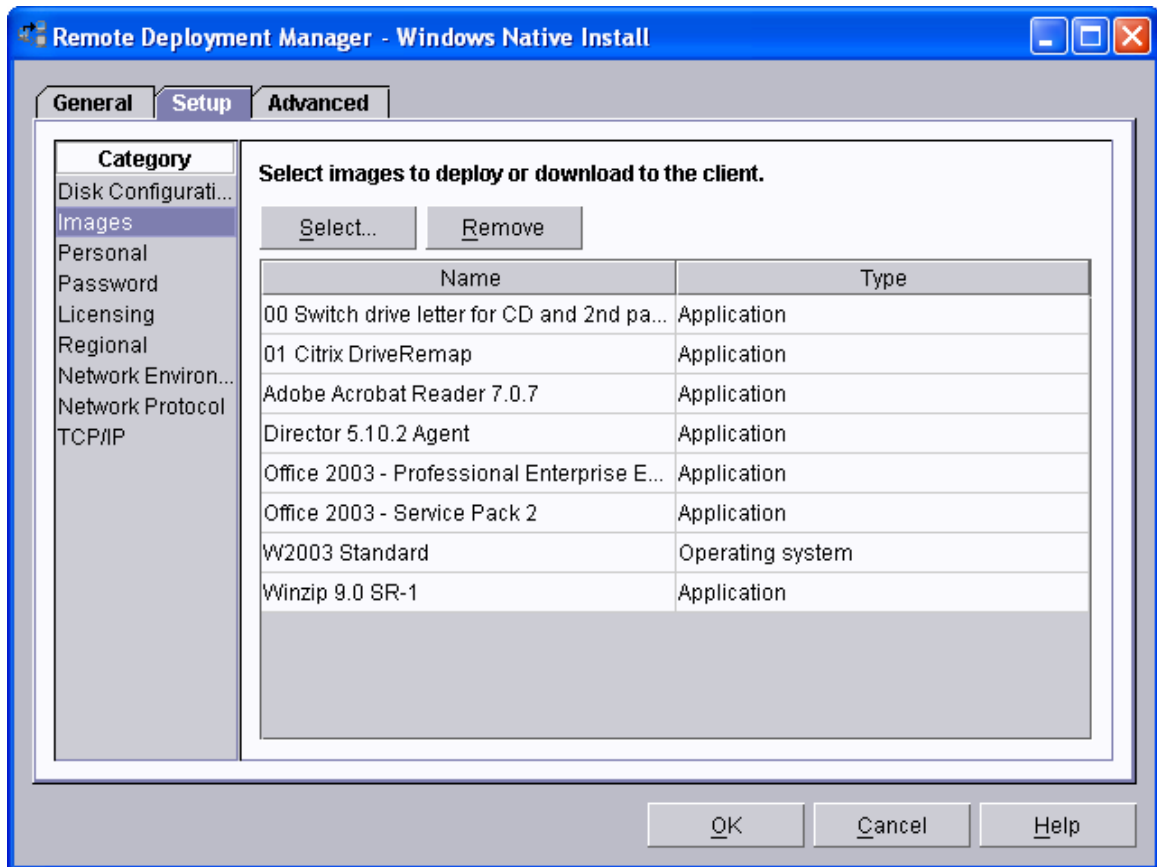
### 3.3.2.1 Create the task

1. Create a *Windows Native Install* task that contains both applications. If you then edit the task, and navigate to the Images category, you will see the images that the task will use.

The picture below illustrates some interesting points about RDM application images. Remember, a *Windows Native Install* task installs applications in alphabetical order, by image name.

- The first two applications must be installed in the proper order. We have used 00 and 01 in their image names to ensure that (alphabetical) order.
- The two Office 2003 applications must be installed in the proper order. We carefully chose their image names to ensure that (alphabetical) order.





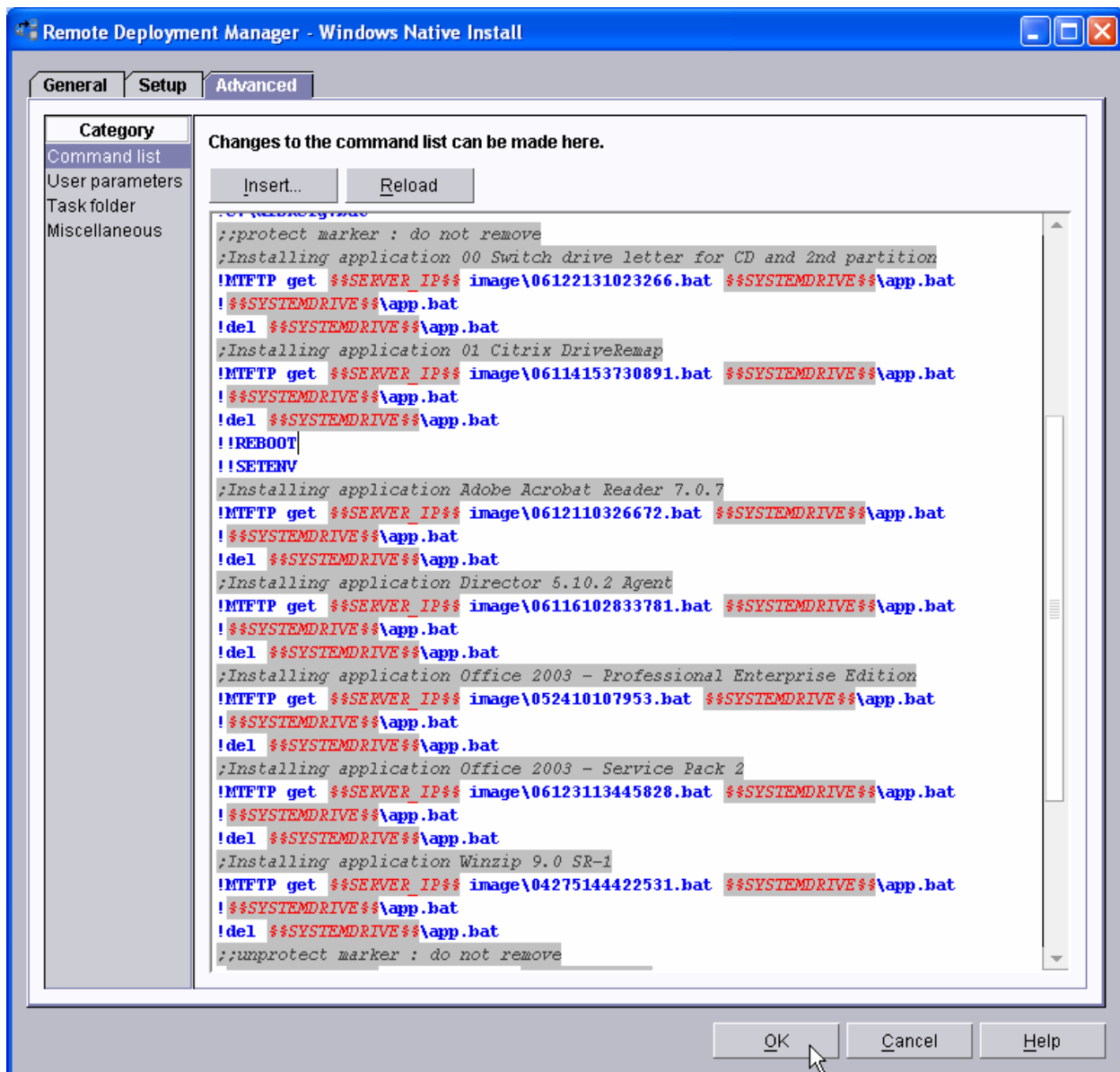
4. Go to the *Advanced* page and edit the command list. Add the following commands between the commands that install the 01 Citrix Drive Remap and the Adobe Acrobat Reader 7.0.7 applications:

```
!!REBOOT
!!SETENV
```

Note the use of variables in the command list.

- The variable names are preceded and followed by `%%`. This is required for most variables that are used in the command list. (For a variable name that is used only on the RDM server, you would precede and follow its name with a single `%`.)
- `%%SERVER_IP%%` is the IP address of the RDM Deployment Server.
- `%%SYSTEMDRIVE%%` is the drive letter plus colon of the Windows boot drive. By using this environment variable, RDM eliminates the need to specify the particular drive letter.

Also note how RDM 4.30 installs applications from the command list.



5. Then select the **OK** button to save the changes.

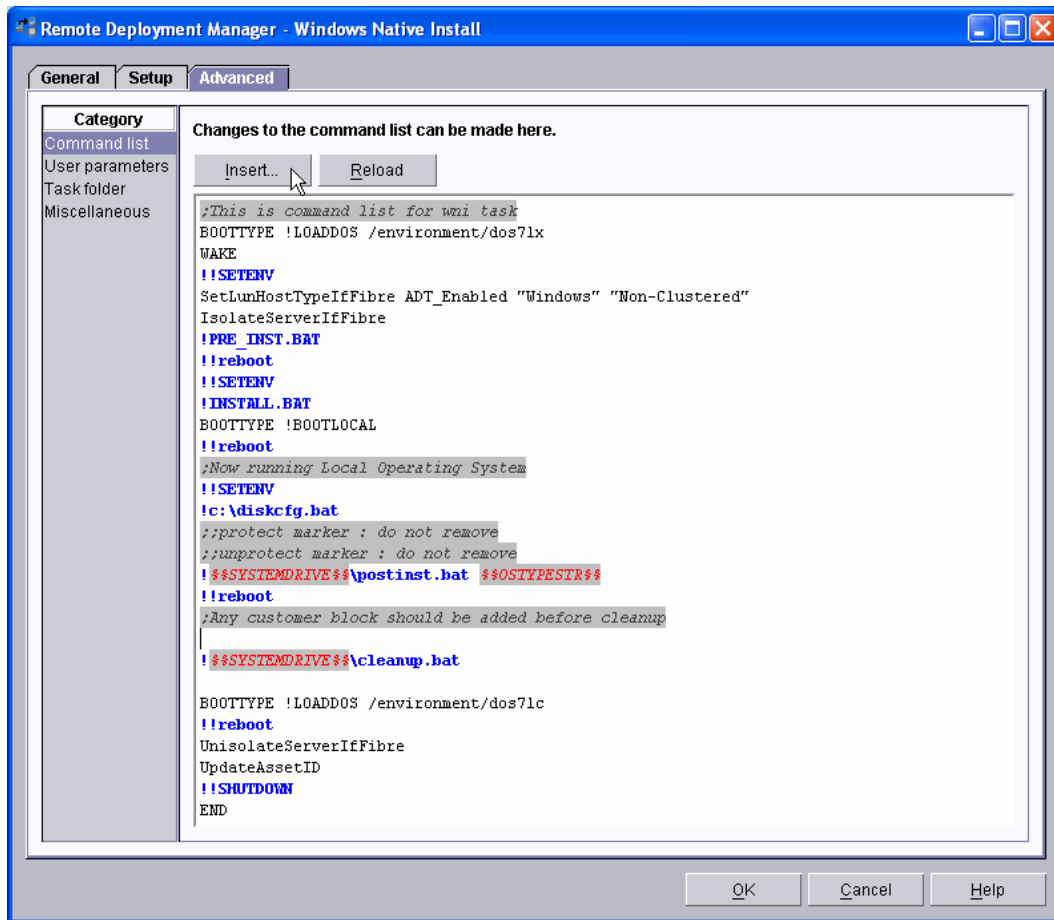
### 3.3.3 Using RDM's command list

You can explicitly add an application install to an RDM *Windows Native Install* CommandList file. Reasons for doing this might be to control when RDM installs the application, or to ensure that RDM will not change that part of the command list when you edit the task.

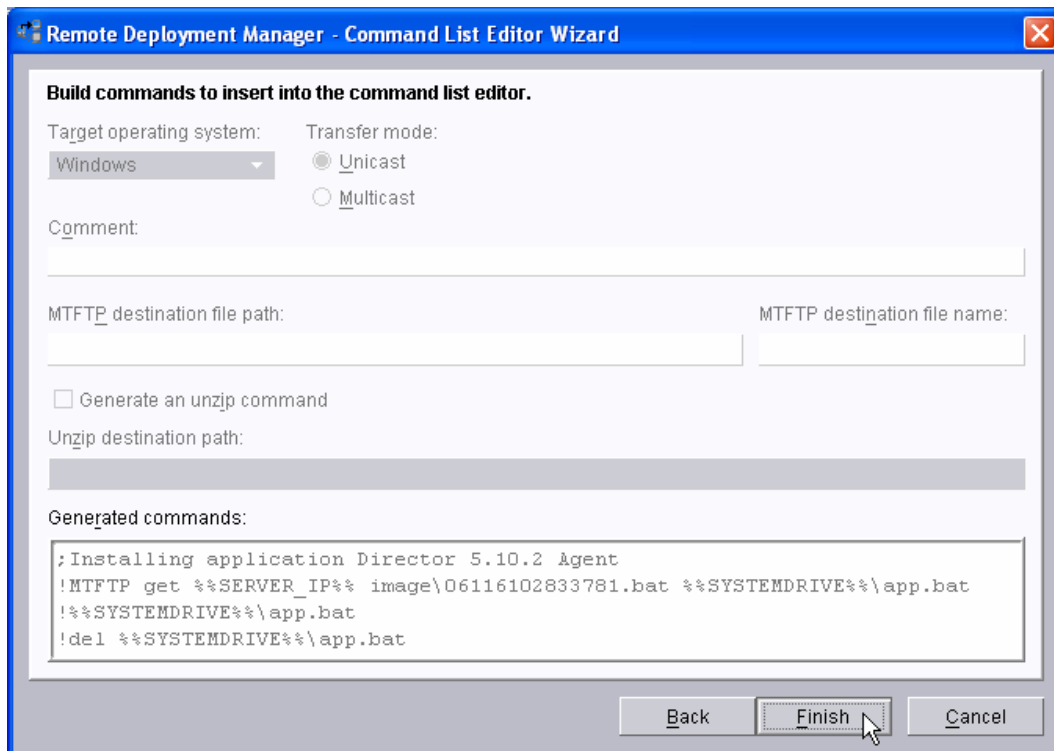
Note: We can use a similar procedure in an RDM *Windows Clone Install* task (see section 4.2.1 on page 52 for a complete description of the procedure).

Here is the general procedure:

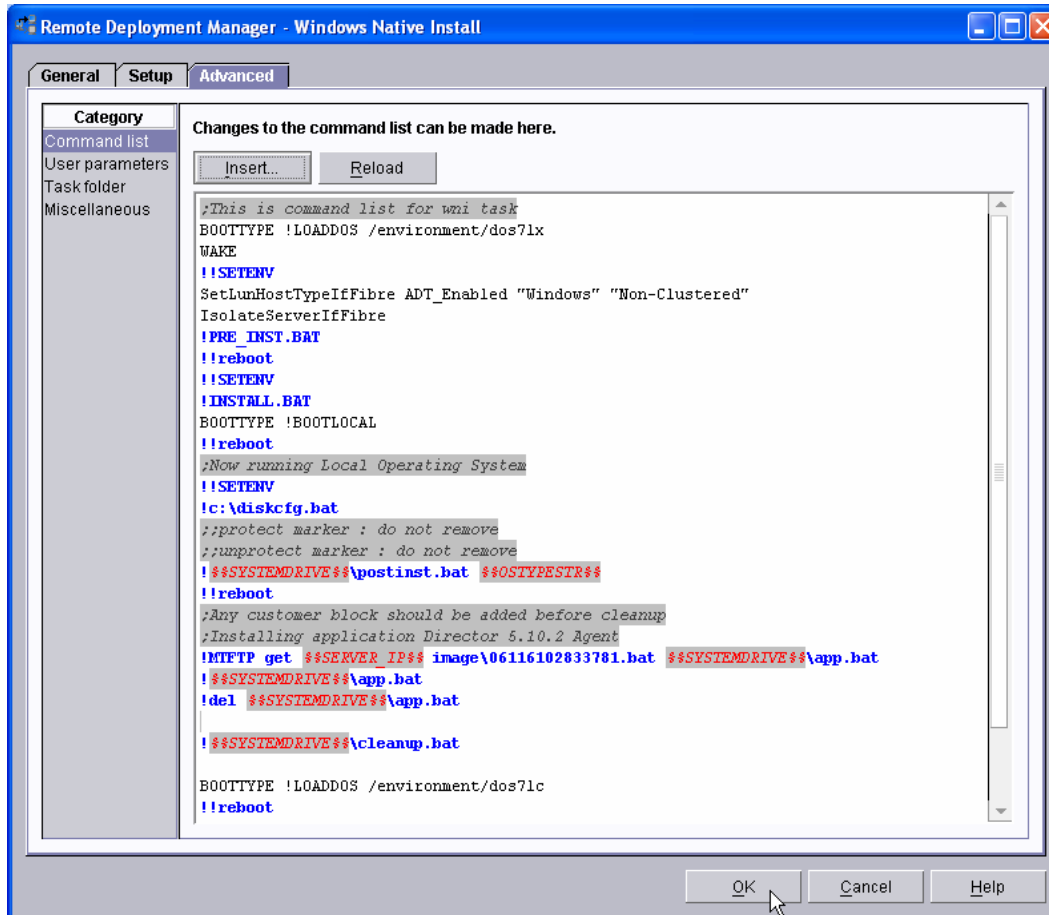
1. Create a RDM *Windows Native Install* application image for each application, using the procedures described in section 3.2 on page 23.
2. Edit the task, and put the cursor on a blank line immediately preceding the %%SYSTEMDRIVE%%\CLEANUP.BAT statement.



3. Click the *Insert...* button to bring up the Command List Editor Wizard window.



- Click the *Finish* button to add statements that download and install the application.



A nice thing about this technique is that you don't have to type anything.

### 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 3.1.4 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:

- 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

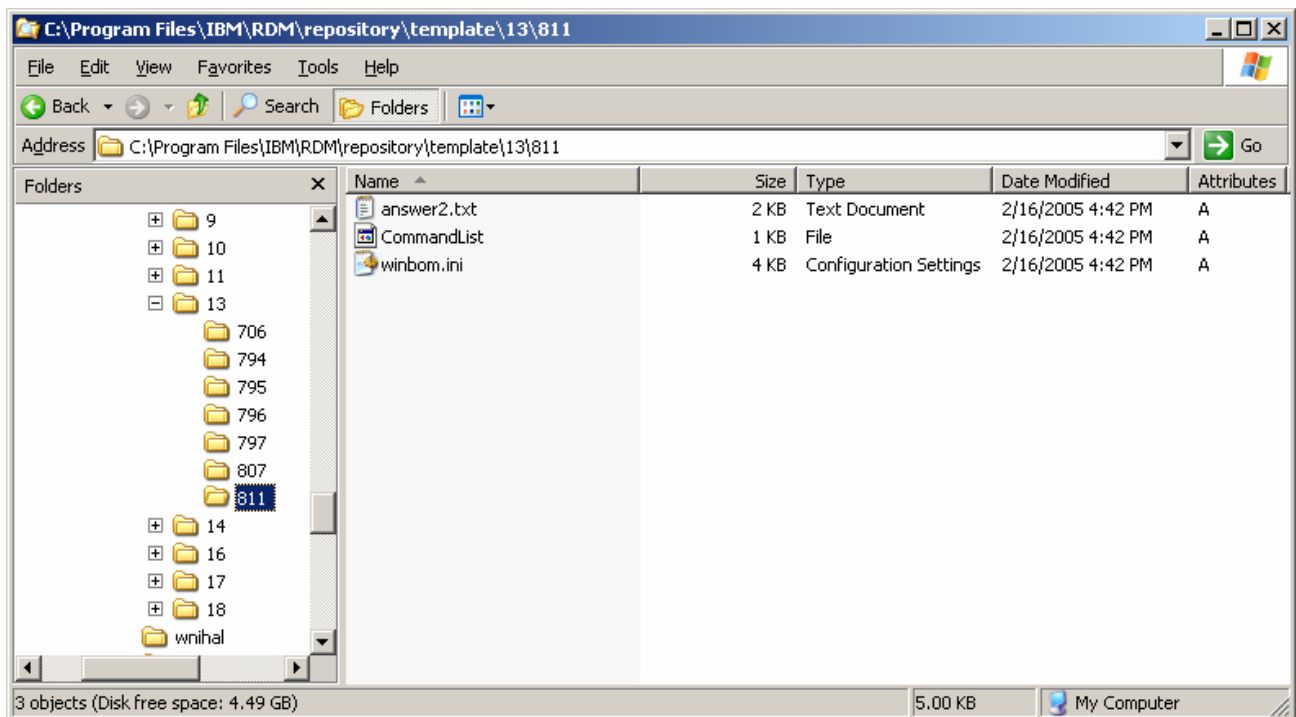
To customize a *Windows Clone Install* task application install, it will be helpful to understand how this task works. In this section, we will explore a typical *Windows Clone Install* task that installs Windows Server 2003 Standard. Assume that we have completed the first procedure outlined in section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

#### 4.1.1 Find the task folder

Use the technique from section 3.1.3 on page 11.

#### 4.1.2 Explore the task logic

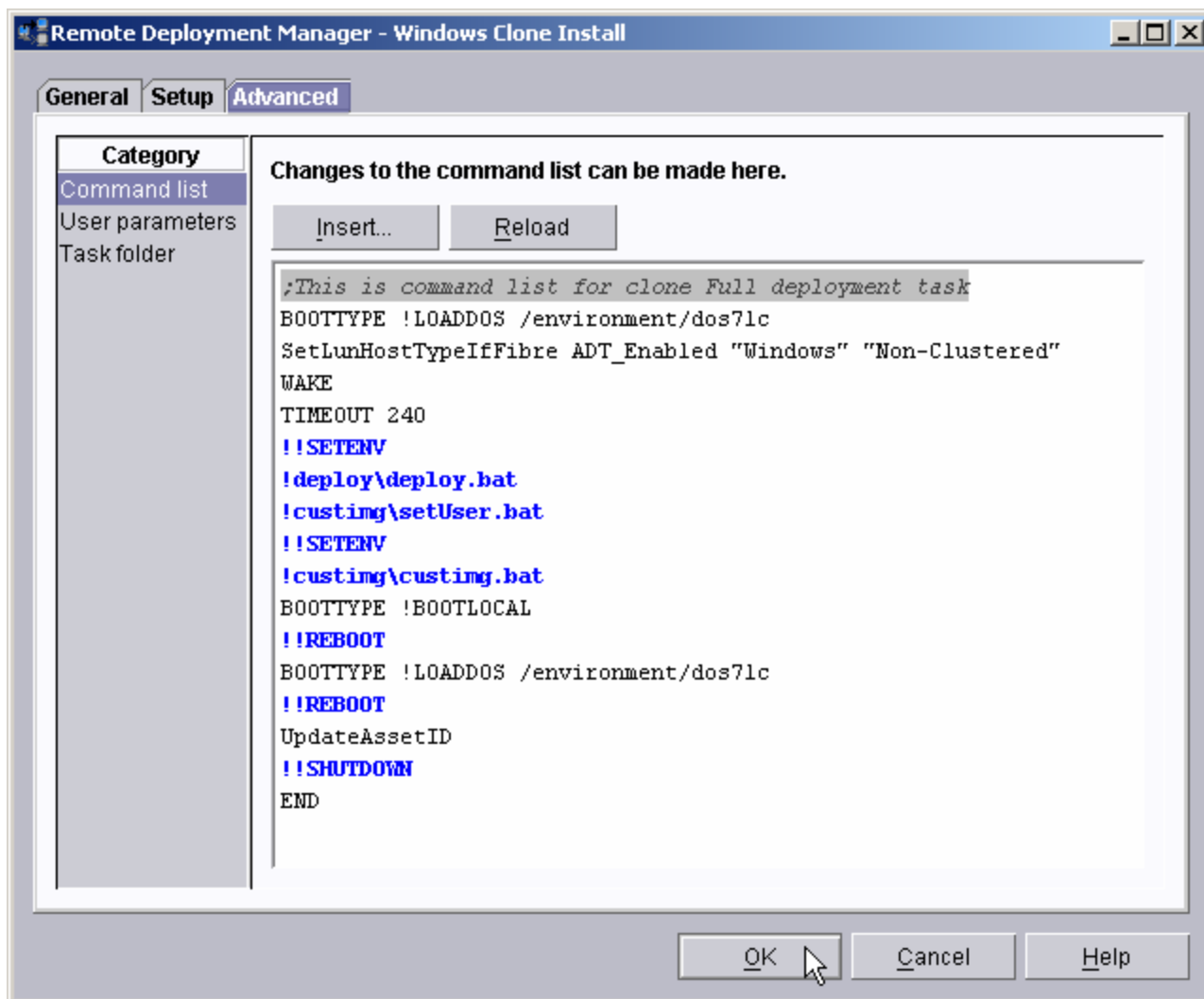
Open Windows Explorer to view the files in the task folder.



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 2 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 2 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).



```
;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
```

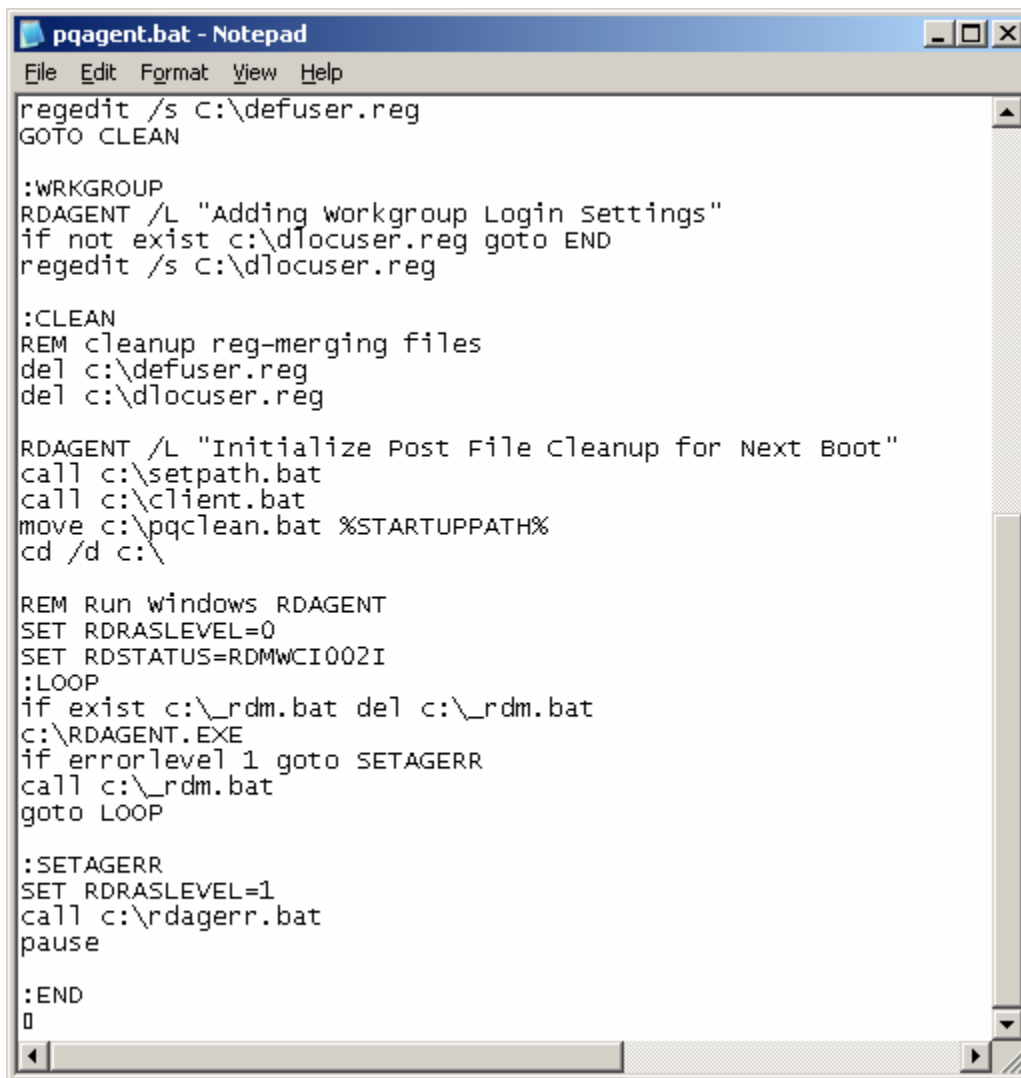
line results in a download of a batch file `_rdm.bat` into the current directory (which is currently assumed to be C:\). The next line,

```
call c:\_rdm.bat
```

runs the next command from the task's CommandList file. You can find this file in the

```
C:\Program Files\IBM\RDM\local\env\71c\custimg
```

directory.



```
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.

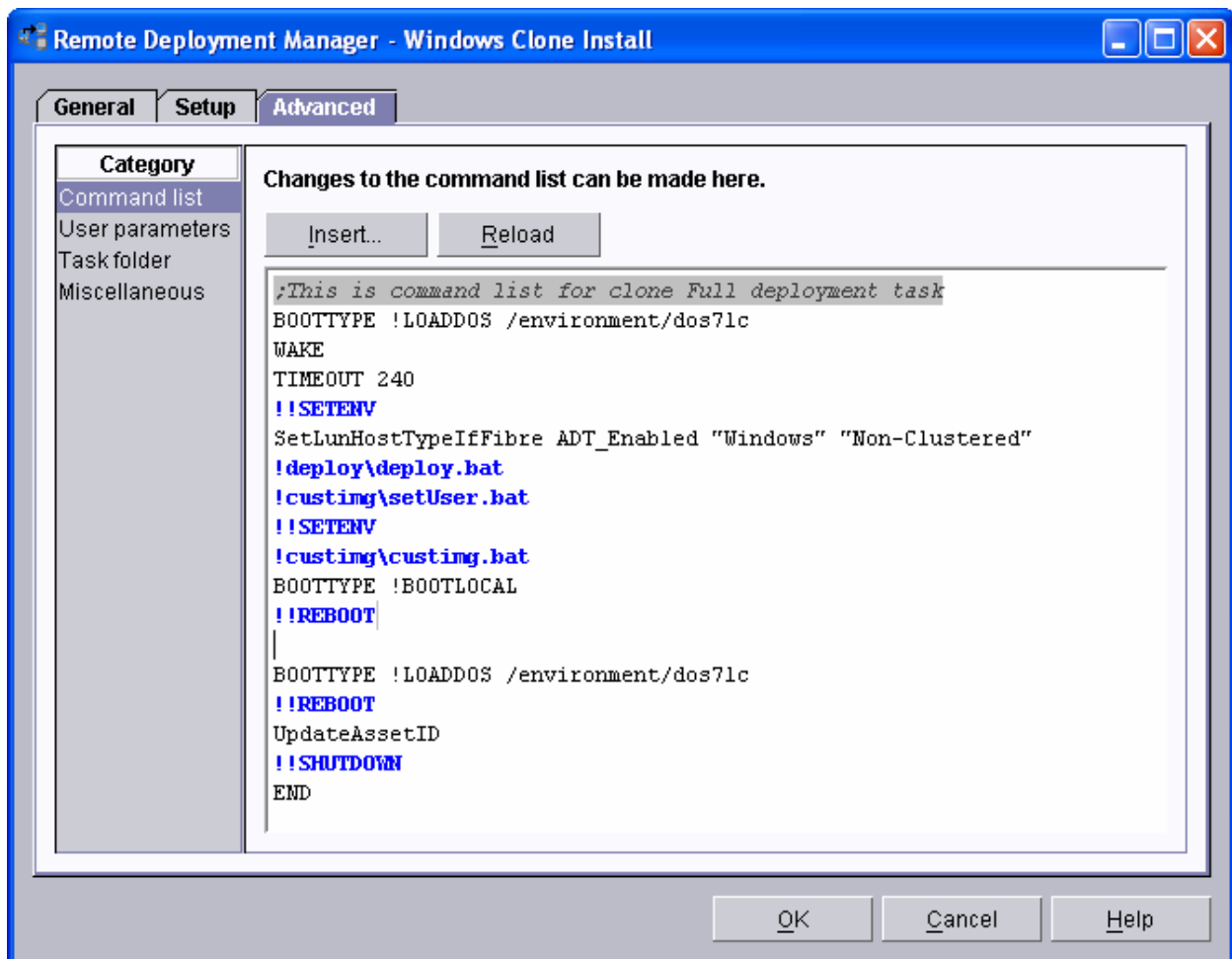
One scenario where this is desirable is when your current *Windows Clone Install* task installs Windows plus a set of applications, but you have some new applications that you want to add to the task. Instead of rebuilding the task from scratch, using the procedure in section 2.3.1 on page 8, you can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

Another scenario where this is desirable is when you need an application that does not clone well. An example of such an application is IBM Director Agent. Although it is possible to include IBM Director Agent in your clone image, it requires that you make several modifications to RDM (see the RDM 4.30 User's Reference for details). It is much easier to leave the IBM Director Agent out of the clone image, and to install it as described below.

Enhancements in RDM 4.30 make this procedure much easier than it was in prior RDM releases.

### 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 48 in section 4.1.2.1 above).



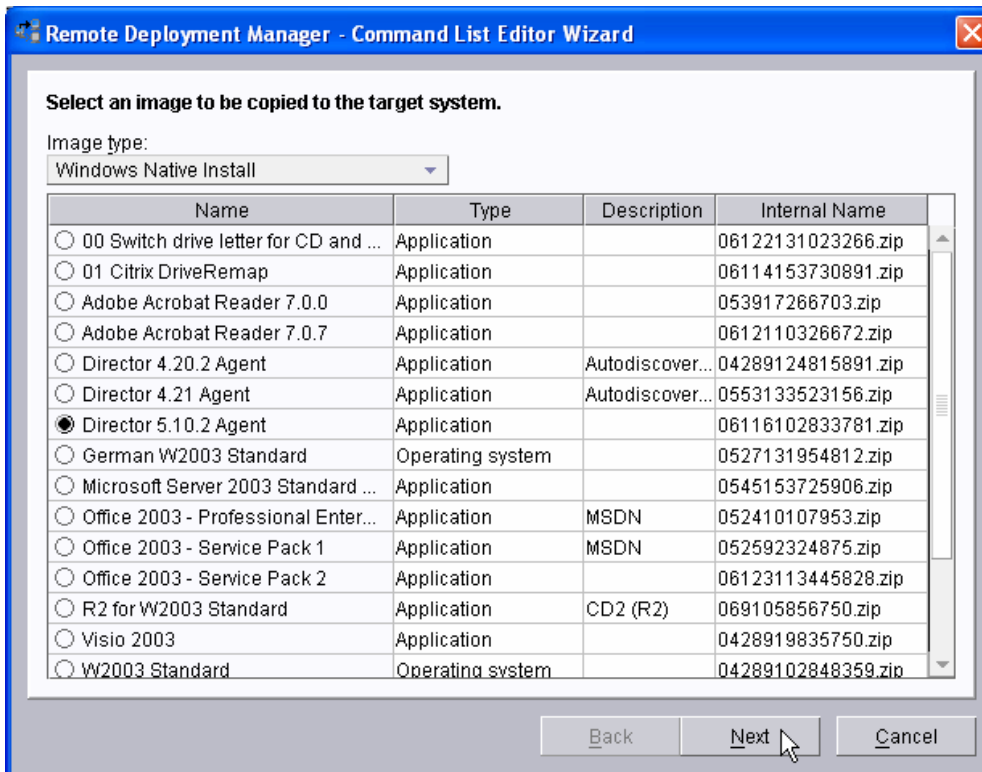
**Important:** One caveat to consider here is that PQAGENT.BAT only runs once. So we have to be able to install all of our extra applications at this point, before the system reboots again. If you have 2 applications whose installs must be separated by a reboot, you cannot install them as described herein.

Here is the general procedure:

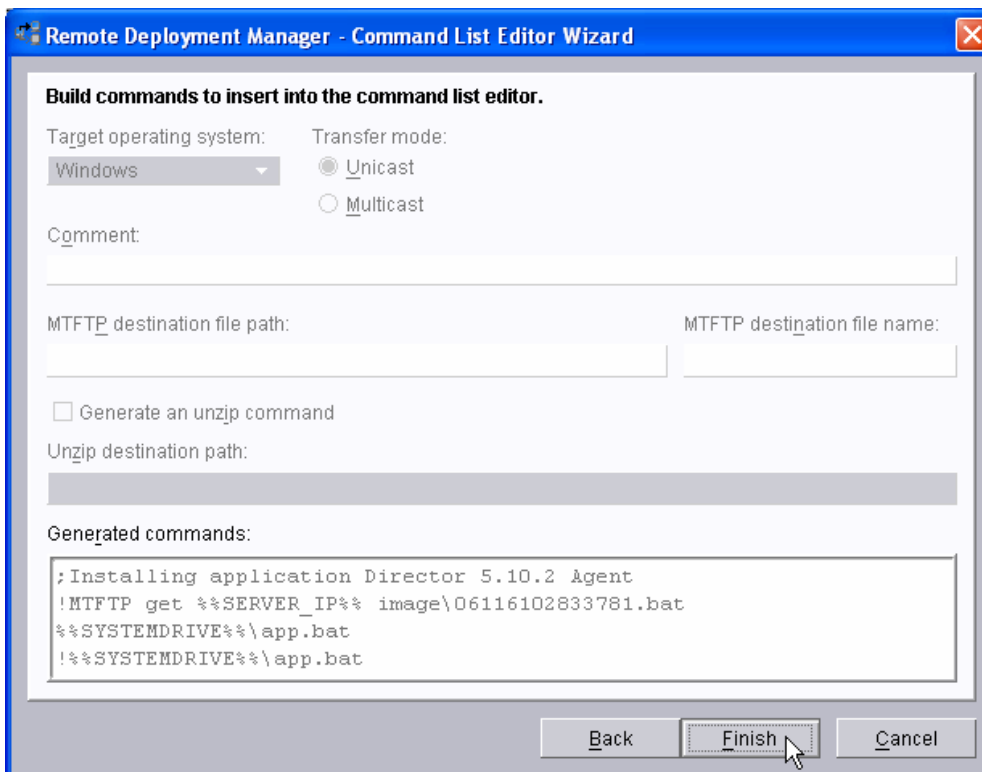
1. Edit your existing *Windows Clone Install* task. Select the *Advanced* page to display the CommandList.
2. Insert a
 

```
!!SETENV
```

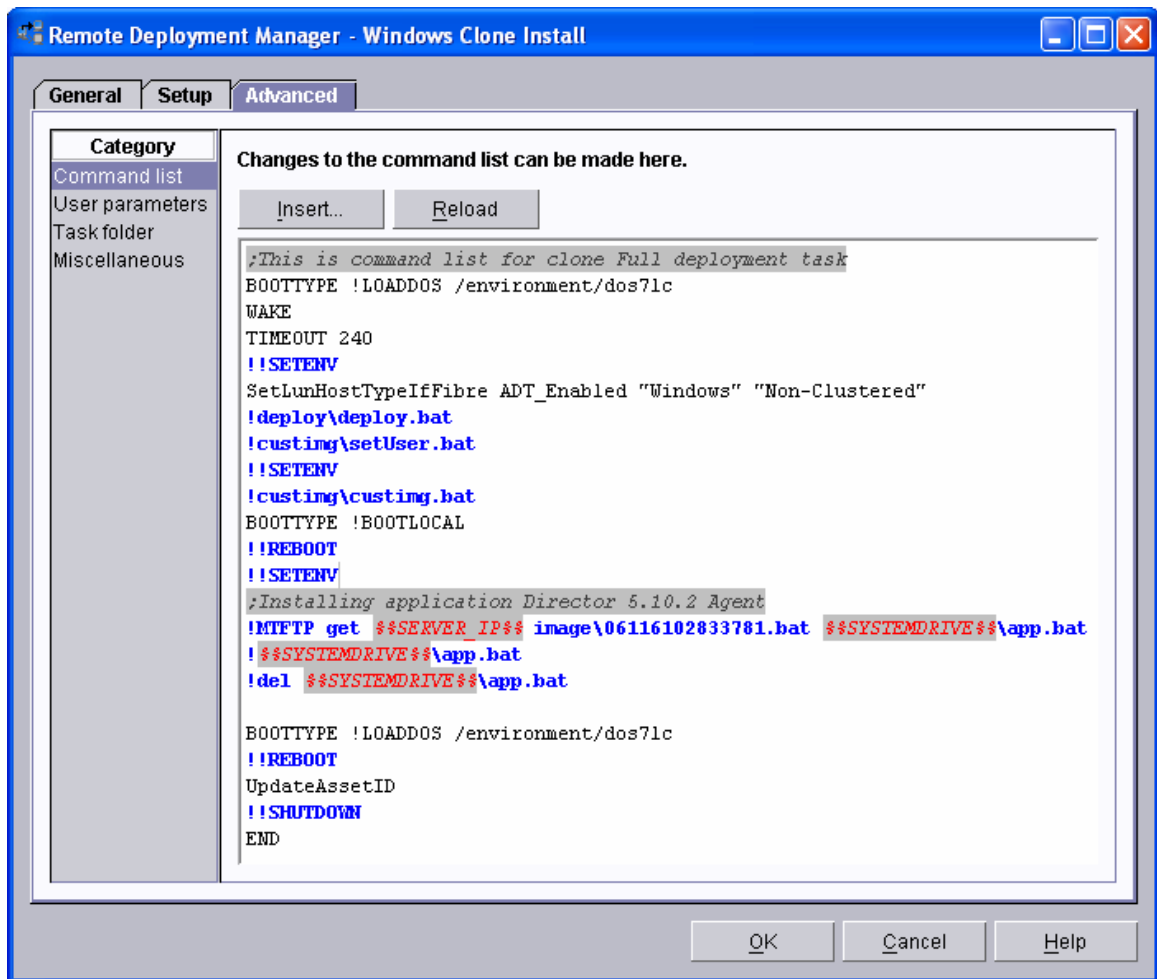
 statement, followed by a blank line at the position shown in the above picture.
3. Put the cursor on the blank line. Then press the *Insert...* button to display the *Command List Editor Wizard* window.



4. 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.

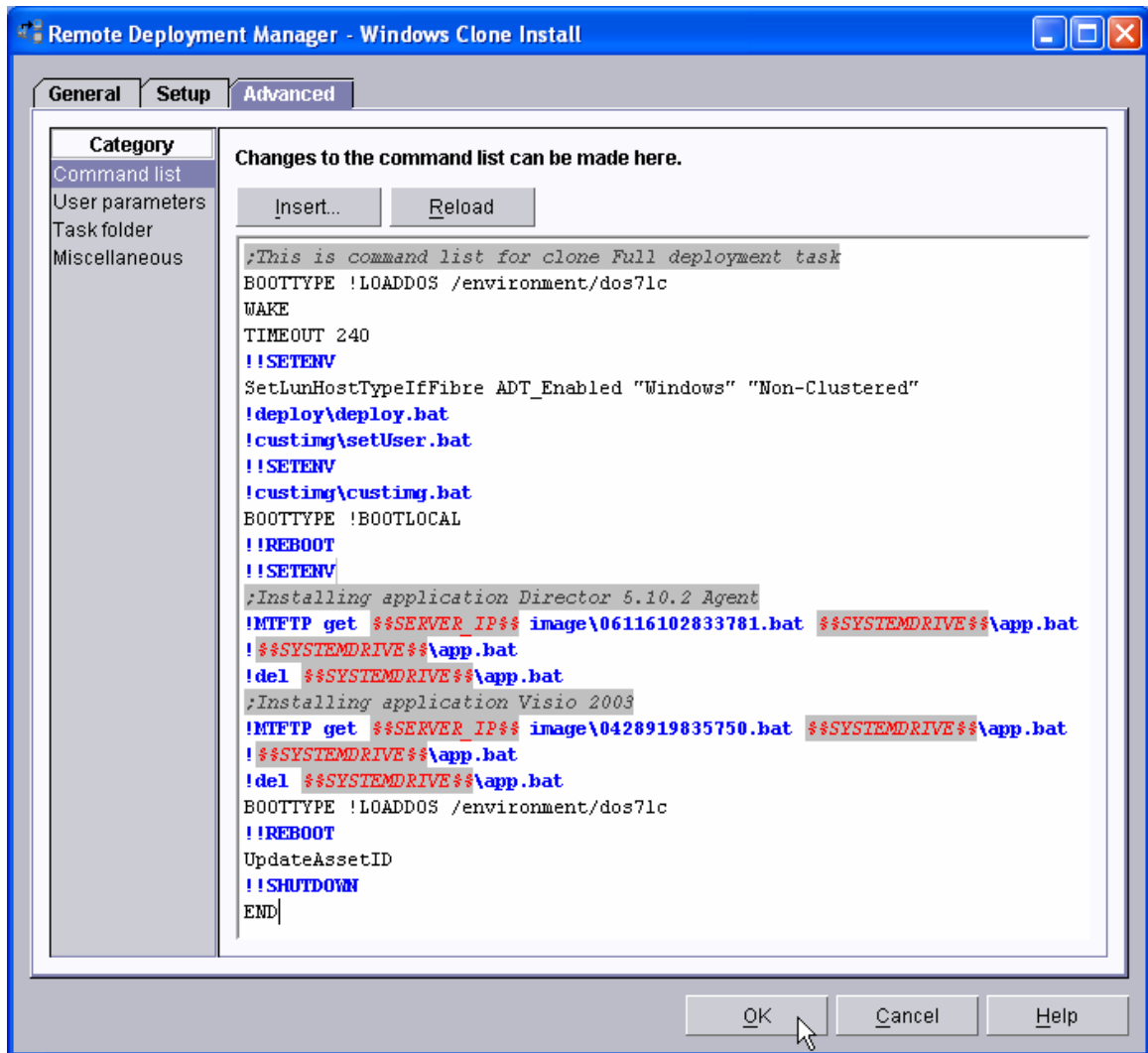


5. Then select the *Finish* button. This will insert the generated commands from the wizard window into the CommandList.



At this point, if we ran the task, it would install the task's Windows image (which also may include other applications) and then install IBM Director Agent. However, we will add a second application.

6. Insert a blank line right before the BOOTTYPE command, and repeat steps 1 through 5 above, selecting the Visio 2003 application. The resulting command list will look like this:



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 IBM Director Agent and Visio.

#### 4.2.2 Install logic

Our technique – installing *Windows Native Install* applications in a *Windows Clone Install* task – is especially effective because it is so easy to set up. It requires minimal data entry, and there are few opportunities to mess it up.





## 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.