realtimepublishers.com™

The Definitive Guide™ To

Windows Desktop Administration

SCRIPTLOGIC

Bob Kelly

Copyright Statement

Chapter 4: Application Management

In the last chapter, we covered the deployment of new systems. With your baseline identified, tested, and deployed, you must now work to keep that baseline up to date with the latest updates and new software requirements. Just keeping Windows itself up to date with service packs and a weekly release of hotfixes is enough to make some a little uneasy. Add to this task requests for new software and a constant barrage of updates, patches, and new releases for your existing software base and the need for a planned process and deployment mechanism becomes clear. In this chapter, we will discuss the creation of a process for deployment as well as the various methods available for the automated installation of software (and the benefits and drawbacks of each). Finally, we will cover a handful of the available software deployment systems with a summary of their capabilities.

The Software Deployment Process

Establishing a process to follow for the deployment of software is a crucial first step to success. When someone hands you a new application that they want sent to everyone in their office, what do you do first? Although larger networks demand a more structured process (largely for accountability), even the smallest of networks should have some sort of approval process. After approval, with which settings should the software be deployed? How should the software be deployed? Which machines should receive the software? These are all questions for which the answer should be crystal clear for every release. To realize this goal, a structured process must be in place.

Particularly when there are several people involved in software deployment, a process can keep everyone in the loop with what is going on. This communication is very important when the user comes back hours later to ask why new software hasn't appeared on his or her desktop yet. Having a documented process to show the user, even if it is a simple bulleted list, will help the user understand that there are steps to be taken—and that the process might take longer than he or she might have expected.

Additionally, a solid process will help to minimize deployments that negatively impact users. The deployment of packages that cause problems for users will hurt the reputation of your package and deployment team—the last thing you need is for users to fear receipt of new deployments and blame every problem they encounter on something you have "done to them."

We will discuss several elements that may be included in the makeup of your own deployment process. Use what makes sense in your own environment, but consider each:

- Receipt
- Integration testing
- Package research
- Package development
- Package testing
- Deployment testing
- Pilot deployment
- Full-scale deployment
- Baseline updates

Receipt

In most environments, there is a board (or individual) that approves software for a network. The request might get to this point through some chain of approval, but odds are you, as the desktop administrator, will not have to approve or disapprove software for use on the network. This, of course, is a good thing; after all, you have better things to do, right?

When software is approved and makes it into your hands to deploy, there are some basic first steps you should take before you start working on a package to deploy the software. The following sections detail these steps.

Deploying the Latest Version

Many organizations make it a policy not to deploy old versions of software. This approach is sensible considering that the update will require a second deployment more rapidly than if you had deployed the latest version from the start. Check the vendor's Web site to ensure that you have the latest version, upgrade, or patch.

One helpful way to track the latest version of software is the online service from TechTracker called VersionTracker. This service notifies users of new versions by comparing the software inventory of a system with the service's online database. See http://www.versiontracker.com for more information.

Existing Older Versions

If a version of the software that you are to deploy already exists on the network, it is likely that this existence will affect how you deploy your new package. In some cases, previous versions must be installed before any new version is installed. If you are deploying software while users are logged on, you might need to ensure that the previous version is not in use before triggering your silent installation. Finally, if you have repackaged a previous version as an MSI setup, you will need to synchronize the component global unique identifiers (GUIDs) of the two packages and properly configure the new MSI as an update of the previous one. For more information about MSI update packages, see the sidebar "Creating Update MSI Files."

Creating Update MSI Files

Vendor-provided MSI files *should* follow Windows Installer rules for updates. However, if you are developing a setup for an internally developed application or repackaging a commercial off-the-shelf (COTS) application, there are some additional considerations for the proper configuration of an MSI update.

The version number of the MSI must be incremented. Major and minor updates are dictated by which version number is incremented. For example, 1.1.0.0 to 1.2.0.0 would be a minor update and 1.1.0.0 to 2.0.0.0 would be a major update. For more information about MSI version numbers, see the article by Matt Goedtel, "The Importance of the Version Field," at http://myitforum.com/articles/6/view.asp?id=5075.

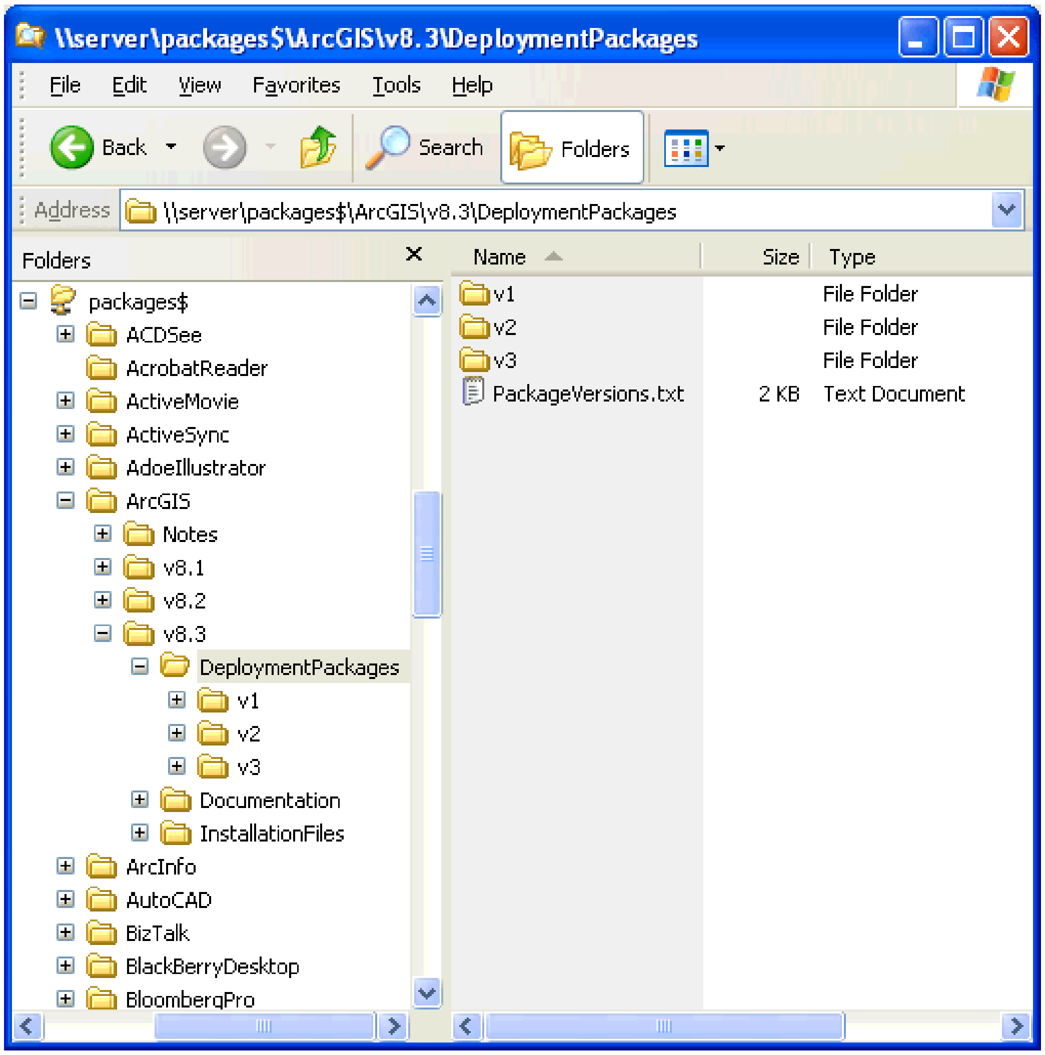
Depending upon the type of update, a varying number of required GUID updates are necessary. GUIDs to be updated might include the PackageCode, ProductCode, and ProductVersion. See the Windows Installer SDK for more details (online at http://msdn.microsoft.com/library/en-us/msi/setup/patching_and_upgrades.asp).

GUID synchronization for like components is also important to the MSI update process. All elements of an MSI package are identified by GUID values. Your MSI authoring/repackaging software randomly generates these values for you. When the same file and version is deployed in one or more packages, such as in an update MSI file, it is important to use the same GUID to identify the files so that the reference counts can be properly maintained by Windows Installer (keeping the system from removing the file from the system during uninstall when another application is still using it). For more information about component, package, product, and upgrade codes, visit http://www.installshield.com/aboutus/news/newsletter/0012_50ef-3.asp.

For more information about Windows Installer, read Jeremy Moskowitz and Darwin Sanoy's *The Definitive Guide to Windows Installer Technology for System Administrators* (Realtimepublishers), a link to which is available from http://www.realtimepublishers.com.

Media Tracking

When you receive installation media (CD-ROMs, floppies, DVDs, and so on), it is a good idea to have a way to store and track them in place. In the pile of CD-ROMs that covers your desk, it can be easy to misplace one, and doing so can obviously be a big problem. Many organizations store installation files on a network share. In fact, doing so is often a necessity because a scripted or command-line install will require the source files to do the job. Put a logical directory structure in place that will help you to quickly identify the software you are looking for in the future. Figure 4.1 shows an example.



Name	Size	Type
v1		File Folder
v2		File Folder
v3		File Folder
PackageVersions.txt	2 KB	Text Document

Figure 4.1: An example package directory structure.

☞ You can save yourself some trouble by avoiding use of spaces in the names of shares, directories, and packages. When a space is used, you will need to enclose the entire path in quotes when calling it in Windows. When a path to a file might be a command-line parameter, enclosing paths in quotes can make complex command-line syntax even more difficult. Eliminating spaces or substituting them with an underscore character (_) can help you to avoid making long command line calls more difficult than they need to be.

With the installation software on the network where you need it, return the CD-ROM to the owner. Some organizations have a media library or configuration management department that stores installation media and tracks licensing. Returning the media once you have what you need minimizes the time it is in your possession. If you must store the installation media yourself, put together a simple filing system to track its point of contact and the date it was obtained. It is also a very good idea to have a place to record the date, a return date, and initials. Then when someone comes to take the CD-ROM back, have that person initial your file for accountability purposes.

☞ Some setups require that the original CD-ROM be used. Either due to a poor setup process or an intentional restriction to deter pirating, it might be necessary to repackage the application directly from the installation media. You might be able to “trick” setup into thinking it is being installed from a CD-ROM by sharing the folder in which you have stored the media and mapping a drive letter to that location. Doing so will make it appear at root, like it would if you were installing from a CD-ROM. Still other setups check for a volume label as a condition of installation. To satisfy this restriction, you must set the volume label for the drive containing the shared folder. If you are repackaging, just work from the removable media to create your package. If you are working to script the installation, steps like these might be necessary depending upon the limitations accidentally or deliberately imposed by the vendor.

Integration Testing

If you have different versions of Windows on your network or have placed additional security restrictions on your workstations, the software might have difficulty running at all. Before you create and start troubleshooting what might appear to be a problem with your deployment package, first ensure that the application is able to operate within your unique environment when installed manually.

Installation and Usage Documentation

Most software installation wizards provide some number of prompts, selections, and buttons to be addressed during the installation process. If you simply install the software as you think it should be installed, you will inevitably find yourself in situations in which the package does not reflect the desires of the user requesting the software.

As long as integration testing must be done, taking the time to do so with the user requesting the software can be a valuable exercise. Install the software with the user, and document each selection. Although the direction might simply be to “accept all defaults,” the user might not be truly aware of what the default settings entail. Some installation dialog boxes might even require that you make a decision that does not provide a default selection—what then? Screen shots of the installation wizard might be overkill, but a simple bulleted list of what was done during installation can let others see the process and might even help you 200 packages later when you have long since forgotten the specifics of the installation.

Another piece of information easy to overlook is the file used to launch the installation process itself. It is not always as simple as a setup.exe on the root of a CD-ROM. There are times when an installation media might have several directories and setup files. Having clear documentation of the one you used can save time otherwise spent hunting around for the proper file. The following list details a very simple example of a software installation document:

- Start install: \bin\eaxsetup.exe
- Accept EULA
- Accept default installation directory
- Begin installation
- OK to reboot

Application Interaction Considerations

What will the impact be on other applications? Many applications depend on components such as the Java Runtime Environment (JRE), Oracle Client, and Microsoft Data Access Components (MDAC). The version (or in some cases the location) of such an installation might significantly affect other applications on a system.

Particularly in situations in which there is an internal development staff or companies are contracted to write custom software for an organization, it is imperative that your baseline configuration be communicated. The version, location, and settings for elements such as MDAC are important to consider early in the development stages. Guiding *custom development* to function in your *custom environment* is an often-overlooked aspect of internally developed or contracted applications. Establish your requirements up front and avoid problems in deployment and compatibility down the road (for more information about how to do so, see the sidebar “Establish Installation Requirements.”)

Establish Installation Requirements

When custom software is developed for an organization, it is naturally the functionality of the application that is of paramount concern—so much so that little attention is given to installation, which can cause great expense when you begin to attempt to integrate the application into your environment. Reengineering or repackaging the setup of an application developed especially for your organization is a wasteful process that introduces unnecessary risk. Some points to consider:

- Provide documentation about your build process. If an image is used, provide a copy for use by your internal development staff or contracted companies. If your image only works on certain hardware, give contractors or developers (or have them purchase their own) compatible hardware. It is important that developers understand the environment in which their applications will be deployed in order to avoid assumptions that might cause alteration for the application to comply with reality.

- Provide system and Group Policy information, particularly those that enforce security settings. If areas of your baseline hard drive might not be written to by users, make sure that the development team understands where they *may* store data or configuration files. Often working in an environment in which users function as administrators of their workstations, it might not be apparent that in your environment users may not write to common user areas, particularly the HKEY_LOCAL_MACHINE hive of the registry. It is important that this possibility be communicated to avoid modifications to your baseline security in order to support the application.

- As stated previously, MDAC, JRE, ODBC, and Oracle Client are just some of the components that applications may be dependant upon. For example, if your environment is at MDAC 2.6, it is important that developers not develop their application using functionality not available until MDAC 2.7.

- Demand a Windows Installer installation package. MSI is well documented with several powerful tools available to create installation packages. It is a very “administrator friendly” format that provides you with maximum control and deployment options. Of course, a bad MSI package can cause problems as well—require that Windows Installer validation tools be used to ensure that MSI rules are followed. In particular, I recommend you require MSI validation be performed using the Independency Consistency Evaluator (ICE). This tool is a feature of most MSI tools and is available free in the Windows Installer SDK.

Package Research

Although many organizations will restrict themselves to one or two packaging (automated installation) methods, it’s wise to review all of your options before moving forward. Being consistent about creating packages for your software deployments has its benefits—for example, you might document and become very proficient with the method you choose.

Some environments are restricted to the use of proprietary repackaging formats (potentially a limit of your deployment system). For example, although not a proprietary format, environments that rely upon the assignment of Group Policy Objects (GPO) as a method of deploying software are restricted to using the Windows Installer (MSI) package format. In a situation such as this, in which you have a single means of deploying software, you might well need to focus your packaging on one particular method. In the following section, we will cover several common packaging methods.

MSI

Microsoft's Windows Installer (MSI) setup format was first introduced with the release of Microsoft Office 2000. This introduction provided a documented standard for installation with rules, structure, and format. MSI files are databases of installation components and instructions that all follow the same rules regardless of the tools used to generate them. Among quite a few other features, MSI files offers support for a command-line driven, silent installation using the following options:

- /qn—No user interface whatsoever
- /qb—Basic user interface (progress bar only) with no modal dialog boxes
- /qb!—Same as previous listing, but with Cancel button disabled

In addition to these options for silencing installation from the command line, you can use public properties (variables within an installation that can be dictated at the command line) to customize the installation even further. The following list provides several properties that might be helpful in a command-line installation (for more information about these properties, see the Windows Installer SDK):

- REBOOT—Dictates how the system handles reboot requirements by dictating whether Windows Installer should force, suppress, or *reallysuppress* the reboot action. The use of force is clear, but the difference between suppress and reallysupress is that suppress will cause an automated reboot if the installation is being run silently (reallysuppress bypasses the reboot action regardless of how the MSI is being run).
- ALLUSERS—Unlike legacy setups that specify a full path to where shortcuts and configuration information are to be stored, MSI handles this itself. Depending upon whether the installation is being performed for the user or the computer, shortcuts are placed in the user profile or the All Users profile as appropriate. The ALLUSERS property allows control of this behavior from the command line.
- INSTALLDIR—Specifies the directory in which the application should be installed. A default directory will always be specified within the MSI, but you may alter this location using the INSTALLDIR property.
- ROOTDRIVE—By default, the Windows Installer will choose the fixed drive with the largest amount of free space as the destination drive for an installation. To override this behavior, you can specify a drive letter (include a trailing backslash when specifying a value for this property).

- ARP—Several properties are available for controlling how an installation is presented within the Add/Remove Programs applet. For more documentation about properties that control the behavior of the Add/Remove programs listing for an application, see the AppDeploy.com FAQ listing on the subject at http://appdeploy.com/faq/detail.asp?id=58.

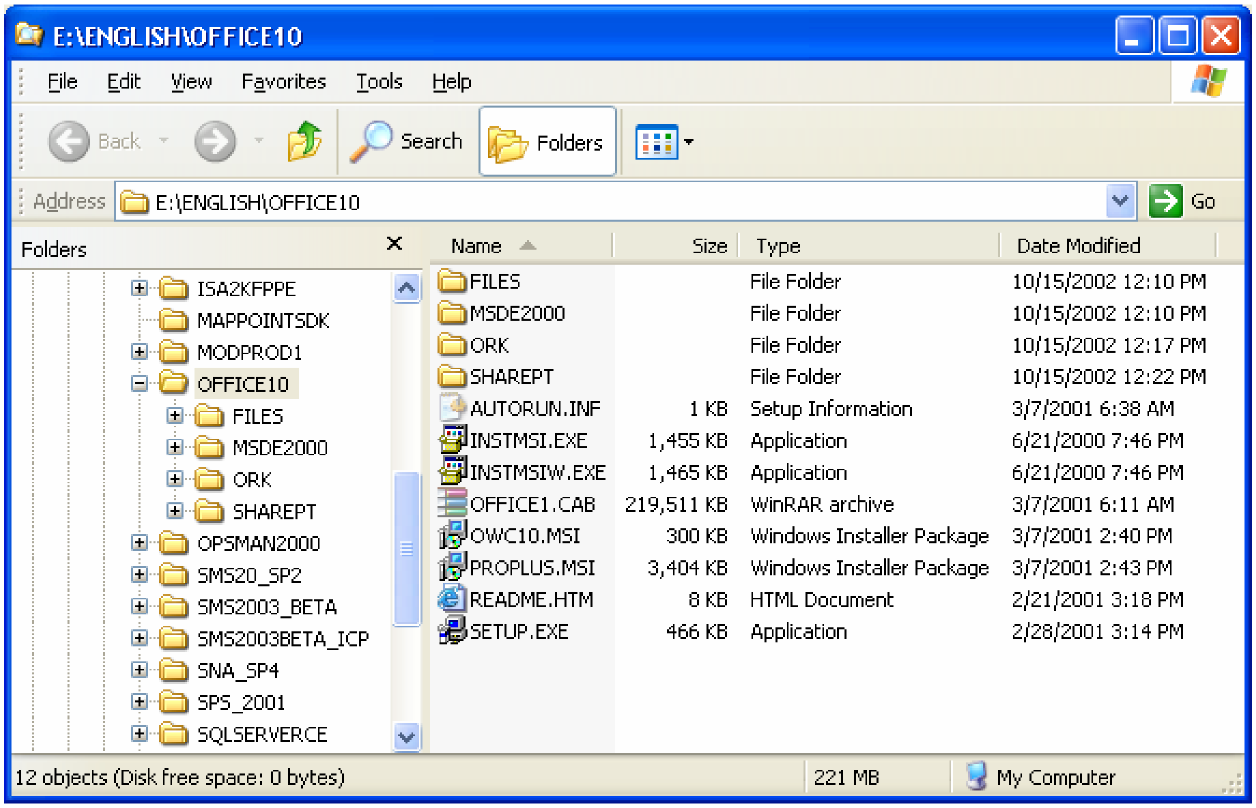
Public properties are always identified in upper case. It is these public properties that you can set at the command line.

Environments that utilize AD's Group Policy as a means of software distribution are limited to this format when assigning software to computers. Because many software vendors have not moved to the MSI setup format, administrators today often repackage installations into this format. Repackaging is covered later in this section.

For more information about Windows Installer, see Jeremy Moskowitz and Darwin Sanoy's *The Definitive Guide to Windows Installer Technology for System Administrators* (Realtimepublishers) at http://www.realtimepublishers.com.

There are an increasing number of vendors making their software available in the MSI format. The development of MSI packages can be a complex process, which has been the primary reason this format has taken so long to catch on. As a bit of encouragement, Microsoft has made the availability of an MSI setup a Win2K and Windows XP logo requirement. When it comes to the MSI setup format, there are no specific implementations; all kinds of software from games to business software are available as MSI setups.

Is that setup an MSI? Just because you see a setup.exe at the root of your installation CD-ROM does not mean it is not an MSI setup. Because people expect to see a setup.exe, and because not every machine has the Windows Installer service installed, a setup.exe is often provided. The executable first checks to see whether the required version of Windows Installer is present on the machine. If it is not, it gets installed and the MSI setup is initiated. If the desired version is present, it simply calls the MSI setup. If the MSI file itself is stored in a subdirectory rather than the root of the CD-ROM, the situation can be confusing. A giveaway that an MSI setup is initiated is, of course, the display of a Windows Installer dialog box when the setup is launched; you can also be reasonably sure you are dealing with an MSI if you see the Windows Installer setup files in the directory with the setup.exe (see Figure 4.2). There are two such files, one for Windows 95/98/ME systems and one for NT, Win2K, and Windows XP systems: InstMsiA.exe and InstMsiW.exe, respectively.



Name	Size	Type	Date Modified
FILES		File Folder	10/15/2002 12:10 PM
MSDE2000		File Folder	10/15/2002 12:10 PM
ORK		File Folder	10/15/2002 12:17 PM
SHAREPT		File Folder	10/15/2002 12:22 PM
AUTORUN.INF	1 KB	Setup Information	3/7/2001 6:38 AM
INSTMSI.EXE	1,455 KB	Application	6/21/2000 7:46 PM
INSTMSIW.EXE	1,465 KB	Application	6/21/2000 7:46 PM
OFFICE1.CAB	219,511 KB	WinRAR archive	3/7/2001 6:11 AM
OWC10.MSI	300 KB	Windows Installer Package	3/7/2001 2:40 PM
PROPLUS.MSI	3,404 KB	Windows Installer Package	3/7/2001 2:43 PM
README.HTM	8 KB	HTML Document	2/21/2001 3:18 PM
SETUP.EXE	466 KB	Application	2/28/2001 3:14 PM

12 objects (Disk free space: 0 bytes) 221 MB My Computer

Figure 4.2: Directory of MSI Installation with Setup.exe and Windows Installer support files.

You should never repackage an MSI setup. You might be able to generate a successful installation by doing so, but upgrading or removing the installation will cause you big trouble down the road. Before repackaging, make certain you are not dealing with an MSI setup. Self-extracting archives that contain MSI setups can trick you into thinking that the setup you are working with is not an MSI. See the sidebar “Accessing Source Files Within Self-Extracting Executables” later in this chapter for more information.

Benefits

Complete control of an MSI setup can be realized without modifying the vendor’s MSI directly thorough the use of transform (MST) files. You can use an MST to specify which features to include, move or delete shortcuts, and add files or registry entries. Some software vendors have even provided MST creation wizards to facilitate the customization of your deployment. Microsoft Office provides a Custom Installation Wizard to generate an MST, and many software vendors have licensed InstallShield’s Tuner product to provide similar functionality, including IBM’s Lotus Notes 6 and NetManage’s RUMBA.

Self-healing and install-on-demand are two powerful new features Windows Installer brings to the table. Windows Installer shortcuts check for key files when launched and will attempt to retrieve any missing files automatically. In this way, an application installed with MSI may also be configured not to install certain features until the user attempts to make use of them.

In addition, Windows Installer provides excellent command-line support, including the ability to set public properties. This feature provides you with extensive command-line control of an installation that will translate to every installation provided in this format.

Drawbacks

Windows Installer must be properly administered or it can cause user dissatisfaction in the way of repetitive calls to repair installations or prompting for needed installation files. In a network environment, you can avoid these kinds of problems through conflict checking and ensuring a resilient path to installation files. However, Windows Installer requires an increased level of planning and administration.

Updating an MSI can be technically challenging due to Windows Installer's requirements to recognize and properly perform an update. See the sidebar "Creating Update MSI Files" earlier in this chapter for details.

A common misconception is that a patch file (MSP) is a way to quickly affect small changes after deployment; however, the creation of an MSP is actually additional (and as an administrator, often needless) work. To generate an MSP, you must first create an entire MSI for the new software update. The MSP is then generated by comparing the old and new MSI packages so that the file's size is reduced to include only the changes between the two. This is primarily done for reducing Internet download sizes or the need to move updates across a WAN link.

Command-Line Installations

Particularly with simple installations that require little or no configuration options, some software vendors provide a means of silently installing software from the command line. Microsoft, for example, provides command-line options for installations of hotfixes and other simplistic installations using a consistent set of command-line switches. The following list provides common Microsoft command-line switches:

- **/A** runs the setup in administrative mode. This option is available only when installing from the original media.
- **/B** specifies the type of setup to be performed. This can be a number, which refers to Typical, Compact, or Custom. The meaning of this number varies across Microsoft products.
- **/C** "" If your product was supplied with a 20-digit license key, use this option to enter the key and bypass the key validation dialog box. You must enter a space before the key and enclose it in double quotation marks.
- **/K** "" If your product was supplied with a 10-digit license key, use this option to enter the key and bypass the key validation dialog box. You must enter a space before the key and enclose it in double quotation marks. The license key can be found on the Product Authorization Certificate or on the CD-ROM casing.
- **/F** performs an installation using only short (8.3) file names.
- **/Q [0 | 1 | T]** runs the setup in silent installation mode. Use /Q1 to suppress the Exit dialog box. Use /QT to suppress all dialog boxes including the background frame window and progress gauge. The /A and /Q options are mutually exclusive.
- **/R** reinstalls the application.
- **/U[A]** Uninstalls the application. If /Q is also specified, the user is not prompted about removing shared components. /UA removes shared components without prompting the user.

Several other common switches can be attempted in an effort to determine support for command-line switches by setup.exe files. Several regularly used switches worth trying out include:

- -/QUIET
- -/UNATTENDED
- -/S
- -/Q
- -/SILENT
- -/SMS
- -/?

☞ Some switches are case sensitive and/or require a dash (-) or slash (/) depending upon their unique implementation of command-line switch support. Try several combinations before giving up on this option as a viable method of silent installation.

Microsoft hotfixes and simple utilities often support command-line installation switches where an MSI was not implemented. In addition, this installation method is often supported by applications designed for distribution (often with built-in distribution mechanisms), including Norton and McAfee antivirus software and management software such as the Altiris, BMC Patrol, Lanovation, and Cognet client agents.

Benefits

If your environment allows it, you should always take advantage of the options developed by the vendor. Success will be as likely as with a manual installation, resulting in your best chance at a clean install. Additionally, such an installation method is supported by the vendor, so you will be able to review documentation and make use of other vendor support services should you encounter any problems.

Drawbacks

Command-line installations often provide few options for customization (or none at all). If you want to install to a specific directory or to determine where any potential shortcuts are to be placed, you might need to use a custom script to perform such changes following the command-line install. As a result, many command-line installations take advantage of a configuration file, or answer file, as discussed next.

Answer File

An answer file lets you customize your installation and still enjoy the benefits of a vendor-supported installation process. Answer files are normally in an INI file format, but can also be as simple as a text file with one or two lines (such as where it should be installed).

To satisfy the demand of desktop administrators, many vendors have taken an effort to provide a supported means of silent installation. Such support provides you with the ability to customize a silent installation using the vendor's supported installation process. For example, InstallShield provides a native ability to create an answer file, then use that file to facilitate a silent installation. Many vendors are aware of this functionality, and have documented it as a supported means of installation. However, many other vendors who used InstallShield unknowingly provide this functionality—so it may or may not be supported by the product with which you're working. Still others who were aware of the capability and did not want to provide it, have broken the process so that it cannot be used.

There are several switches supported by InstallShield legacy setup.exe files. In particular, you will want to use the -R switch to create your answer file and -S to perform the installation silently using this answer file. Supported switches are described in the following list:

- **-S** initiates a silent installation using the answer file (.ISS) in the current directory or in the location specified by the F1 switch.
- **-SMS** holds the process until the installation is complete. When scripting or using a management system such as Microsoft SMS that maps a drive only for the duration of the installation, you should use the -SMS switch. Without it, control of the system is immediately relinquished and the only means of monitoring its activity will be via Task Manager's Process tab. This switch is case sensitive.
- **-R** records the dialog boxes and choices made during an installation process. By default, the answer will be named SETUP.ISS and will be stored in the System Root directory (typically C:\windows or C:\winnt). You can use the F1 switch to specify this location and file name.
- **-F1<path to ISS file>** specifies an alternative location and name of the response file (.ISS file).
- **-F2<path to log file>** specifies an alternative location and name of the log file created by the silent InstallShield installation. By default, a setup.log log file is created and stored in the same directory as that of the InstallShield setup.iss file.
- **-m<filename>** causes setup.exe to automatically generate a Management Information Format (.mif) file at the end of the setup (used by Microsoft SMS for reporting inventory and status messages). Do not include a path—the .mif file is always placed in the Windows folder.
- **-uninst** runs the setup as an uninstall.
- **-verbose** provides more detailed information when a setup.exe error occurs.

☞ You can pass switches in any order and precede them with the dash (-) or forward slash (/) character. With the exception of the SMS switch, these parameters are not case sensitive. When using long path and file names and those with spaces, enclose the switch in double quotation marks so that they are not treated as command-line delimiters.

The following list walks you through the process step by step:

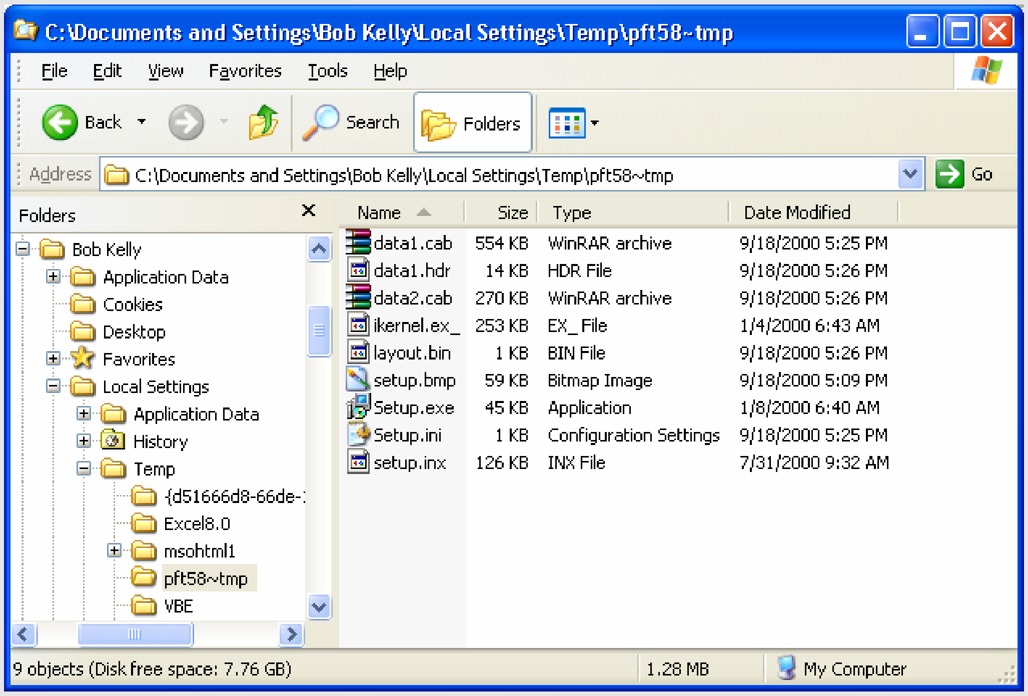
1. Execute

   ```
   "setup.exe -r"
   ```

   at the command line. Note that the program will actually be installed on the system on which you perform this action.
2. Answer all dialog boxes through installation. If an option to reboot is presented at the end, you may click OK, but be aware that running the created answer file will reboot the system as well. You might want to specify that the installation wizard not initiate a reboot so that you can control it yourself using a script or utilizing the capabilities of your deployment tools.
3. Go to your system root directory, for example "c:\windows" and locate a file named SETUP.ISS.
4. Copy SETUP.ISS and the installation files (setup.exe and supporting directory structure) to a network share.
5. A simple batch file containing "setup.exe –s –SMS" will initiate a silent installation of the software using the SETUP.ISS file you have stored with setup.exe.

Accessing Source Files Within Self-Extracting Executables

Is your setup.exe several megabytes in size? This file is likely a self-extracting archive, which decompresses itself into the current %TEMP% directory prior to execution. To access these files so that you may attempt this answer file process, simply launch setup.exe (with no switches) and go to the %TEMP% directory to locate the extracted setup files. You can easily identify the %TEMP% directory using the %TEMP% environment variable—on Windows 9x systems, it is typically equal to C:\windows\temp, and on Win2K and later systems, it is typically equal to C:\documents and settings\<user name>\local settings\temp. The files are extracted to a subdirectory created in %TEMP% with a small randomly generated folder name as Figure 4.3 illustrates.


C:\Documents and Settings\Bob Kelly\Local Settings\Temp\pft58~tmp

Name	Size	Type	Date Modified
data1.cab	554 KB	WinRAR archive	9/18/2000 5:25 PM
data1.hdr	14 KB	HDR File	9/18/2000 5:26 PM
data2.cab	270 KB	WinRAR archive	9/18/2000 5:26 PM
ikernel.ex_	253 KB	EX_ File	1/4/2000 6:43 AM
layout.bin	1 KB	BIN File	9/18/2000 5:26 PM
setup.bmp	59 KB	Bitmap Image	9/18/2000 5:09 PM
Setup.exe	45 KB	Application	1/8/2000 6:40 AM
Setup.ini	1 KB	Configuration Settings	9/18/2000 5:25 PM
setup.inx	126 KB	INX File	7/31/2000 9:32 AM

9 objects (Disk free space: 7.76 GB) 1.28 MB My Computer

Figure 4.3: Directory structure of temp with an InstallShield temp directory present.

As stated, the vendor might be unaware of this capability or might have intentionally crippled this capability. One field commonly used to hide support is a license string field. Because the vendor does not want their software installed on multiple machines with the same license string, this field may be hidden from capture during the "-r" (record) process. Another common reason for failure is the presence of a dialog box during the silent installation that was not presented when you created your answer file. For example, if Netscape is on your system, you might be asked whether you want to install a plug-in, whereas, without Netscape, the question is never presented. For this reason, you must be sure to test this method of silent installation thoroughly to be confident of success.

Because the installation is performed silently, it is difficult to determine the cause of a problem when one is encountered. The setup.log file that is created contains a section titled ResponseResult. This section includes a ResultCode value that will contain one of the following values when complete—typically your only clue in determining what might have gone wrong when testing your installation. Table 4.1 presents the result codes.

ResultCode Value	Definition
0	The installation was successful
-1	A general error was encountered
-2	An invalid mode was specified
-3	Required data was not found in the ISS file
-4	The system does not have enough memory available
-5	A specified file does not exist
-6	The setup cannot write to the response file
-7	The setup was unable to write to the log file
-8	An invalid path to the ISS file was specified
-9	An invalid list type was specified (string or number)
-10	A specified data type is invalid
-11	An unknown error was encountered during setup
-12	Installation dialog boxes are out of order
-51	The setup cannot create the specified folder
-52	The setup cannot access the specified file or folder
-53	An invalid option was selected

Table 4.1: Result codes and their definitions.

Benefits

As with the command-line installation, answer files offer the advantage of a vendor's support as well as the benefit of using the installation logic developed for this application. In addition, when tested successfully, answer files provide the ability to customize the installation to the degree provided by the vendor in the product's installation wizard.

Drawbacks

When an answer file installation does not work, the installation process simply aborts. This action could be considered positive during deployment, but during development, you are left with no clue as to what went wrong outside the previously discussed result codes. “Dialog Out Of Order” is not very helpful when you are unable to see which dialog box is causing the problem.

Many setup files spawn other setup processes to install prerequisites or required dependencies. Because it is the setup.exe that is making the call to these installations, you are unable to alter this behavior in order to silence those spawned installations. You can try installing them beforehand to see whether the setup program will then bypass the call, but I have found this to work in very few cases and might not be supported by the vendor. If you have a problem, asking the vendor for help is not likely to be an option unless they have documented the process or otherwise acknowledged it as a means of installation. Give it a try, but when no command-line or answer file support is available to you, repackaging is a typical next resort.

Repackaging

The process of repackaging an application is typically performed by comparing the file system and registry of a computer prior to and after installing a program. The delta is packaged into an MSI, EXE, or proprietary format, which, when applied to a similarly configured system, will effectively install the software.

> I’ve had several people inquire about statements that often exist in EULA documents that state “You may not distribute or repackage the software without written permission.” This statement does not refer to packaging and distribution in terms that a desktop administrator would be familiar with. This statement is referring to bundling the software with software that you plan to sell (or otherwise externally distribute) outside of your organization, including using components of their software within your own software or your installation media.

When no support for a silent installation is made available by the vendor, repackaging is a common means of generating one yourself. If the setup is not an MSI and no command-line support is available, repackaging can provide you with an MSI, executable (EXE), or proprietary formatted installation that can give you the silent installation you desire.

Benefits

When you repackage a setup, you get exactly what you want—a silent installation with all the customization and settings you decide upon. A great number of vendors do not realize or want to address the common need for a remote or automated deployment of their software. As a result, repackaging the vendor-provided setup is often a desirable means of creating such support on your own.

Drawbacks

If you have a problem with a repackaged installation, few vendors will offer support. In addition, it can often be unclear as to what may and may not belong inside your repackaged version of the setup.

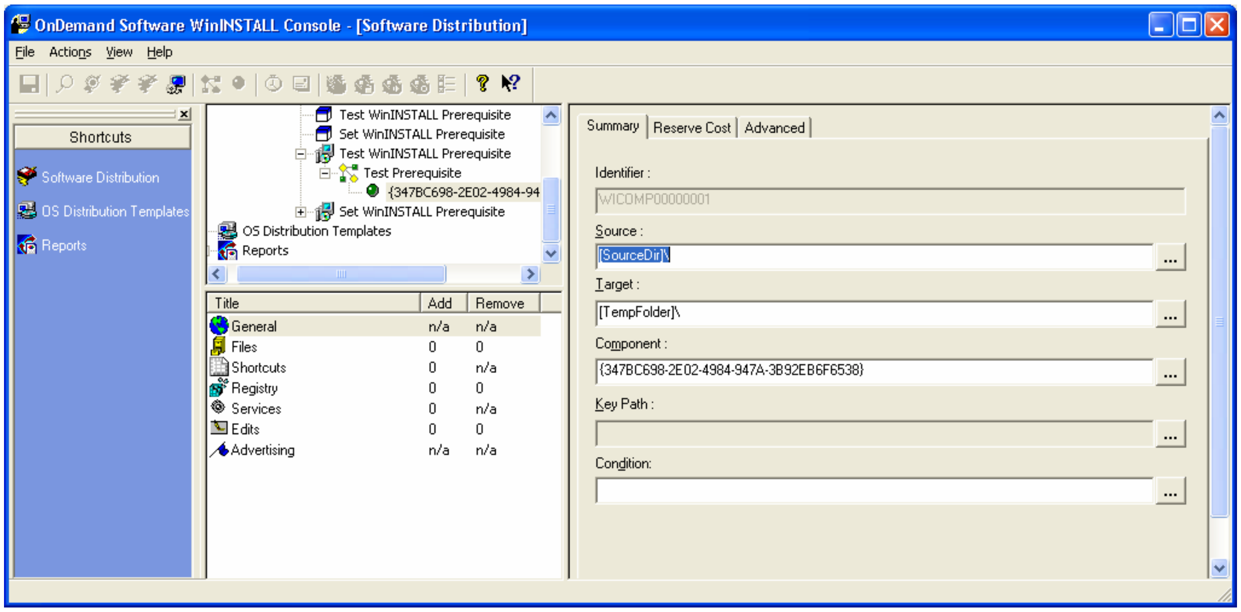
Package Development

With the installation media, your installation documentation, and your decided method of creating your software package, it is time to actually create your package. Your deployment system often dictates which methods you can use. Because a command-line installation (using switches or an answer file) requires very little work in this phase, we will focus on repackaging in this section (which requires the most work in this step of the process). For example, we will cover a couple of the most popular software packages for repackaging, focusing on those providing support for the Windows Installer (MSI) setup format desired by most administrators today.

> For a complete list of repackaging tools, for both MSI and legacy EXE packages, visit the AppDeploy.com tool area at http://appdeploy.com/tools/browse.asp?r=1.

OnDemand Software's WinINSTALL

OnDemand Software's WinINSTALL (see Figure 4.4) has a long history as an administrator tool for both the repackaging and deployment of software packages. Its repackaging capabilities were included in Win2K Server, and though Windows Server 2003 will not ship with a repackaging tool, OnDemand will be making an updated version of its repackaging software available for free in support of Windows Server 2003.


OnDemand Software WinINSTALL Console - [Software Distribution]
File
Actions
View
Help
Shortcuts
Software Distribution
OS Distribution Templates
Reports
Test WinINSTALL Prerequisite
Set WinINSTALL Prerequisite
Test WinINSTALL Prerequisite
Test Prerequisite
{347BC698-2E02-4984-94
Set WinINSTALL Prerequisite
OS Distribution Templates
Reports
Summary
Reserve Cost
Advanced

Title	Add	Remove
General	n/a	n/a
Files	0	0
Shortcuts	0	n/a
Registry	0	0
Services	0	n/a
Edits	0	0
Advertising	n/a	n/a

Identifier : WICOMP00000001
Source : [SourceDir]\
Target : [TempFolder]\
Component : {347BC698-2E02-4984-947A-3B92EB6F6538}
Key Path :
Condition:

Figure 4.4: WinINSTALL console.

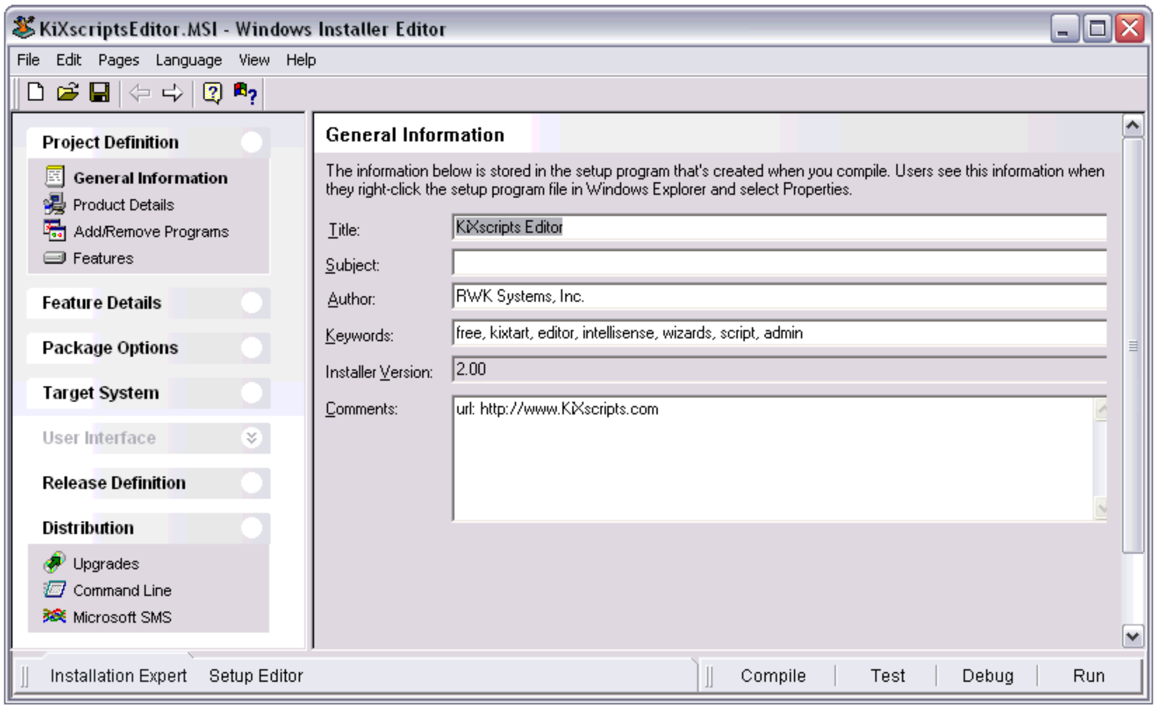
The following list offers some of the features that help WinINSTALL 7.5 stand out among similar tools:

- Provides delivery mechanisms for both push and pull distributions including calendar-based scheduling and deployments via an NT service or network logon script; it also provides an ability to perform an installation locally from a hidden partition
- Automatically synchronizes packages across distributed WinINSTALL servers

- Compares remote desktop and laptop application installations to the original image as a means of troubleshooting
- Automates notification of deployment actions via email, SNMP, or the Windows event logs
- Creates unattended installations of Windows OSs and lets you create and deploy them through the same GUI interface.

Wise Package Studio

Wise Package Studio also has a long history in the administrator community. The product's WiseScript repackaging capability has been around for several years and is familiar to many through Microsoft's licensing of an early release in the form of the SMS Installer. Wise expanded the interface and capabilities of WiseScript while providing a similar interface for the creation of MSI setup packages. Further developed into a full suite of tools for the creation, modification, and management of packages, Wise Package Studio (see Figure 4.5) also provides a powerful conflict checking and resolution tool that reads package components into a database so that they can be compared against selected packages to find potential conflicts.



General Information

The information below is stored in the setup program that's created when you compile. Users see this information when they right-click the setup program file in Windows Explorer and select Properties.

Title:	KiXscripts Editor
Subject:	
Author:	RWK Systems, Inc.
Keywords:	free, kixtart, editor, intellisense, wizards, script, admin
Installer Version:	2.00
Comments:	url: http://www.KiXscripts.com

Figure 4.5: Wise Package Studio Windows Installer Editor.

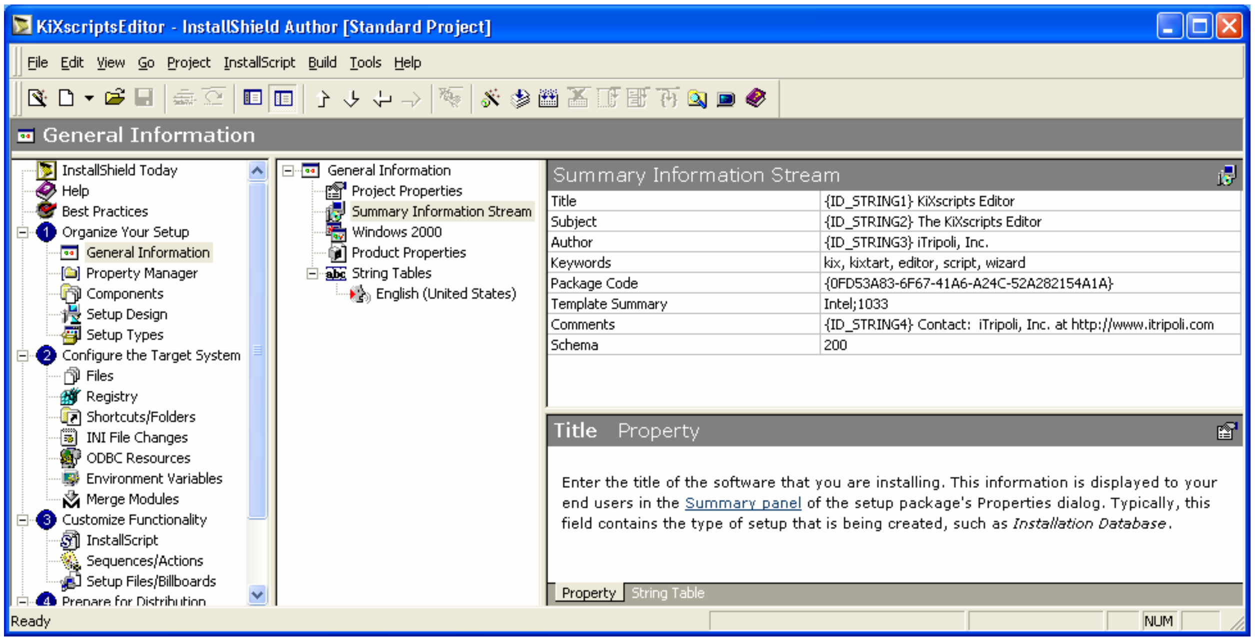
Features provided by Wise Package Studio 4.0 worth mentioning include:

- Pre-built, customizable, "point and click" processes built on best practices promote consistent, reproducible results and reduce errors
- Workbench interface provides a project-based framework that guides you through all phases of a package and release process
- Provides the ability to control access to repackaging projects with user security features

- Workflow features let you assign project tasks to individual team members or add items to a project's "To Do" list
- Application Gateway feature provides a Web-based portal that tracks applications from request to distribution, providing a way of requesting an application and managing the application preparation process
- Virtual Capture feature provides an ability to perform captures on a "non-clean" machine without re-imaging between captures

InstallShield AdminStudio

InstallShield AdminStudio provides much of the same functionality present in Wise Package Studio. InstallShield is known as a leader in setup software and have a long history in its offering of such tools for developers. Although a repackaging tool has been available with the InstallShield Developer (see Figure 4.6) tool (and earlier releases under different names), it has not been until the release of AdminStudio that a directed focus on the administrator community has been prevalent. The company has put significant resources into this tool and has produced a very powerful feature set, offering several new tools and features with each release. In addition to its other functionality, InstallShield AdminStudio provides repackaging, validation, and conflict resolution features.



Title	{ID_STRING1} KiXscripts Editor
Subject	{ID_STRING2} The KiXscripts Editor
Author	{ID_STRING3} iTripoli, Inc.
Keywords	kix, kixtart, editor, script, wizard
Package Code	{0FD53A83-6F67-41A6-A24C-52A282154A1A}
Template Summary	Intel;1033
Comments	{ID_STRING4} Contact: iTripoli, Inc. at http://www.itripoli.com
Schema	200

Title Property

Enter the title of the software that you are installing. This information is displayed to your end users in the Summary panel of the setup package's Properties dialog. Typically, this field contains the type of setup that is being created, such as *Installation Database*.

Figure 4.6: InstallShield AdminStudio's package editor, InstallShield Developer 8.

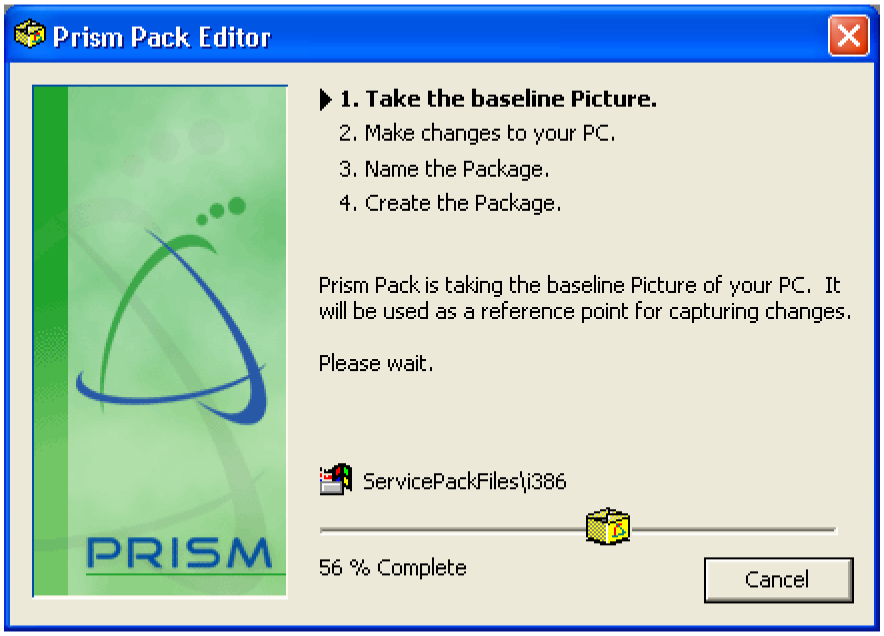
The following list offers some of the features provided by AdminStudio 5.0:

- Allows you to catalog all of your application deployment packages into a single enterprise conflict solver so that you can check your applications for a vast range of potential conflicts before you deploy them into your production environment
- Includes the full capabilities of InstallShield Developer—AdminStudio includes Developer with a selectable interface specifically tailored to systems administrators

- A process-oriented interface allows your team to step through projects, visually tracking what's been done and which tasks remain
- Enables distribution of Windows Installer packages to network and FTP locations, as administrative installations, or using Marimba channel publish technology
- Features its proprietary InstallMonitor technology, which aims to eliminate the need for snapshots by monitoring and recording all changes made during the installation

New Boundary Technologies' Prism Pack

Previously known as Picture Taker, New Boundary Technologies' Prism Pack is a packaging tool geared toward administrators that has also implemented recent updates to include support for the MSI format. Expanding upon this new functionality, a proprietary transform (MST) capability is available as well.



1. Take the baseline Picture.
2. Make changes to your PC.
3. Name the Package.
4. Create the Package.

Prism Pack is taking the baseline Picture of your PC. It will be used as a reference point for capturing changes.

Please wait.

ServicePackFiles\i386

56 % Complete

Figure 4.7: Prism Pack snapshot process view.

In addition to the functionality listed earlier, the following features help Prism Pack 5.0 stand out among similar tools:

- Interface is designed to simplify package customization for deployment of a single package for a diverse environment
- Inclusion of InstallShield Developer and Tuner products offer you full editing, customization, and MST support for Windows Installer files
- Smart Variables feature lets you turn a hard-coded value in a package into a variable that resolves itself as you tell it—prompt a user or pull a unique setting from each user's environment or from another variable

- Prism Conflict Checker helps to prevent errors before they occur by evaluating multiple deployment packages against one another for conflicts in the system and program files and the registry prior to deployment
- When it fixes software, it only reinstalls missing or broken files or components making the process faster and more bandwidth efficient than MSI's method of reinstalling at the feature level

Package Testing

It is obviously important to check your package before hitting users with it. Even the most experienced desktop administrators realize there are problems that occur, and it is best to take care of them before anyone has to know about it!

Package Quality Assurance

To ensure that package is ready for deployment, run your package from the command line. When it is complete, reboot if the installation will reboot during distribution but do not reboot if the distribution process will not. Now, log off and log on again as a user with standard access to the system (not an administrator account).

Walk through the procedures for testing the application you drafted when you had the user walking through the installation with you. If possible, have your test user return and validate your package by using the software and verifying any desired optional features are present.

For Windows Installer setups, a good test of your package is to prove that a user can successfully cause MSI to restore missing files and execute properly in the context of an account without administrative access to the computer. The following command line will let you advertise the setup to your workstation while logged on as a regular user:

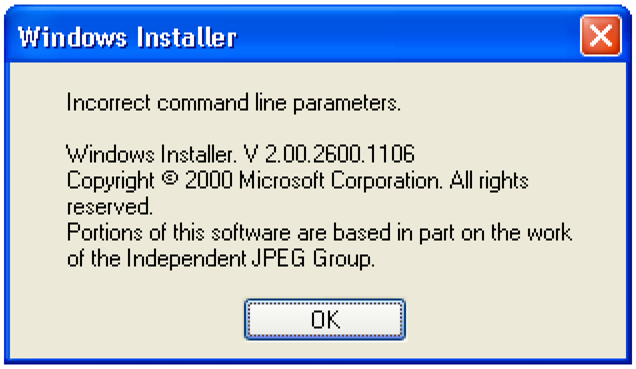
```
runas /user:seti-1\bkelly "msiexec /j z:\mysetup.msi /qb"
```

Advertising only installs any MSI shortcuts; the files will be seen as missing and installed on first use. Double-click the shortcut to trigger an installation and launch of the application. To account for spaces in a path or file name, you might need to precede the double quotes within the command line with backslashes (\) as in the following example:

```
runas /user:seti-1\bkelly "msiexec /j \"z:\my setup.msi\" /qb"
```

Deployment Testing

Just because you successfully run a package from the command line does not mean that you will have the same results when running it via your deployment system, as Figure 4.8 illustrates. File paths are often the cause for problems in deployment testing. If there are spaces in the path to the setup file (or any other file or path you must specify), make certain that you include quotes around them. When you add the possibility of switches and switches that include paths with quotes, your command line can get a bit ugly and require some playing around to figure out the correct placement.



Incorrect command line parameters.

Windows Installer. V 2.00.2600.1106
Copyright © 2000 Microsoft Corporation. All rights reserved.
Portions of this software are based in part on the work of the Independent JPEG Group.

Figure 4.8: When an invalid command line is passed to the Windows Installer, the system presents this dialog box.

Pilot Deployment

Before you send your package to hundreds or thousands of workstations, a period of real-world use is instrumental in a successful rollout. Many organizations make the mistake of allowing their network administrators to serve as pilot users. As they likely have administrative control of their systems, this test does not give you a good picture of how normal users will be affected—the main goal of a pilot deployment. Even if the administrators primarily use accounts with normal user access, the fact that they have administrative accounts might still hinder proper testing. For example, if an installation performs actions at next logon, they might be unsuccessful for a user with normal access. Even if administrators don't use their administrator accounts to test the package, the simple act of logging on as an administrator could hide a potentially big problem.

Volunteers Anyone?

If you ask for volunteers from different business functions within your organization, you will get better coverage of different user habits than if administrators perform this task. The idea of getting new software before everyone else is enticing to some users, who might happily volunteer to take part in your pilot software deployments. Ensure that the users taking part in the pilot testing understand the risks involved. Having users sign an agreement to acknowledge the understanding of these risks is an action taken by many organizations to alleviate potential problems down the road.

Collecting and acting on pilot results is crucial to the package testing process. Often a fixed group of users receive all packages and simply notify desktop administrators if a problem is encountered. Although better than nothing, relying on reports of problems alone is not an efficient or effective method of determining success. Ask specific questions of your pilot users and, to keep it from becoming routine, change your questions regularly.

Normally communicated via email, some kind of tracking method should be in place for future reference. Store responses in a separate mail folder, and for responses of significance, use the Save As option of your email software to store a text version of the email in the package folder structure (see Figure 4.1 earlier in this chapter).

A next step in using email tracking for pilot results is to make use of Outlook forms or similar functionality using other mail systems. More robust tracking systems include making use of databases and Web-based forms and reports. Whatever method you choose, keep retention of collected information in mind.

Full-Scale Deployment

Here we go—deployment time! Roll out the software in groups, over a couple of nights (or weeks if you have a very large network). Doing so will help minimize the impact of any problems you might have missed in your testing thus far.

User Notification

Some board or group of managers for the departments in your organization should be notified of your deployment schedule on a regular basis. Some organizations limit deployments to once per month or quarter or even bi-annually, but many are on an almost constant release schedule. Notify departments and users as appropriate to ensure that they have an opportunity to notify you of critical activities that might demand a modification of your deployment schedule. Some organizations post a Web page or spreadsheet of their deployment schedule for users to view as desired. Most organizations make use of internal email to send more focused announcements. Include in your communications to users information about what will change and which new features will be included in a deployment. If you're installing an update, explain what is new. If you're deploying a new application, provide information about why you're deploying it and how to use it. Make sure that users know what is coming—and providing them with a list of new features and benefits will even have them looking forward to the release.

Track Successes and Failures

Don't wait to count how many complaints there are to determine success. Most deployment suites provide some method of reporting to notify administrators of success or (if an error is encountered) failure. For failures, learn as much as you are able about what went wrong so that you can improve your process or take further parameters into consideration for your next deployment.

Management will want to know what the success rate was for the deployment. You can calculate this rate of success by dividing the number of successful installations by the number of target systems:

```
(Successful Installations / Total Target Systems) × 100 = Success Rate
      740 Successful installations
      932 Target systems
      79.3% Success Rate
```

To help with your success rate, establish what will be counted as a *deployable system*. If a machine is powered off or no longer exists, these machines should be included in the calculation you present to management:

```
[Successful Installations / (Total Target Systems - Powered Off or
Invalid Systems)] × 100 = Success Rate

      740 Successful installations

      932 Target systems

      128 Machines powered down

      4 Machines no longer exist (or are invalid entries)

      92.5% Success Rate
```

Baseline Updates

To maintain the benefits of a consistent baseline, as updates are introduced to your network, a critical task is to ensure that new and rebuilt systems are also up to date. You have a couple of options available to help ensure this capability.

Image Updates

Using your deployment packages to update your baseline image is one reliable method used for ensuring consistency between those systems updated with packages and those with a new image applied. Another common method is to make use of the installation documentation created while developing packages to help ensure that the images contain the same configuration and settings as those systems receiving distribution packages.

Unattended Installation Script Updates

If your installation process is script based, be sure to replace old installation scripts with those of your more recently deployed installation scripts. Because this task requires little effort compared with that required to update an image, you might choose to implement the new installation scripts at the time they are deployed. In this way, you will not have to be concerned with updating new machines via your software deployment system.

☞ INOSOFT Garibaldi is a tool to help manage images, installation source files, and scripts. It helps generate and track versions of installation scripts and facilitate adding your scripts to existing systems and to your baseline using an unattended installation process. For more information about Garibaldi, visit http://www.inosoft.de/garibaldi/Default.asp?Sp=en.

Package Availability

Don't keep your packages a secret. Particularly in large environments in which several offices might have a requirement for applications with similar functionality, seeing that a package has already been generated and deployed may be a determining factor in the software they select. Where there is a significant backlog of applications waiting to make their way through your package and deployment process, helping the user to select an application you have already deployed will save the user time waiting for package development and will reduce the number of applications you need to manage.

Package Deployment System

With your package in hand, how do you intend to get it out there? Chances are, unless you are establishing a new network or significantly increasing the size of a small network, you probably have an existing deployment system. Even if you do, this section will still be of interest to help you get an idea of functionality that you might not have and will want to request of your deployment system vendor.

It is beyond the scope of this book to provide sufficient detail about every deployment system available. So rather than list just a few of the most popular deployment systems and only a brief summary of the functionality they provide, Table 4.2 provides a list of many of the solutions available to help those who need one begin the search for the right tool.

What is the right tool? Every organization will have its own set of requirements. It is easy to get lost in the sea of features made available by these systems. Rather than focusing on any one feature, try instead to create a list of features in which you are interested. You can then be more methodical in your approach to finding the best tool for your organization. Besides, there is no sense in paying for features you do not need.

Software Distribution Vendors

A large number of companies have taken a place in the market to provide a tool to address your software deployment needs. Several companies means several different takes on the subject, with each providing its own perspective—resulting in a unique interface and feature set to address a wide range of issues common to many organizations. Table 4.2 describes a few of your options.

Company	URL	Description
Altiris	http://www.altiris.com	Altiris provides several desktop administration tools in its Client Management Suite including Deployment Solution, Application Management Solution, and Software Delivery Solution. Its Web-based interface is modular in nature, providing several capabilities that can be added to satisfy a wide range of issues. Also part of the Client Management Suite: Inventory Solution, Application Metering Solution, and Carbon Copy Solution (remote control).
ASDIS Software	http://www.asdis.com	ADSIS Software provides a solution of the same name, which serves to provide central management of application servers, automate installation of new systems, and manage changes to computers throughout a network. By their own metrics, ASDIS implemented more than 15 million software changes on 36,000 systems in the year 2000.
Baramundi Software	http://www.baramundi.de	Baramundi provides OS-Deploy, App-Deploy, and an enterprise management framework, as well as inventory and other add-ons. As the names imply, the system is capable of remotely deploying unattended installation scripts for Windows OSs as well as automated software deployments.

Boss	http://www.boss-uwin.com	Boss offers U-Win, a complete system for workstation inventory, Windows OS migration, application deployment, and the migration of user data and settings.
Broccoli Software	http://www.broccolisoftware.com	ADS is a management tool for deployment of Windows software applications. ADS restricts the availability of software based on user groups as well as informs users of new and updated applications and patches as they become available. In addition to deploying software, you can also use it to deploy security updates, service packs, and drivers.
CenterRun	http://www.centerrun.com	CenterRun executes all application changes from a central console, simulating changes prior to making them, and maintaining a complete audit history. CenterRun tells you who installs what where. It also compares the actual state of an application with what it should be as recorded in a central repository. This feature captures out-of-band changes to the application environment.
Cognet	http://www.cognet.com	PowerQuest recently purchased Cognet and its Cognet Manager product. Cognet Manager provides a software deployment and configuration solution designed for ease of use. It provides "Plain English Policies" that can affect application configuration to provide each user with a tailored installation.
Computer Associates	http://www.ca.com	Computer Associates' Unicenter Software Delivery product provides automated software delivery throughout heterogeneous environments with rollback and version control capabilities. It is also designed to support remote users, PDAs, and mobile devices.
Contrado	http://www.contrado.com	Contrado offers SystemBuilder, a solution for the creation, deployment, or migration of Windows systems. It aims to provide a single-step virtual process, eliminating the need for remote staging, hands on touches, and multiple deployment tools.
Executive Software	http://www.execsoft.com	Executive Software provides Sitekeeper as its tool to handle software installation, license tracking, and software and hardware inventory. Supporting all releases of Windows since Windows 95b, Sitekeeper is focused on making system management simple, with a very small learning curve and no training required.
Hewlett-Packard	http://h18013.www1.hp.com/products/servers/management/rdu/description.html	Geared toward server management, the Remote Deployment Utility (RDU) is an application that remotely deploys driver and management agent updates to network attached servers. You can operate RDU from an IT administrator's workstation, where remote servers are selected for software updates.

IBM	http://www.unitysite.com	IBM's Unity provides a deployment solution for Win2K and Windows XP systems. It can roll out new OS deployments and allows for the creation of MSI deployment packages. It features a replication capability that allows for hierarchical distribution of software packages over a WAN and will automatically discover the fastest link to the package servers.
InfraDirect	http://www.infradirect.com	InfraDirect provides its solution, Xpursuit, to provide application installation and licensing from a single central management console. It features application conflict prevention and a lab environment in which applications may be analyzed before distribution. Client/server communications are MD5 encrypted and authenticated through LDAP standard directory structures for increased security.
Insystek	http://www.insystek.com	Insystek provides a couple of products for desktop administration including NSCM Control Center. This tool lets you inventory, manage services, install, upgrade, or remove applications all from a single console interface. Insystek also provides a similar product (built on the same technology) named SoftDist LE, which is designed for small to midsized Windows environments.
LAN Supervision	http://www.lsvi.com	LAN Supervision (LSVi) provides Change Management Facility (CMF), which offers software and configuration management features. It features self-repair, task scheduling, version control, and built-in rollback and workstation configuration capabilities. It requires no scripting and boasts scalability up to 40,000 clients. LSVi also offers OnDemand Software, a self-service deployment solution.
LANDesk Software	http://www.landesksoftware.com	LANDesk software offers its LANDesk Management Suite with OS deployment. It allows you to build and deploy images with its OS Deployment Wizard and is compatible with leading third-party imaging solutions as well. It also allows for targeted multicasting of software packages with scheduling capabilities. LANDesk also feature a Pre-execution Environment (PXE) server solution that does not require a PXE server for each subnet.
ManageSoft	http://www.managesoft.com	ManageSoft provides a software management solution for desktops, servers, and mobile devices. It features software deployment, asset tracking, license management, Windows migration, and mobile/remote management capabilities.

Marimba	http://www.marimba.com	Marimba provides separate management solutions for desktop and mobile management, server management, and embedded management. The desktop/mobile management solution offers software distribution, inventory management, subscription management, and modules for migration and infrastructure.
Microsoft	http://www.microsoft.com/management	Microsoft provides several solutions for management of Windows systems including Systems Management Server (SMS), IntelliMirror (Group Policy), Remote Installation Services (RIS), and Software Update Services (SUS). Microsoft SMS and IntelliMirror make up a large percentage of the management solutions in use today. Both are powerful and regularly enhanced. As a result of the popularity of these solutions, many companies feature interoperability through sharing of data or by developing add-in products to enhance the features provided out of the box.
Mobile Automation	http://www.mobileautomation.com	Mobile Automation's Mobile Lifecycle Management Suite aims to provide an all-in-one solution for discovering, securing, managing, maintaining, supporting, and migrating all desktops, laptops, servers, and handheld devices from any location via a single management console.
NetSupport Solutions	http://www.netsupport-solutions.com	NetSupport Solutions offers NetInstall, a software packaging, distribution, and installation system that also provides creation, distribution, and execution of configuration scripts. There are two editions of NetInstall, a Standard Edition (SE) and an Enterprise Edition (EE), which includes additional features for use in large and distributed network environments.
New Boundary Technologies	http://www.newboundary.com	New Boundary Technologies provides its Prism Deploy software for the installation and updating of software on local and mobile clients and servers. It provides automatic targeting of computers based on hardware, software, or network configuration without a prior inventory scan.
Novadigm	http://www.novadigm.com	Radia Software Manager is Novadigm's change and configuration management tool that allows policy-based control of deployments, updates, repairs, and removals. Other tools in the Radia family include its Software Manager (Web-based user installations), Inventory Manager (hardware and software), OS Manager (imaging and settings migration), and Desktop Manager (change and configuration management).

Novell	http://www.novell.com/	Novell offers ZENworks for Desktops, which runs from Novell or Windows servers and requires no desktop client for management of Windows 98 and later systems. It provides application deployment, policy management, software and hardware inventory, desktop imaging and remote management.
On Technology	http://www.on.com	ON Command SiteManager is provided by On Technology as a desktop management system optimized for smaller organizations. It provides a single console interface for deploying OSs, installing applications, updating software, and performing disaster recovery and Windows migrations. For larger networks, ON Command CCM offers an open and scalable system providing much the same functionality.
OnDemand Software	http://www.ondemandsoftware.com	Often thought of as only a repackaging tool, OnDemand Software's WinINSTALL features automated software distribution to desktop and mobile clients. It creates packages and allows for deployment of Windows OSs and applications.
Outerbounds Technologies	http://www.outerbounds.com	Outerbounds Technologies offers PC lifecycle management through its Empirum software. It features automated Windows migrations (scripted, not imaging), hardware and software inventory, disaster recovery and backup, and electronic software distribution.
PatchLink	http://www.patchlink.com	PatchLink provides PatchLink Update, a cross-platform patch discovery and distribution utility that provides patch, software, data, and task deployment across the Internet. It automatically detects patch-related security vulnerabilities on client systems and provides the ability to correct them across all platforms and enterprise boundaries.
Quest Software	http://www.quest.com	DeployDirector from Quest Software allows you to deploy, rollback, update, and manage your client-side Java applications, accelerating release cycles and reducing time-to-repair.
ScriptLogic	http://www.scriptlogic.com	A graphical network administration tool, ScriptLogic includes features to support enterprise software deployment. Although it does not include a packaging component, the tool can be used to facilitate the deployment of existing packages (for example, MSI files) such as those containing applications, service packs, and so on.
Snow Software	http://www.snow.no	Snow Software offers its Snow Distribution client management tool for the automatic distribution of applications, updates, and OS changes. It provides centralized Start menu and shortcut management as well as its own scripting language.

St. Bernard Software	http://www.stbernard.com	Focusing on the tracking and deployment of service packs, hotfixes, and other patches, St. Bernard Software's UpdateEXPERT assesses client systems for missing patches based on the company's database of available fixes. You can research available fixes, scan your systems for them, then deploy updates to those machines you wish to update.
Vector Networks	http://www.vector-networks.com	Vector Networks provides centralized desktop management via its PC-Duo Enterprise suite. Its modular architecture gives you the ability to purchase just the modules you require, including Inventory Management, Software Distribution, Software Metering, Help Desk Issue Tracking, Diagnostics and Remote Control.
XcelleNet	http://www.xcellenet.com	XcelleNet Afaria focuses on systems management for mobile devices. It provides the capability to deploy applications and content, automatically back up data, and track hardware and software information.
Xyro B4 Software	http://www.xyro.com	Xyro B4 provides its Active Distribution Console (ADC) for the one-to-many distribution of software or even data files via IP multicasts. It is also able to execute remote commands on several machines simultaneously allowing for its software installation capabilities.

Table 5.2: Software deployment solution vendors.

✎ These and many more products are detailed on AppDeploy.com at http://www.appdeploy.com/tools.

Summary

Each environment must have its own deployment process, and as the desktop administrator, you'll most likely be tasked with the creation of this process. In this chapter, we discussed how determine with application deployment method will work best for your organization as well as explored some of the tools available to help you do so.

In the next chapter, we will turn our focus to security, which has become a hot topic in the past few years. We will discuss security policy, tools, things to watch out for, and the benefits and drawbacks of implementing a locked down environment.