Deploying Windows 7 from A to Z

Microsoft Corporation

Published: January 2010

Abstract

In this article, Jeremy Chapman, a senior product manager at Microsoft, documents the high-level steps for IT professionals to perform an enterprise-scale desktop deployment project—starting with Windows XP and moving to Windows 7.

Microsoft

Copyright information

Contents

Deploying Windows 7 from A to Z 5

Migrating User Files and Settings from Windows XP to Windows 7 7

Application Management and Preparing for a Windows 7 Deployment 9
- Application Compatibility 10
- Application Packaging 13

Choosing an Image Strategy and Building Windows 7 System Images 14
- Quick History Lesson for System Imaging 14
- Building Your Image 16
- Getting to Thin Images 17

Automating the Migration from Windows XP to Windows 7 End-to-End 18
- Automating the End-to-End Migration Process 19
- Tricks for More Automation 23

Deploying Windows 7 from A to Z

By Jeremy Chapman, senior product manager, Microsoft® Corporation

Notes

For a complete view of Windows 7 resources, articles, demos, and guidance, please visit the Springboard Series for Windows 7 on the Windows Client TechCenter.

For a Web version of this document, see the Deploying Windows 7 from A to Z in the Windows 7 Technical Library (http://go.microsoft.com/fwlink/?LinkId=180726).

As the enterprise operating system deployment guy, I often get requests for a "one-page" article about performing enterprise-scale desktop deployment or how to migrate hundreds or thousands of computers running Windows® XP to Windows 7. Although I have seen full-length books printed on the head of a pen, I don't think it is possible to explain the entire set of desktop migration tasks when moving from one operating system version to another within one page. If you are upgrading one computer from Windows Vista®, or performing a clean installation on your personal computer (coming from any recent version of Windows), there is one-page guidance available on the Microsoft Web site: Installing and reinstalling Windows 7. But it probably won't satisfy you if you want to perform these tasks more than a few times.

Let's start by stating a few assumptions:

1. You are an IT professional looking to move multiple computers or users from Windows XP or Windows Vista to Windows 7.
2. The computers you are transitioning to Windows 7 have user data, settings, and applications that somehow (either partially or completely) need to be migrated to Windows 7.
3. You don't want to manually transfer user files through file copy or use manually-operated consumer tools (such as Windows Easy Transfer) from the legacy computer to Windows 7.
4. You would prefer to have applications be part of the customized operating system you install or automate application installation as part of the total deployment process.
5. In the best case, you would prefer that the entire process is as automated as possible.
6. You have some previous experience with operating system installation, deployment, or system imaging.

You might be thinking, “What about using the common processes of hard-drive cloning or sector-based imaging to duplicate a reference installation?”

The good news is that with advances in system imaging, you no longer need to spend hours saving user data off an old computer, cloning the hard drive of a reference computer, and then taking the time to restore the data you saved in the first step. Although the hard drive cloning process is probably the most common practice used, it has several disadvantages, including:

- The time required to clone each system.
- A proliferation of hard-drive images based on various hardware types.
- Large image sizes and the corresponding storage space consumed.

- The potential cost of hard-drive cloning software.
- The excess time spent rebuilding and maintaining multiple images.
- Manual processes for activating end user licenses.

You have a few options when migrating from Windows XP, and many of them depend on the size and complexity of your environment. We highlight four primary options for migrating from Windows XP to Windows 7 (or Windows Vista) in the Choosing a Deployment Strategy article on Microsoft TechNet. In fact, that article goes into greater detail than this document, and it does a great job of pointing to tools and resources.

I'm using quite a few terms interchangeably throughout this document. When I use the terms like "migrate from Windows XP" or "operating system deployment" or "transition from Windows XP", I am talking about the following major steps we cover in any operating system deployment:

- Collecting existing user data and settings (if they exist)
- Installing the operating system
- Installing drivers and applications
- Activating the operating system
- Joining a domain (if necessary)
- Restoring user data and settings
- Providing the flexibility to customize which applications we install by user role, and applying language preference, locale, time zone, and so on based on user needs

I usually refer to this collective process across multiple computers as "deployment," and there are a few other terms we'll define before concluding this introduction. Subsequent sections will refer to installation scenarios, so let's define the main ones:

1. Refresh Computer. This is when a user has a computer with files, settings, and applications. We will be installing the new operating system to that existing computer and assuming the same user keeps that computer. In this case, we try to keep user files and settings locally on that computer to save time, storage, and network bandwidth. Some refer to this as an "in-place wipe and load" (without actually wiping the user's data) or loosely as "upgrading" a computer.
2. Replace Computer. This is when the user is getting a new computer or a computer is reassigned from another user and the user data and settings need to move off the old computer through some method and onto the new computer. This scenario tends to take the most time compared to Refresh Computer and New Computer. Some refer to this as "side-by-side" migration, but it isn't necessary for the computers to be physically near each other or otherwise connected in this scenario.
3. New Computer. This is when there is no requirement to migrate pre-existing user data or settings. New Computer is used for a new hire or for a secondary computer, or if an old computer was lost or damaged and user data was not previously backed up. Some refer to this as "bare metal" deployment, but in most cases there is some OEM preinstalled operating system we will be replacing.

Now we have listed the assumptions for this document, listed a few reasons why you may want to look at your existing deployment process if it involves hard-drive cloning, roughly defined the all-up operating system deployment process, and defined the primary installation scenarios. So far, I've gone over a page in length, but this provides the backdrop for the upcoming sections. In the next section, I'll describe the options and recommendations for user data and settings migration when moving from Windows XP to Windows 7.

Migrating User Files and Settings from Windows XP to Windows 7

I'll start with data migration purely based on the automation steps in the chain I listed in the first section and because there aren't many things that excite me more in the Windows 7 deployment space than hard-link migration. Think about the speeds of large file moves versus file copies; that is essentially the speed advantage that hard-link migration can have. If you're inpatient like I am, and you hate waiting 3-4 hours per computer for the user state to migrate from one operating system to another (especially considering the operating system plus applications combined often takes under 45 minutes), then you might appreciate hard-link migration like I do.

To be fair, you could use technologies and tactics like roaming profiles and folder redirection, and you could disallow the creation of local e-mail stores, at which point many of the problems associated with user state migration might go away. But in this case, I'll cover the traditional case of user state as part of the user's computer.

If you manage desktop computers, you probably have multiple users with files in every possible location on their computers, and there are settings like Internet Explorer favorites, known wireless network connections, and application settings that you want back in the new operating system. We can handle all of these things except for migrating the applications themselves. In most cases, we'll want to test the compatibility of older applications, and not necessarily attempt to migrate them in-place from Windows XP to Windows 7. Plus, we can completely automate the installation of managed applications at deployment time or capture them as part of the custom operating system image. I'll cover all of this in the next section when I talk about application management.

Bringing this back to user files and settings, we know that user data is typically stored in a couple of places:

- "C:\Documents and Settings" in Windows XP
- "C:\Users" in Windows Vista and newer versions of Windows

Then the question that begs to be asked is...

“Can't we use a utility such as Robocopy to move the files from "C:\Documents and Settings" to a folder somewhere on the network that is named for the computer and then copy them back to the "C:\Users" folder after we're finished?”

Well, yes and no; it isn't usually that simple though. Unless you have draconian standards and policies about where your users can save data on their local drives, this won't suffice. We also

have to look through settings in the registry that we want to retain on the new computer, create and enable user accounts, and some of us might want to block some files from getting migrated. For example, you may not want Joe in the marketing department to store his 100 GB music and video collection on your managed computer, so we may want to block certain file types in the migration process (hopefully we let Joe know about this in advance so he has time to back up his media collection). On the other hand, Joe may not know where his Microsoft Office Outlook® PST files are, and even if we catch that PST file with our Robocopy command, he'll probably call the Help Desk asking where his cached e-mail is after we've installed Windows 7.

The good news is that we have a tool for this, the User State Migration Tool (USMT). The even better news is that if you use the Microsoft Deployment Toolkit (MDT) or System Center Configuration Manager 2007 SP2, it's already part of the end-to-end operating system deployment process. You may have seen the migration demos from Windows XP to Windows 7 occur in as few as 18 minutes (yes, with several gigabytes of data being migrated, too), but if you haven't, check them out:

- As part of the MDT: Windows 7 Deployment Tools Part 4 Demo
- As part of Configuration Manager: Microsoft Management Summit 2009 Keynote Demo

Both of these demos use the Hard-Link Migration feature for the Computer Refresh scenario (I defined this and other scenarios in the first section). In the old days, USMT could support a Computer Refresh without moving the files off the computer, but it was basically a file copy and double-instancing of files locally. Now the files do not move on the disk when migrating from Windows XP—we simply reassign pointers to files to the appropriate locations in Windows 7. The index of hard links consumes around 200 MB, so you don't need a lot of free disk space to use Hard-Link Migration. Think about how much faster it is to "move" a 1 GB file versus copy it to another location on the same disk; that is why Hard-Link Migration is so much faster. It doesn't really matter if we move 5 GB or 50 GB in the migration; the times will be pretty similar. The amount of time depends on the number of files we move and not the size of the files.

The User State Migration Tool is now part of the Windows Automated Installation Kit (AIK). The USMT installs simply through a file copy. After you install the Windows AIK, by default, the USMT tools for 32-bit and 64-bit operating systems are located in the "C:\Program Files\Windows AIK\Tools\USMT" folder.

You can download the Windows AIK from the following Microsoft Web site: Windows Automated Installation Kit for Windows 7.

You can perform a simple Computer Refresh with hard-link migration by using normal Windows 7 installation media along with USMT files on a USB drive I outline this process in a couple of locations:

- In video form: Migrating from Windows XP to Windows 7
- In written form on TechNet: Windows XP to Windows 7 Hard-Link Migration of User Files and Settings

This process will quickly migrate files from the default-created Windows.old folder when you install Windows 7 onto a computer running Windows XP. If you don't follow the default process and format the Windows partition during the installation process, Windows.old won't be there to

migrate from. So remember to follow the default installation process to keep your data and create Windows.old.

You might be asking another question at this point...

"What if I am performing a Computer Replacement and the data needs to move from my users' old computer that is running Windows XP to my new computer that is running Windows 7?"

Computer Replacement is pretty common, and the tools are built to handle this. Normally, the data is passed from the old computer to a network share and then from the network share to the new computer. Both Configuration Manager and MDT can be used to automate the entire computer replacement migration process using a network share. You can even encrypt the migration store on the network (as Configuration Manager does by default) to ensure data stores cannot be compromised.

Another useful addition to the migration tools for Computer Replacement is support for Volume Shadow Copy. That means that we can gather files even while they are in use. One thing to point out with Computer Replacement is that you can't get the speed benefits from Hard-Link Migration, because we are at the mercy of physics moving the data to an external network location or external hard drive.

What about the types and locations of files that get migrated? The User State Migration Tool adds a new algorithm to find more user files than previous releases versions of USMT. The control file (migdocs.xml) uses shared and comprehensive file detection logic with Windows Easy Transfer. So if you used USMT in the past, and you had to extensively modify the previous control files (for example miguser.xml) to add file coverage, the new migdocs.xml should be a welcome addition.

There are other methods to move files between operating systems, but USMT was developed specifically to manage the migration of user data and to support as many mainstream options as possible for customers with a large number of computers to manage. USMT does a great job migrating user files and settings. For people who are looking for a user interface for USMT, I would recommend that you use it in conjunction with the Microsoft Deployment Toolkit or with System Center Configuration Manager. If you have used USMT in the past without a lot of success, the current version of USMT offers several enhancements to increase speed, flexibility, and process reliability to make the migration portion of your operating system deployment easier and more predictable.

In the next section, I'll take on the topic of application management, including application compatibility and automating application installation, as well as touch on hardware inventory and compatibility.

Application Management and Preparing for a Windows 7 Deployment

If you are like most desktop service managers, you probably have several applications that you manage; and depending on your users, there may be several applications that you don’t know about. There are a few places where “Standard User” or “Least-Privileged User” accounts aren’t

the norm, and users can install whatever applications they want. If this sounds familiar, then you probably have a bit of work to do in terms of detecting, rationalizing, and testing the applications that your users have to prepare for Windows 7. You can address compatibility issues by using a variety of approaches, and we'll discuss all of these approaches in a minute.

The next major area you'll need to worry about is how to get those applications or corresponding newer versions onto your users' desktops that are running Windows 7. There is a fair chance that you've been cloning hard drives over the past couple of decades and don't necessarily have automated installation packages for all the applications that you want to deploy into the new Windows 7 environment. Some applications aren't packaged for silent installation, making it hard to customize installations to include only applications that are specific to user roles. Along with other factors, this means there are a lot of organizations that have 20 to 50 applications present on every workstation—even if the users only need 5-10 of those applications on average. With the newer deployment tools, you can save your money on these applications and install what is required per user instead of giving them everything. This will save you on licensing and may improve the performance of Windows if some of those applications launch at startup. I'll talk about the application packaging process after we dig a bit deeper into application compatibility.

Application Compatibility

Contrary to popular belief, the application compatibility process doesn't begin with testing. The first step you'll probably want to take is to collect an inventory of your hardware and software assets. Be prepared to discover more applications than you thought you had. One tool you can use is the Application Compatibility Toolkit or "ACT."

You can download the ACT from the Microsoft Web site: Application Compatibility Toolkit.

The Application Compatibility Toolkit isn't a magic wand that you wave at applications to "make them work," but it does provide the tools to inventory your applications, hardware, and devices; evaluate runtime compatibility of applications while collecting data; and compare what you've collected to a central database managed by Microsoft with compatibility data from ISVs and the IT pro community.

When I present the toolkit at events, I often get asked, "Hold on, did you just say the Application Compatibility Toolkit discovers hardware and devices, and not only applications?"

Yes, it's true. ACT finds the applications wherever they may reside and it also reports hardware and devices that are detected. One thing to point out is that while most inventory tools relegate themselves to only data found in Add/Remove Programs, ACT also looks in multiple locations of the registry (Run, Run once, file name extension handlers, application paths, and so on), and in services. ACT tends to find any application on the system or otherwise launched by the user while we're performing an inventory. After you deploy the lightweight Data Collection Package agent in ACT, ACT detects the information on your users' Windows XP workstations and sends the information back the network location you specify, then processes the data and reports its findings to you.

Check out a video on TechNet about Data Collection Packages: Creating Data Collection Packages to Generate an Application Inventory

The following graphic shows a sample application inventory in the Application Compatibility Manager:

Application Name	Version	Company	Vendor Assessment	Community Assessment	Computers
Microsoft Digital Image Suite Anniversary...	11.1.2018.0	Microsoft Corporati...	W		1
Microsoft Document Explorer 2008	9.0	Microsoft Corporati...		1 0 0	2
Microsoft Dynamics AX 4.0 Client	4.0.2501.1...	Microsoft Corporati...	W		1
Microsoft Firewall Client	4.0.3441.6...	Microsoft (R) Corp...	W		1
Microsoft Firewall Client	4.0.3442.6...	Microsoft (R) Corp...	W	7 0 0	3
Microsoft Forefront Client Security Antimal...	1.5.1951	Microsoft Corporati...			2
Microsoft Forefront Client Security State A...	1.0.1697	Microsoft Corporati...		2 0 0	2
Microsoft Malware Protection	1.5.1954.0	Microsoft Corporati...			2
Microsoft MSDN 2005 Express Edition - E...	8.0	Microsoft Corporati...			1
Microsoft Office 2003 Web Components	11.0.6533	Microsoft Corporati...		14 0 1	1
Microsoft Office Access 2007	12.0.4518.0	Microsoft Corporati...	C	44 0 0	4
Microsoft Office Communicator 2005	1.0.557	Microsoft Corporati...		23 2 0	2
Microsoft Office Communicator 2007	2.0.6338	Microsoft Corporati...			1
Microsoft Office Communicator 2007 R2	3.5.6881	Microsoft Corporati...		1 0 0	1
Microsoft Office Communicator 2007, MUI	2.0.6362.0	Microsoft Corporati...		1 0 0	2
Microsoft Office Enterprise 2007	12.0.6418...	Microsoft Corporati...	C		1
Microsoft Office Enterprise 2007	12.0.6215...	Microsoft Corporati...	C	2 0 0	1
Microsoft Office Enterprise 2007	12.0.6215...	Microsoft Corporati...	C	1 0 0	2
Microsoft Office Live Meeting 2005	7.3.2015	Microsoft Corporati...		8 0 0	2
Microsoft Office Live Meeting 2007	8.0.6338	Microsoft Corporati...		9 0 0	1
Microsoft Office Live Meeting Add-in Pack	7.0.2010.5	Microsoft Corporati...		2 0 0	2
Microsoft Office Visio 2007				11 0 0	1
Microsoft Office Visio 2007 Service Pack...		Microsoft		2 0 0	1
Microsoft Operations Manager 2005 Agent	5.0.2900	Microsoft Corporati...		6 0 0	2
Microsoft Product Studio	2.10.6703	Microsoft Corporati...		4 1 1	2
Microsoft Save as PDF or XPS Add-in for...	12.0.4518...	Microsoft Corporati...		17 0 0	4
Microsoft Silverlight	2.0.39959	Microsoft Corporati...		4 0 0	1
Microsoft Silverlight	2.0.30884	Microsoft Corporati...			4
Microsoft SQL Server 2005 Backward co...	8.5.1698	Microsoft Corporati...		5 0 0	1

Many IT shops have an inventory that they are confident in and they don't necessarily want to deploy agents to their users—it's completely understandable. In this case, is there anything better than using the consumer compatibility site to search applications and devices one-by-one?

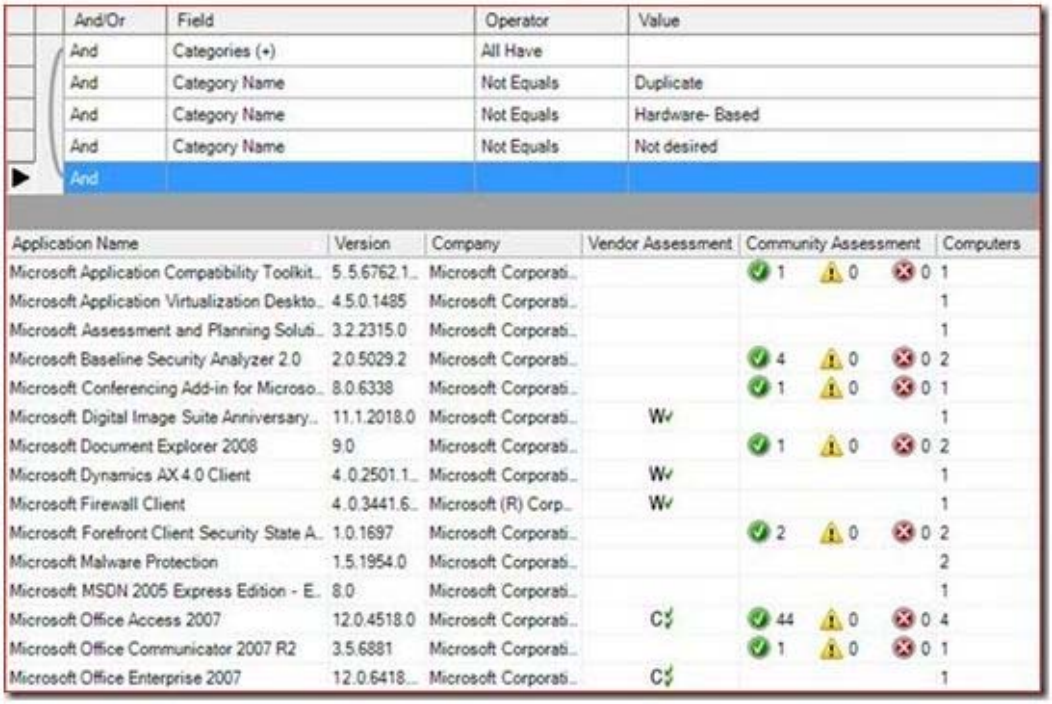
Consumer compatibility site

If you have more than about 20 applications, using the consumer compatibility site probably isn't your best option. We also publish the following list of known compatible applications that you can use to query your application inventory database:

Windows 7 Application Compatibility List for IT Professionals

What's next? If you have a thorough inventory, it is extremely likely that you won't want to move all of the applications you find from Windows XP into your new Windows 7 environment. There might be five different media players or eight different applications that read PDF files on your users' collective computers. In fact, many companies can eliminate 90% or more of the applications they inventory because they are duplicate in nature, hardware-based, or undesired. You can use the filtering provided in ACT to help reduce the list.

The following graphic shows a categorization of applications by using the Toggle Filter in Application Compatibility Manager:



And/Or	Field	Operator	Value
And	Categories (+)	All Have	
And	Category Name	Not Equals	Duplicate
And	Category Name	Not Equals	Hardware- Based
And	Category Name	Not Equals	Not desired
And			

Application Name	Version	Company	Vendor Assessment	Community Assessment	Computers
Microsoft Application Compatibility Toolkit...	5.5.6762.1...	Microsoft Corporati...		1 0 0	1
Microsoft Application Virtualization Deskto...	4.5.0.1485	Microsoft Corporati...			1
Microsoft Assessment and Planning Soluti...	3.2.2315.0	Microsoft Corporati...			1
Microsoft Baseline Security Analyzer 2.0	2.0.5029.2	Microsoft Corporati...		4 0 0	2
Microsoft Conferencing Add-in for Microso...	8.0.6338	Microsoft Corporati...		1 0 0	1
Microsoft Digital Image Suite Anniversary...	11.1.2018.0	Microsoft Corporati...	W		1
Microsoft Document Explorer 2008	9.0	Microsoft Corporati...		1 0 0	2
Microsoft Dynamics AX 4.0 Client	4.0.2501.1...	Microsoft Corporati...	W		1
Microsoft Firewall Client	4.0.3441.6...	Microsoft (R) Corp...	W		1
Microsoft Forefront Client Security State A...	1.0.1697	Microsoft Corporati...		2 0 0	2
Microsoft Malware Protection	1.5.1954.0	Microsoft Corporati...			2
Microsoft MSDN 2005 Express Edition - E...	8.0	Microsoft Corporati...			1
Microsoft Office Access 2007	12.0.4518.0	Microsoft Corporati...	C	44 0 0	4
Microsoft Office Communicator 2007 R2	3.5.6881	Microsoft Corporati...		1 0 0	1
Microsoft Office Enterprise 2007	12.0.6418...	Microsoft Corporati...	C		1

This is important, because it is much easier to test 100 applications compared to 1000 applications and you can often reduce your inventory list in a couple of hours. I'd personally rather test 100 applications than 1000, and I'm guessing that most would agree.

Watch a video on TechNet about working with ACT inventories: [Analyzing Compatibility Data Returned by Data Collection Packages]

Now that we have the rationalized list of applications and static data from ISVs as to whether they are compatible, the fun starts with testing these applications without ISV information. You should find that most of your applications work in Windows 7 and this is especially true for packaged (ISV) applications that were released in the last three years. The in-house developed applications that you've had for five or more years may require special attention. The major issues to look for are applications that like to run under administrative context and have free reign of the computer they run on, applications that are locked to a specific Windows version number, or Web applications that require Internet Explorer 6. If you have already deployed Internet Explorer 8, and your users generally have Standard User accounts, then you won't have as much work to do. You can find more information about why applications may experience issues running on Windows 7 in the following guide on TechNet: [Understanding Application Compatibility].

After we find out what is not working, we have a couple of options to make them work:

- For packaged applications from ISVs, the best approach is always to find an application that runs natively on the version of Windows you want to install it on. Sometimes there are free updates for these applications and sometimes not. Using the application as intended and tested by the ISV for Windows 7 ensures that the ISV can support your users and you are running their application in a way that they have tested it.

- For in-house developed applications, the best approach is to recode the application to make it run natively. If you don't have the source code or there is an easy fix, you can use compatibility fixes (or "shims") to get the application to run without recoding it.

For more information about shims, see the following TechNet site: Managing Shims in an Enterprise.

Also check out the video about commonly used application shims: Mitigating Application Compatibility Issues with Common Compatibility Fixes

If you just can't make an application work, then it isn't necessarily "game over." If you've exhausted all other options to make the application run natively, then you can use Virtual PC to run the application in a Windows XP environment. Virtual PC with RemoteApp integration is much more intuitive for end users than it has been over the years. With Virtual PC, we can now publish shortcuts on the physical computer's desktop or Start menu to applications that are contained in the virtual machine, and applications can be launched individually without exposing the whole Windows XP desktop. The trade-off is that you'll have two operating systems to manage per user.

If you get to this point and you have a managed environment, I'd recommend that you look into Microsoft Enterprise Desktop Virtualization so you can manage the virtual environment. For more information, see Microsoft Desktop Optimization Pack.

After you've worked through all of your applications—inventoried them, rationalized them, and mitigated incompatibilities—you might think the fun is over. Almost. Now that everything is known to be working, you'll want to figure out how to install your applications in an automated way. In the next section, I'll talk about approaches to get to a thinner image. But if you've been packing applications into your base operating system image and performing sector-based captures of your reference computers' disks, then you may want to look at ways to reduce your image count and use the hardware and language neutral benefits in Windows 7 to get a single image. Getting to a single image in a larger organization typically means application packaging work is required.

Application Packaging

For some, application packaging and figuring out how to automate the installation of your applications will be as easy as finding the silent installation commands from the vendor. Usually these are found in installation guides, Internet forums, or by using the ever-handy "/help" or "/?" commands in the command line.

For in-house developed applications, there is a decent chance that silent installation commands don't exist for all of your applications and those applications will need to be packaged for the first time or repackaged if the installer package didn't work with your new configuration. (Common examples are 16-bit installers when moving to a 64-bit operating system or operating system version checks in packages looking for Windows version 5.1).

There are a couple of tools to help you create Windows Installer packages as easily as possible. These are handy because they tend to follow normal msiexec.exe silent installation commands. Microsoft Application Virtualization also provides what is essentially a packaging mechanism with the application sequencing it uses to create virtual applications. For more information about these tools, see the following Web sites:

- Flexera Software AdminStudio
- Microsoft Application Virtualization

You can avoid packaging some applications by not including those applications in the standard operating system build process and by prestaging installation files locally or making them accessible to users on the network. In the cases where everyone in the organization needs the application anyway, you can install those applications on your reference computer and create a custom image using ImageX. We'll talk about the balance of how many applications to include in the custom image in the next section.

Choosing an Image Strategy and Building Windows 7 System Images

Now that we've introduced deployment, data migration, and application compatibility, let's focus on imaging. Now this is not the imaging that involves photos and cameras, but the imaging of computer disks.

Quick History Lesson for System Imaging

Imaging tools have been around for a long time, and the most basic tools essentially back up an entire hard drive, sector-by-sector. Then we can restore that drive, if desired, to the same computer or another computer. This is basically a form of drive cloning, and it was popularized in the 1980s and 1990s. Like I said in the first section, this type of imaging is fairly archaic when used for deployment in the enterprise, because you'll need to maintain an image per Hardware Abstraction Layer (HAL) type. For people managing Windows XP, you'll often see an image per language or region as well. What does this mean? Usually it means tens or hundreds of images to manage for many organizations, all requiring maintenance when "Patch Tuesday" or similar events or updates come around.

But sector-based imaging can't be that bad right? Well, let's say you have everything centralized and you have 20 images to manage and up to 20 computers in your lab. When that critical update hits, we'll spend an hour rebuilding each of those computers, maybe an hour configuring them, and then up to three hours recapturing the image with sector-based imaging tools. That means 100 hours per month if you maintain images monthly and 1200 hours per year. To be fair, you aren't clicking and configuring things manually the entire time, but it's probably fair to say you'll spend two hours per system across all tasks, and you'll eventually have terabytes of system images to find space for. If you are using the System Preparation Tool (Sysprep) to generalize the image for installation on other computers, you get only three passes of the tool per system image over its lifetime. You generally need to capture each image before running Sysprep and afterward, so you can start next month with an image before you ran Sysprep, or else you need to start completely from scratch each time and apply all the changes since the last service pack.

Fast forward to 2003 when engineers were determining the future of system imaging, and along comes the Windows Imaging Format or "WIM" file. At the time, I was working with the Systems

Management Server (SMS) and Solution Accelerator teams at Microsoft, and WIM was a prerequisite for the Operating System Deployment Feature Pack released in 2004. WIMs are file-based and compressed images that can also save the contents of a drive. WIMs used with Windows XP were a pretty good option from a deployment standpoint, based on the reduced image size and ability to pass that package over the network, but they were still tied to one Hardware Abstraction Layer (HAL) type.

Fast forward to around 2006 and the early iterations of Windows Vista…

WIMs used for Windows Vista and Windows 7 imaging and deployment take on a whole new meaning. Remember those tens or sometimes hundreds of images to maintain and up to five hours per month per image? With Windows Vista and Windows 7, you can get down to a single image per operating system architecture (for example, a 32 bit image and/or a 64 bit image). As an example, right now I am in an airplane writing on a Fujitsu U820 ultra mobile Tablet PC uniproc that I built using the same image I've applied to my bigger and less airplane-friendly Lenovo T60P 15" multiproc laptop, in addition to countless other hardware types.

But it gets even better than only a single image to manage for all hardware (and languages by the way, too). Remember the five hours or so we would spend building, configuring, and recapturing that old sector-based image? We can mount the file-based images of Windows Vista or Windows 7 to a file folder and service them offline. In other words, I need one computer in my lab to use as a reference machine for all computers, I can use a free tool in the Windows Automated Installation Kit called ImageX to capture and apply system images, and I don't necessarily even need to use that reference computer in my lab to service my one image on Patch Tuesday. I can mount the image in a folder on my image storage server, use the Windows 7 and Windows Server 2008 R2 in-box tool called dism.exe ("Deployment Image Servicing and Management," in case you're wondering) and enumerate the contents of the image to see packages, updates, drivers, and features. I can then modify these contents offline by using dism.exe—again without building that reference lab computer. Those five hours it took you to apply the three critical patches on Patch Tuesday can take as little as about two minutes to mount the image, ten minutes to service it, and two minutes to unmount it. I'm usually pretty happy if I can save four hours and 45 minutes performing an otherwise boring, but necessary task. And instead of doing it 20 times and using 20 physical computers, I do it once. Makes sense, right?

To show some of that, here is a video of Sysprep and ImageX that shows how to generalize and capture a custom image: Preparing an Image using Sysprep and ImageX

Here is a video that shows dism.exe servicing an offline mounted Windows 7 image: Deployment Image and Servicing Management

I had to take a brief excursion from the deployment task at hand to give the history lesson, because in all my interactions and talks with IT pros and my desktop admin friends lately, I see two common issues when it comes to imaging:

1. The majority of people I talk to are still using the sector-based imaging tools they've been using for decades.
2. The majority of people aren't maintaining Windows Vista or Windows Server 2008 images, so they aren't able to perform offline image management.

Even more troubling are the situations where Windows Vista or Windows Server 2008 are in place, but people are using the 20-year old tools and processes to manage them. They aren't using or aware of Sysprep, so they need an image per HAL type or lots of luck that the image that has not been prepared with Sysprep installs on foreign hardware. (This scenario without using Sysprep, albeit unsupported, is still somewhat common).

Building Your Image

Windows Vista and Windows 7 are delivered by a file-based WIM image and image-based setup. That DVD you might have or the ISO file you downloaded contains a 2+ GB file called install.wim in the Sources directory. The amazing thing about this WIM is that it actually can contain multiple operating system captures. In fact, the Windows Server 2008 R2 Enterprise image contains eight operating system variants and the 32-bit edition of Windows 7 Ultimate contains five variants.

This would normally be larger than the Windows 7 Enterprise install.wim with one variant or a custom captured image with a single operating system image, right? Not really. WIMs use single instancing of shared files, so you can have multiple operating systems available in an image that might be about the same size as one captured operating system.

This is important as you determine your image strategy, because you could, for example, have multiple operating systems of differing languages packed into a single WIM file, and even with multiple languages these should only be marginally larger than a single language WIM image. WIMs can also be used to compress and deliver data, so you can package multiple applications, drivers, and packages into the data WIM, then mount and call them at install time by using scripted operating system installations.

Now that you know a bit about WIM files, let's cover the basics of imaging strategy. There are three primary strategies used for imaging and all are valid depending on the use case:

1. **Thick Image.** I like to refer to this as the "old school" approach to imaging where you basically build a reference machine, install all possible applications to ensure users have the applications they could ever possibly need and usually more. After that is done, you apply software updates for the operating system and all the applications, then you use Sysprep on the computer to capture the image. Then you make sure everything works and ensure that Sysprep didn't affect any applications.
2. **Thin Image.** This approach takes things to the other extreme. Little or nothing is installed on the reference computer, and you use Sysprep to capture that image. Or some will just use the image as shipped in the Windows 7 retail DVD or ISO with zero customization. This strategy assumes that you'll be customizing the installation with applications and other necessary data dynamically at deploy time. This also means that all of your applications are packaged for an unattended installation, or you are willing to prestage them for users to install when they want, or you use something like Application Virtualization (App-V) so application profiles follow users regardless of the device they log on to.
3. **Hybrid Image.** In between Thick and Thin is a Hybrid Image, where applications that everyone uses or needs are captured in the base image (perhaps your VPN software, your

antivirus software, your version of Microsoft Office, and the App-V client). Aside from those core applications, additional applications are layered on at deploy time based on user needs.

All three of these strategies can be justified, though I personally tend to favor thin images. The thick image approach is useful in situations where the company has a homogeneous environment, uses a single language, and all users use and need exactly the same set of applications. When using thick images in larger organizations, the trade-offs are that you pay for several applications that may not be necessary for all users, images are larger and multiple applications can affect performance, plus the image is more difficult to maintain, and flexibility is greatly reduced.

Thin images are the most flexible and easiest to maintain, but customizations need to happen at deploy time, and that means that applications are packaged for a silent install and application updates can be installed silently as well. Installation speeds can be slower compared to thick images because each application needs to install itself one-by-one at deploy time and more automation is required. Hybrid images include many of the components of thick images, without necessarily wasting licensing costs, required disk space, and often the performance hit of multiple unused applications.

Getting to Thin Images

If you currently use thick images, you might be asking, “What tools are there to move to thinner images then?”

Enter deployment task sequencing. Recognizing the limits of using thick images, many people have developed task sequencing engines to not only install applications, but also perform the other common operating system deployment tasks in an automated way. Task sequences are extremely important for the Computer Refresh and Computer Replacement scenarios, because they provide the following:

1. Validate that the target hardware can install the operating system
2. Capture user files and settings
3. Invoke an installation environment like the Windows Preinstallation Environment (Windows PE)
4. Customize the installation environment
5. Apply the operating system image
6. Apply drivers required by the hardware and connected devices
7. Apply software updates
8. Apply applications based on your selections
9. Join the computer to a domain
10. Reapply user files and settings
11. Configure additional attributes such as BitLocker™ Drive Encryption or server roles

All of this is completely automated by using deployment task sequencing—you launch it for a minute or schedule it centrally if using System Center Configuration Manager and the rest just

happens without you needing to touch the computer. For someone new to the space, it sounds difficult to get configured, but these are standard in-box task sequences, from the following sources:

- Microsoft Deployment Toolkit
- System Center Configuration Manager (the enterprise-class console)

Here's a video of what preparing a build looks like by using the Deployment Workbench in the Microsoft Deployment Toolkit 2010: Deployment Workbench in Microsoft Deployment Toolkit 2010

The task sequence brings together the tools we need for the deployment to end-to-end. I like to think of everything we're using in terms of music. If you think of unattend files, the User State Migration Tool, Windows PE, applications, and drivers as instruments, then the task sequence is like the conductor and sheet music. The end product is a symphony of automation that you have complete control over. When everything is finished and ready for automation, you can pick how you want to deliver your builds.

I'll conclude this section with a video that shows a fully-automated migration with user data from Windows XP to Windows 7 that I built myself (but did not narrate) using the free tools described previously: Windows XP to Windows 7 Migration

Automating the Migration from Windows XP to Windows 7 End-to-End

This is the fifth and final part of the series about deployment and all of the subtasks you need to perform. Admittedly, I set out to write this on one page, and I think with small font and a decent-sized plotter, you may just be able to get this to print onto a single page, but most people won't. For the astute readers out there, you might be thinking two things, "I just read several pages of what is essentially intro content?" or "Wait, Jeremy, you forgot all about drivers!"

To the first question, I was on the team that originally wrote the ominous "1500 pages of desktop deployment guidance" back in the Business Desktop Deployment (BDD) 2.0 and 2.5 days. We've "streamlined" that down to a couple hundred pages, but in doing so, we left out most of the content about project management, business cases, Microsoft Solutions Framework, and so on. The result is the often-requested, "click-here, do this" guidance, but if you crave the all-up project management best practices content, we've posted archived versions of MDT and BDD releases:

- See the abbreviated version of the MDT: Microsoft Deployment Toolkit 2010
- See the full versions of the BDD and MDT: Business Desktop Deployment and Microsoft Deployment Toolkit Archive

If you're one of those guys who refuses to use stuff branded as Windows XP SP2 or Windows Vista because it doesn't look current, the archives may not be for you. If you want sample project plans and project templates for deploying an operating system and you are alright replacing an occasional "Vista" with a "7," then you'll be able to get value. Here's a secret: the full set of

processes for deploying Windows XP is roughly the same as for deploying Windows 7. Sure the tools have gotten better and cover more areas and scenarios and the operating system is more flexible for imaging and offline servicing, but the overall process is basically the same it has been for years.

Remember the “BDD Wheel?”

All of this still applies today to whatever operating system you are deploying. Even though I have the source files for this image that we built using Microsoft Office Visio® in 2003, fundamentally, I could use that original graphic today with Windows 7. That is why I’ve covered most of these areas in the first four sections.

Automating the End-to-End Migration Process

I will answer the driver question as I go into the process automation. To be honest, I want to downplay drivers a bit, because if you’ve installed Windows 7 (or even Windows Vista), you’ll know that in-box driver coverage is in a whole new league compared to Windows XP on current hardware. The important items like mass storage and network drivers are pretty well-covered, as are most other driver categories. You will need to ensure that display drivers are included in your process and that any other OEM-specific applets you want to keep are there, and we’ll cover that in a minute.

Relatively soon, you’ll need to think about the Windows activation method that you’ll be using for Windows 7. Windows XP activation usually entailed that you bake a key into your unattend process. You can still do that with the revised unattend.xml by using Windows System Image Manager and a Multiple Activation Key (MAK), or you can set up a Key Management Service (KMS). I recommend KMS, because we don’t need to worry about managing keys in the build

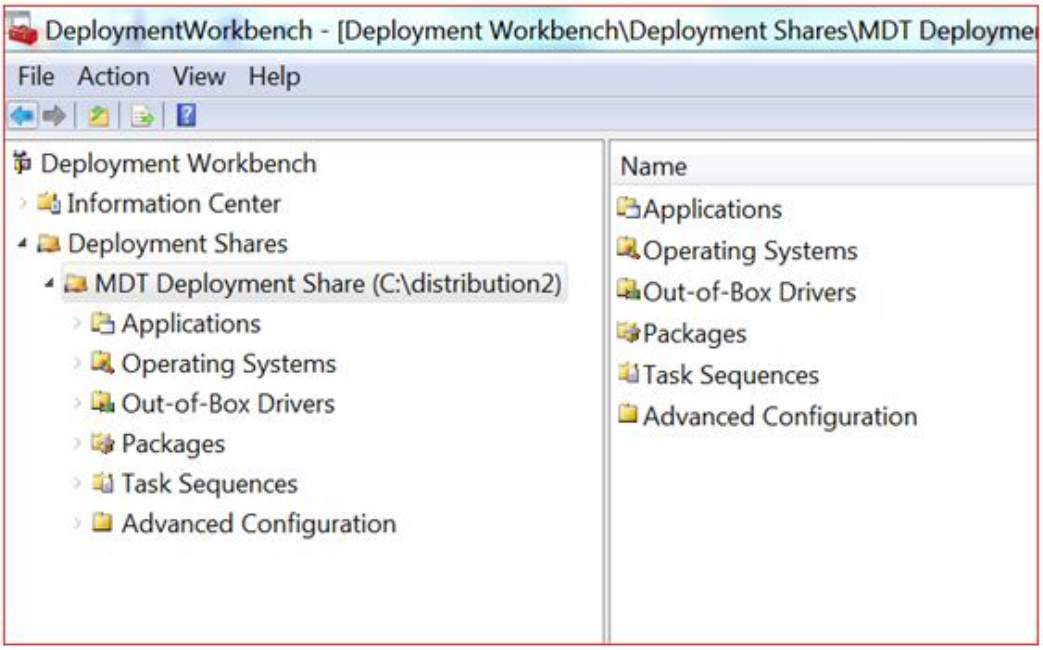
process. When the system is online and can reach the KMS server, it activates automatically—no keys to manage or potential key leaks during the deployment itself.

Read more about this in the following guidance: Volume Activation

I was using “fun” a bit sarcastically in previous sections, but automating processes is truly the fun part of deployment for me. Whether you are using Microsoft Deployment Toolkit 2010 or System Center Configuration Manager 2007 SP2 to build your deployment task sequences, you basically need to add the same ingredients to your build:

1. An operating system
2. Applications
3. Drivers
4. Packages

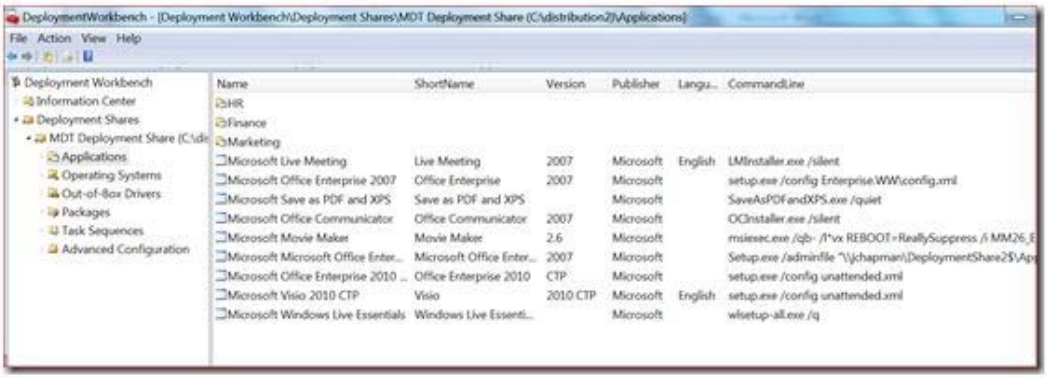
Here is what that looks like in the MDT 2010 Deployment Workbench:



I actually like my sequence of ingredients better, because the operating system is the only “required” piece of the four I listed. If you have a thick image with applications, and you only want to use the task sequence to capture and reapply the user state and maybe pull Microsoft software updates from your update server or Windows Update, you can get by with only the operating system image. For a thin image, you can get by with importing a set of flat Windows 7 retail source files.

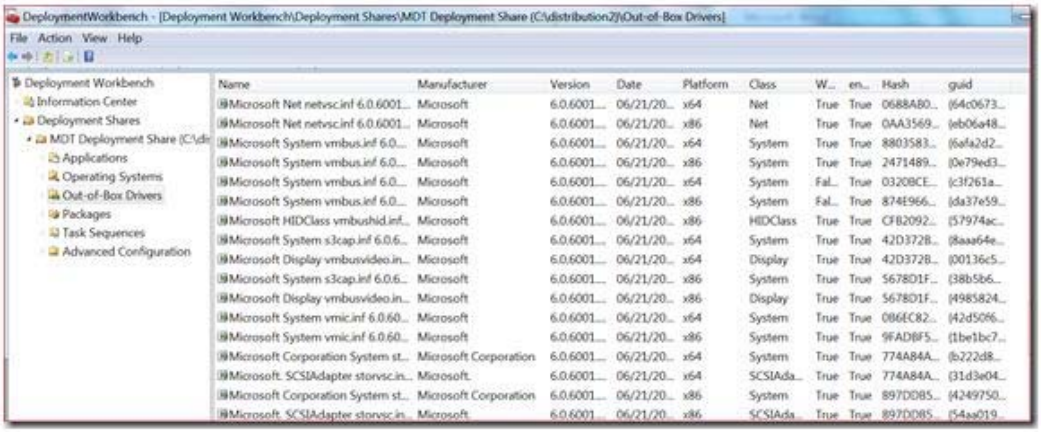
The next step we take is to add our packaged applications. We need the application source files and the silent install commands. We can also set up application dependencies. For example, if Microsoft Office Communicator is dependent on Microsoft Office 2007, by choosing to install Communicator, we’ll automatically preclude that by the entire Office package.

Like everything else in the Deployment Workbench, you right-click the item, select “New Application” and follow the wizards. When you’re done, you’ll have something that looks like the following screenshot:



As you can see from the “CommandLine” field, install command lines vary dramatically depending on the application you’re installing. Now is also the time where you want to check for drivers that were shipped as application installer packages and you’ll be treating them as applications in this case.

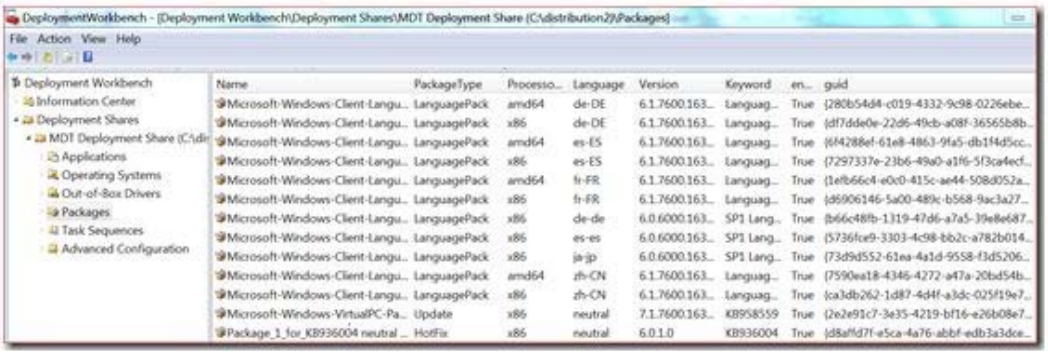
Now you can add “normal” drivers with .inf extensions to the “Out-of-Box Drivers” menu. Drivers will be installed automatically by default based on their Plug and Play (PnP) IDs, or if you want total control, you can group them by hardware make and model using Selection Profiles in MDT 2010. After you’ve imported drivers into the Deployment Workbench, it will look something like this:



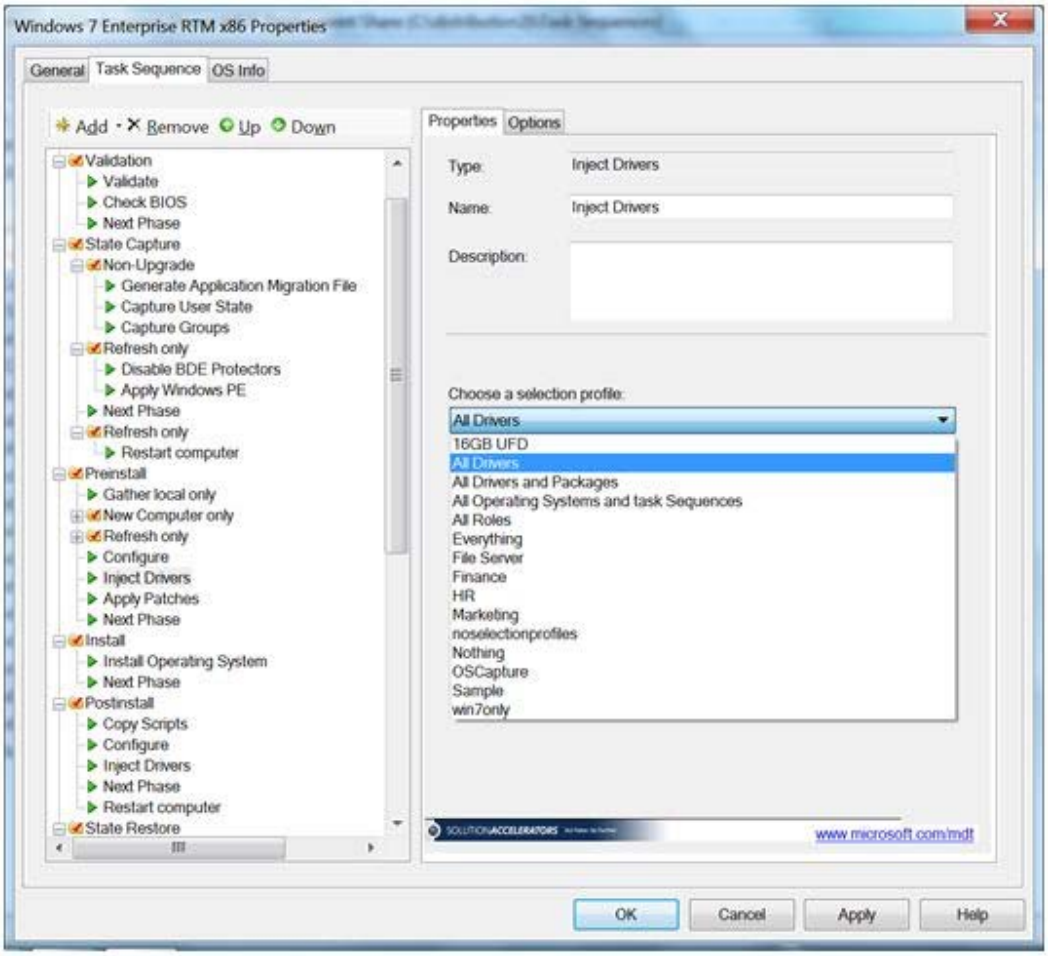
If you are using a straight Windows Deployment Services environment with Windows Server 2008 R2, we can also use the driver store in Windows Deployment Services to keep drivers centrally on our deployment server and have them install based on PnP IDs. Likewise, System Center Configuration Manager has a great driver store to control how drivers are installed. The last step you can take for driver installation is to opt through the deployment task sequence to pull from Windows Update for updates and drivers. Chances are, between these

approaches and an optional call to Windows Update, that even untested devices will successfully install with most (if not all) drivers ready to go.

The last ingredient we'll add to our build are packages. Packages can be Cabinet (.cab) or Microsoft Update (.msu) files. There is a mechanism to do this in the Deployment Workbench and in Configuration Manager. When we detect language packs in MDT, we'll even expose them as optional packages (as opposed to installing everything that qualifies for our operating system). After you've added packages, you'll see something like this with varying package types:



At this point, we have all of our ingredients. MDT 2010 and Configuration Manager also have vital components like USMT and Windows PE to migrate user files and give us an installation environment for laying down the Windows 7 image. If you want to customize Windows PE, you can do that by using MDT 2010, however, MDT will automatically create the custom Windows PE environment for you as part of the standard build process. I introduced task sequences as the glue that combines all of the migration processes together, and I listed the core steps in the last section. Making all of this easier is the fact that the standard client and server task sequences in MDT and Configuration Manager will do essentially everything they need to do, and in many cases, you won't need to create any custom tasks. Here is what the task sequence looks like and everything it does:



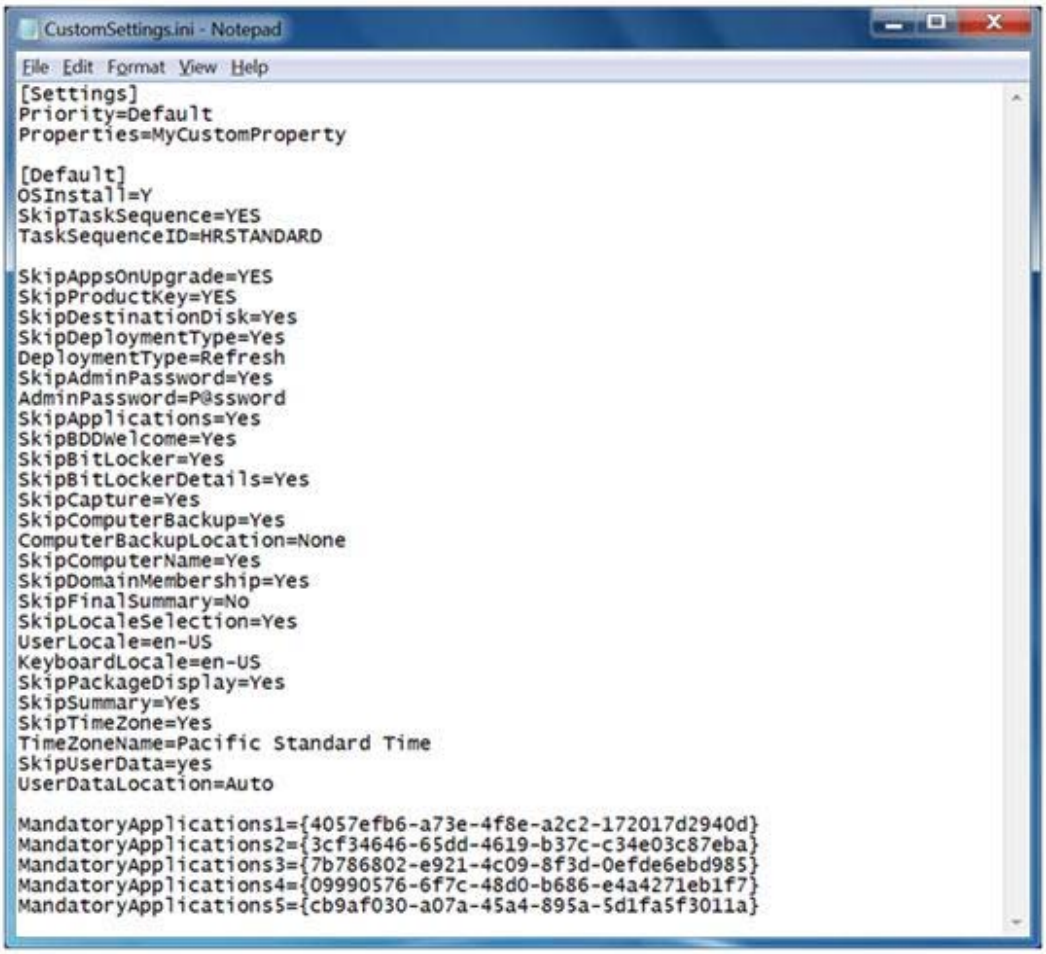
Among all of the steps, there are a couple of “Inject Drivers” steps that allow you to either rely on the PnP ID mechanism by default or choose a selection profile from the drop-down list as seen previously. After you’ve created the task sequence and everything else is done, you can start testing. Now you can use a file server, Windows Deployment Services, or standalone media to deliver your Windows 7 builds. This will perform all the steps I listed in the last section, and you’re on your way to migrating Windows XP computers to Windows 7 in a highly automated way.

Tricks for More Automation

Before I finish this article, I want to cover one last idea. When you get everything working by using MDT 2010, the first thing you’ll probably want to do is remove some of the Lite Touch Deployment wizard screens like I’ve done in following video: Windows 7 Deployment Tools Part 4.

Here is the trick. You can modify a file called “customsettings.ini” in your MDT 2010 distribution share. The file in located in the …Deploy\Control folder. By using this file, you can eliminate one, many, or all of the wizard screens by prepopulating the fields that you would otherwise have to fill

in, or you can skip screens and use the defaults in other cases. A completely automated customsettings.ini file looks like the following screenshot:



I've also added mandatory applications to control which applications are part of this build. By using a similar customsettings.ini file with the standard client refresh task sequence, you will automate all migration steps.

With that, you have the basis to start automating migration from Windows XP to Windows 7. In less than 20 pages, I have covered a lot of ground, and I've been pointing along the way for more guidance per task. If you've gotten value out of this series, I would encourage you to:

- Subscribe to our blog: Deployment Guys Blog
- Keep checking back to the TechNet site for the latest content: Springboard Series for Windows 7 on TechNet