Tivoli Provisioning Manager for OS Deployment
Version 7.1.1.16

Getting Started
Tivoli Provisioning Manager for OS Deployment
Version 7.1.1.16

Getting Started
Note
Before using this information and the product it supports, read the information in "Notices" on page 21.
Chapter 1. Product overview

Tivoli® Provisioning Manager for OS Deployment is a database-driven, network-based deployment solution.

Using Tivoli Provisioning Manager for OS Deployment, you can perform the following tasks:

- Cloning and unattended setup:
  - Windows operating systems
  - Linux operating systems
  - Solaris operating systems
- Unattended setup:
  - AIX
  - VMWare ESX

In addition to BIOS targets, you can also manage Unified Extensible Firmware Interface (UEFI) enabled targets.

Using industry standards such as Wake on LAN and vPro, PXE and OpenBOOT, ODBC and JDBC, DMI and PCI, Microsoft system preparation tool (Sysprep), Kickstart, Autoyast, Jumpstart and NIM, you can install operating systems and selected software on hundreds of computers simultaneously.

You can deploy from a CD or DVD, disk partition, or a network with either unicast or multicast downloading.

Components

You can familiarize yourself with the components of the product.

The product is composed of:

- Server-side components installed using a typical setup program
- Web interface components
- Target-side applications automatically installed by the target through the network

The server-side components are:

- The OS deployment server, running in the background as a service. This component provides the PXE remote-boot capability and multicast file access to the remote-boot targets.
- A deployment database, accessed through the standard ODBC/JDBC interface, and through the product TCP-to-ODBC/JDBC Gateway service. This database stores the Bill of Material for every target, including information about what software modules must be installed.
- The web interface used to prepare and supervise deployments. Using the web interface, you can configure the OS deployment server, manage objects used during a deployment, and create recovery CDs and DVDs. You can also see targets in their different groups. You can control the parameters and status of each target, for example, when a target requests a remote-boot through PXE.
The web interface, typically installed on the server, uses the web interface extension, running in background as a service. Optionally, you can install the web interface extension on targets, so that you can take control of them remotely. The web interface extension is useful, for example, to restart the computer when a deployment starts, or to perform an operating system inventory. Several features of the web interface are not available when the web interface extension is not installed on the server. The web interface extension is designed to run as an operating system application, that is, a Windows operating system application or a UNIX operating system application.

The target-side tools are:

**Deployment engine**

The deployment engine is a stand-alone mini-operating system used for all management tasks to be performed outside of the real operating system, such as operating system installation. You never have to install this deployment engine manually on target systems, because it is downloaded automatically when target systems are instructed to boot on the OS deployment server in their DHCP configuration. After the deployment engine is started, it loads a ramdisk which can perform operations on the target, such as:

- Partition and format disks
- Install a complete Windows or UNIX operating system
- Clear the disks

The deployment engine is a stand-alone management platform and does not require an operating system.

**Web interface extension**

The web interface extension runs on Windows, AIX, Linux, and Solaris operating systems, either in the background as a service or as a command-line tool. It provides cross-operating system connectivity to the management platform for targets with an installed operating system. It can be used to collect information from targets, to remotely initiate tasks on a target from the OS deployment server, or to trigger commands on the OS deployment server from a target.

The web interface extension can access all files on the hard disk and other drives through the operating system. On Windows operating systems, it can also access the system registry, and all system information published by drivers using the Windows Management Interface (WMI). In addition, for all Intel-based operating systems, the web interface extension can install a hook on the boot process to force a PXE boot or to start the deployment engine in offline mode, to trigger pre-operating system management tasks such as migration or recovery.

When you install the product you are installing the OS deployment server. The target side is embedded into the server, and sent to computers through a network connection such as a network boot or PXE boot, or through media such as a CD, a diskette, or a hard disk partition.

**Note:** You can boot targets over the network without using PXE.
The following steps describe the installation process:

1. The installer, containing server-side and target-side components, installs the server-side and potentially the web interface extension on the server computer.
2. You can then create a CD can containing both the deployment engine and the web interface extension.

3. Computers boot, loading the deployment engine either from the PXE server (OS deployment server), or from the CD.

4. The deployment engine loads a ramdisk, Linux deployment engine for Linux, WinPE for Windows.

5. The ramdisk can now perform tasks, including operating system installation.

6. The targets boot into the Windows, Linux, or Solaris operating system, with the web interface extension optionally running.

Product topology

The product topology to automate remote deployment of operating systems and virtual images can be as simple as a single LAN implementation or a more complex implementation involving a large enterprise with multiple subnets with remote sites connecting through a low-speed communication link.

When planning your topology, consider the following issues:
- Operating system of the computer where the OS deployment server is installed
- Location of the database and the name of the database engine, for example, DB2, Access, and so on.
- Single OS deployment server or a hierarchy of servers
- Location, type, and version of the browser

The following diagram shows a typical product topology with a single LAN, a single OS deployment server separate from the DHCP server, and a number of target computers.
Chapter 2. Setting up a system profile by unattended setup

Use unattended setup to set up a system profile without actively being involved when the process occurs.

To set up an automated system profile by unattended setup, performing the following steps:
1. Install the OS deployment server on a computer.
2. Create a system profile by providing the installation CDs and choosing installation options.
3. Create software modules for most commonly used business applications and other software, using the web interface.
4. Optional. Design automatic binding rules and fixed configuration parameters for fully unattended deployment on unknown targets.
Chapter 3. Setting up a system profile by cloning

A cloning-mode system profile requires you to configure a target, prepare it for cloning, and run the cloning process directly on the computer. However, the native mode of operation of Tivoli Provisioning Manager for OS Deployment is centered around cloning-mode system profiles, because this method of deployment is faster than unattended installation.

To set up an automated deployment system by cloning, perform the following steps:

1. Install the OS deployment server on a computer.
2. Install a clean operating system on a target, and run the appropriate system preparation tool, if necessary.
3. Restart the target with network-boot enabled. Instead of its own operating system, the Tivoli Provisioning Manager for OS Deployment kernel must start on the target (remote-boot through PXE).
4. Create a system profile and OS configuration with the clean operating system.
5. Create software modules for most commonly used business applications and other software, using the web interface.
6. Optional. Design automatic binding rules and fixed OS configurations parameters for fully unattended deployment on unknown targets.
Chapter 4. Choosing how to boot your target

Tivoli Provisioning Manager for OS Deployment implements a PXE network boot application. PXE is a type of DHCP extension. It is supported by the vast majority of hardware vendors, including IBM®, Lenovo, HP, and Dell.

You can run a Preinstallation Environment (PXE) boot on your target from either the network or from media.

**Network PXE boot**

You can boot a target from the network using a *kernel* or *kernel free* mode. Depending on the type of network boot required, a different program is run to contact the OS deployment server and deploy the system profile on the target computer. To run the PXE boot from the network, ensure that the DHCP server is configured.

Network booting for both BIOS and Unified Extensible Firmware Interface (UEFI) targets is supported. For the targets that support both booting modes, depending on the target configuration, the boot might occur in either BIOS or UEFI.

**PXE emulation media in a non-PXE environment**

If the network card in a particular target system does not support a PXE boot or if PXE is not available for policy or security reasons, you can build a boot media on the OS deployment server, and use it to boot the target computer and connect it to the OS deployment server to deploy an image. After the boot, the behavior of the computer is identical to any PXE-booted system.

PXE emulation is currently not supported for UEFI firmware.

**Types of PXE network boot**

You can boot a target from the network using a kernel or kernel free mode.

You can boot a target from the network using one of the following modes:

**Kernel mode (BIOS only)**

During the boot, the preload program automatically starts an OS Deployment kernel on the target to download and install either the Windows or Linux operating system according to the information stored in the OS deployment server database. This kernel is a mini operating system that contacts the OS deployment server and runs the deployment on the target machine. The advantage of using this kind of boot is that the download of the deployment engine is faster than the download of the deployment engine of the kernel-free mode.

**Kernel-free mode (BIOS)**

During the boot, the preload program automatically starts the PXE Linux tool on the target to download and install either the Windows or Linux operating system according to the information stored in the OS deployment server database. This tool, which is part of the SysLinux suite, allows the Windows or Linux kernel to be started without using any OS deployment server program. During the target reboot the following window is displayed:
The advantage of using this kind of boot is that it minimizes hardware compatibility issues.

**Note:** If you boot in kernel-free mode, you have the following limitations:

- Redeployment is not supported.
- In a Linux virtual machine hosted on an ESX hypervisor, Windows deployment tasks, such as destroy hard disk contents, are not supported because Linux settings prevent WinPE from functioning correctly.
- Only kernel-free hardware configuration rules based on the workstation model are handled and not rules based on the workstation devices.
- The deployment menu on the target is not available if the target is started in kernel-free mode.
- The target kernel-free boot is not supported for KVM 5.7 guests and earlier versions. It is only supported on KVM 6.0 guests and higher versions.

**Kernel-free mode (UEFI)**

During the boot, the preload program automatically starts the UEFI switcher tool on the target to download and install Windows operating system according to the information stored in the OS deployment server database. This tool allows the Windows to be started without using any OS deployment server program. During the target reboot the following window is displayed:
The advantage of using this kind of boot is that it minimizes hardware compatibility issues.

**Note:** If you boot in UEFI kernel-free mode, you have the following limitations:
- Only Windows deployment is supported.
- Redeployment is not supported.
- Only kernel-free hardware configuration rules based on the workstation model are handled, not rules based on the workstation devices.
- The deployment menu on the target is not available if the target is started in kernel-free mode.

The following table describes the operating systems that you can install by booting a target in a kernel or kernel-free mode:

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Kernel mode</th>
<th>Kernel-free mode (BIOS)</th>
<th>Kernel-free mode (UEFI)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows Server 2008</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Windows 7</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Windows 8</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Windows Vista SP1 (or later)</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Windows Server 2003 SP2 and R2 SP2</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Windows XP Professional</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Windows Server 2012</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Windows Server 2008 R2</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SUSE Linux Enterprise Server (SLES) 11, GA SP1, SP2, and SP3</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SUSE Linux Enterprise Server (SLES) 10 GA and SP1, 2, 3, 4</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SUSE Linux Enterprise Desktop (SLED) 10, GA and SP1, 2, 3, 4</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>
Table 1. Operating systems you can install by booting a target in kernel or kernel-free mode (continued)

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Kernel mode (BIOS)</th>
<th>Kernel-free mode (BIOS)</th>
<th>Kernel-free mode (UEFI)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red-Hat Enterprise Linux (RHEL) 6, GA and Updates 1,2,3,4, 5 Server, Client, Workstation</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Red-Hat Enterprise Linux (RHEL) 5 GA and Updates 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 Server, Client, Workstation</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Red-Hat Desktop 10, GA and SP 1,2, 3, 4</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Red-Hat Enterprise Linux (RHEL) Server 4, Update 8,9</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>VMware ESX Server 4.0, 4.1</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>VMware ESX Server 3.5 Update 4</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>VMware ESX Server 3.0.2</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>VMware ESXi Server 4.1</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>VMware ESXi Server 5, 5.1, and 5.5</td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

Note: Use the kernel-free mode to deploy:
  - SLES 10 64-bit on an IBM x3850 server
  - SLES system profile on an HP Blade HPBL460c G6

The following table describes the tasks that you can perform by booting a target in a kernel or kernel-free mode:

Table 2. Tasks you can perform by booting a target in a kernel or kernel-free mode

<table>
<thead>
<tr>
<th>Task</th>
<th>Kernel mode (BIOS)</th>
<th>Kernel-free mode (BIOS)</th>
<th>Kernel-free mode (UEFI)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perform hardware configuration</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Deploy unattended setup (Windows)</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Deploy unattended setup (Linux)</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Deploy unattended setup (VMWare ESX)</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Deploy unattended setup (VMWare ESXi)</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Deploy cloned profile</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Deploy unattended or cloned profiles: redeploy mode</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Capture a reference machine (cloning)</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Collect RAID inventory, Hard disk destroy</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Network boot process

Understand the process for Intel/AMD x86 computers, Intel/AMD x86-64 bit computers based on the Unified Extensible Firmware Interface (UEFI) standard, SUN SPARC and IBM PowerPC® machines based on the OpenBoot standard.

A network boot application is unique for a number of reasons:
• It is invoked by the BIOS or UEFI firmware during the initial bootstrap of the computer, before any operating system or disk boot manager.
• It depends on the presence of a special chip on the network card: the PXE boot ROM.
• It can communicate with a server over the network because of a programming interface provided by the bootrom. PXE is the informal standard for this kind of programming interface.
• It is downloaded from a special OS deployment server, and does not depend on any local storage device. It can work on computers without hard disk, CD, or diskette devices. If a local storage device is present on the computer, it is accessible to the network boot application. However, a network boot application does not fail if the storage device is corrupt or broken.
• It gets its IP and other boot parameters from a central DHCP server.

A typical network-boot sequence goes through the following phases:

1. **Start up:** A user or a wake-up event starts the remote-boot computer up.
2. **Network boot:** The BIOS configuration or boot order, a hot key, for example, F12, or the wake-up event or the UEFI boot sequence instructs the computer to boot on the network.
3. **IP address discovery:** The remote-boot target broadcasts a DHCP request for an IP address. Any DHCP server that recognizes the hardware address of the target or that has a pool of freely distributable dynamic addresses, sends an IP address. The target takes the first response and confirms it to the server. In addition to the IP address, the server gives some other network parameters to the target and information about the boot procedure to follow.
4. **OS deployment server discovery:** In the case of PXE remote-boot, the target computer then proceeds to discover the OS deployment server. The OS deployment server is responsible for delivering a network boot program to the target. It is not necessarily the same computer as the DHCP server. The target responds to the first OS deployment server which replies and downloads a small program using a simple multicast protocol (MTFTP).
5. **Server connection:** If the network boot program is the deployment engine, it establishes a secure encrypted connection to the OS deployment server and receives instructions from the server to determine the name of the program to run.
6. **Pre-OS configuration:** The deployment engine then downloads from the file server everything required by the OS configuration specified by the system administrator. These file transfers are done using a secure, robust and efficient unicast or multicast protocol. You can perform many actions at this stage: installing an operating system from scratch, repairing a corrupted operating system, or performing an inventory.
7. **OS booting:** When instructed by the OS configuration, the deployment engine removes itself from memory and allows the computer start the operating system, as if the target is booting normally from the hard disk. This ensures full compatibility with the operating system and avoids all problems of the traditional diskless remote boot.

**Note:** Tivoli Provisioning Manager for OS Deployment has a fault-tolerant server architecture, with the OS deployment server always available to the target. However, in case of severe network failure, targets can be configured to work offline, without network access. In this scenario, the deployment engine is stored on a permanent storage medium, such a hard disk partition or a CD, and started from that medium instead of being loaded from the network.
Chapter 5. Deployment process

The deployment process includes hardware discovery, hard disk partitioning, installing operating systems, and installing other software modules.

A typical deployment is made up of a number of events.

You start the target remotely, possibly using Wake on LAN or vPro. The target starts up on the network. It receives IP information from the DHCP server, and a bootstrap program from the OS deployment server that directs the next steps.

The target performs hardware discovery through DMI and PCI system calls. The target queries the ODBC/JDBC database for all relevant information regarding the operating system and the software modules to be installed, and dynamically generates a deployment plan. If specified in the database, it prompts the user (or the administrator on the web interface) for additional deployment information.

Depending on the planned deployment steps, the bootstrap can start WinPE for Windows or Linux deployment engine for Linux in a ramsdisk to perform some of the configuration and installation tasks.

**Note:** To provision Windows Vista, Windows 7, and Windows 2008 operating systems use WinPE 3.x. To provision Windows 8 and Windows 2012 operating systems, use WinPE 4.x.

If specific hardware configuration software must be run early (such as RAID controller configuration), it is started. After a subsequent reboot, the target resumes the process. The target partitions its hard disk according to the information retrieved from the database. A small bootstrap is also written to the hard disk, to be able to take over any subsequent reboot on the hard disk. The target starts a multicast transfer for all files needed during the deployment.

The content of the hard disk partitions is built by merging the base operating system image and all incremental images and other software updates. The data is then written onto the disk with optimal efficiency. An answer file is dynamically generated, providing all necessary information for an unattended customization of the operating system. Unattended setup command lines for software modules are prepared.

The target computer boots on the hard disk, and completes operating system installation, then issues unattended setup command lines for the selected software modules. The target reboots and Tivoli Provisioning Manager for OS Deployment takes control again.

If other software modules have been configured to be installed after one or more reboots, they are installed and the operating system is started again. If the computer is still connected to the network, it automatically updates the deployment status in the database.

When everything is done, the deployment process terminates by either turning off the target, starting up the operating system, restarting the target, or displaying a green banner page.
Chapter 6. Universal images

With Tivoli Provisioning Manager for OS Deployment, you can create universal images for Windows operating systems.

When creating a system profile by cloning, Tivoli Provisioning Manager for OS Deployment makes a copy of the state of the hard disk of a particular hardware. The system profile thus created corresponds to a particular hardware type on which the profile can be deployed. In theory, a different image is required for each hardware type. Switching from a parallel hard disk on the cloned computer to a SCSI or a serial-ATA hard disk on the deployed computer is particularly problematic.

A universal image is a cloned image that has been prepared with all drivers for disk types and hardware abstraction layer (HAL) variants encountered in the pool of computers to be deployed. Because Tivoli Provisioning Manager for OS Deployment enables the addition, in real time, of the appropriate driver packages (driver injection) during deployment, the image cloned from one type of hardware can be made to work appropriately with hardware of another type. Therefore, a single cloned image can be sent to many hardware types. The driver selection and installation necessary for the image to work are performed automatically.

Advantages of universal images are:

Image storage space
Because Tivoli Provisioning Manager for OS Deployment can modify an image and add drivers through driver injection in real time during an image deployment, only one image and a collection of driver packages need storage space as opposed to an image for every hardware model. This is true for the parent server and every distributed copy in the network.

Image maintenance
Instead of building a new image every time a new model of hardware or driver is released, you are only required to package the driver, to set new rules for the deployment of that driver, and to test the deployment and rules.

Image replication
Minimal images mean that less time and resources are used to move these images around the network to where they are needed, on a target for deployment or on another provisioning server, for example.
Chapter 7. Shared repository and its cleanup

Managing the data directory of a Tivoli Provisioning Manager for OS Deployment server.

In Tivoli Provisioning Manager for OS Deployment, the shared repository reduces the storage space necessary to hold all server objects. The shared repository is a content-indexed dictionary of files where every file is individually stored, indexed by a global Message Digest 5 (MD5) and a subsidiary list of MD5s for each 32 KB block. In the Tivoli Provisioning Manager for OS Deployment disk image, the file content is not stored, but only the list of MD5s.

By using an MD5 algorithm to individually identify each file being stored in the image repository, you can eliminate the need to store duplicates of any file. For example, one Windows XP image can take 3 GB of storage, but two variations of an XP image might take less than 4 GB.

Shared files are not stored individually but are embedded in common pool files that are better cached by the system. There are also some index files, so that the system can quickly locate a file given its MD5 description.

**Note:** You can back up these files at any time, because after a content-indexed file is added, it never changes its location in the index. The consistency is therefore guaranteed at all times. This is, by definition, in a content-indexed repository: when a file has changed, it is considered to be a different file, because it has a different MD5 index.

With regular use, the shared repository typically grows as new files are added. If, for some reason, you want to remove files from the shared repository that are no longer used, run a mark and sweep procedure to first mark all the archives that you want to preserve, and then sweep all the other files out of the registry.

**Note:** This procedure is not useful on the target, except for special applications such as frequent local backups.

Tivoli Provisioning Manager for OS Deployment provides a number of primitives for running a mark-and-sweep.

To keep your Tivoli Provisioning Manager for OS Deployment server operational at all times, ensure that you observe the following best practices:

- Tivoli Provisioning Manager for OS Deployment is not a system backup software. Use the server only to store golden images for initial deployment and not individual backup images. The size of the shared file repository used to store images is not expected to be more that 50 GB. If it is larger, then you are not using the product as intended.

- Save each deployment object created on the server, for example, system profiles, software packages, and so on, in a .RAD export file for backup purposes, and store each deployment object on a separate storage device.

- As for all critical applications, the administrator must perform periodic backups of the complete server data directory. The backup is essential to recover from a disk failure or from any other data corruption problem.

- On a regular basis, stop the Tivoli Provisioning Manager for OS Deployment server and perform maintenance on the shared repository, using the `rembo`...
command-line option -chkshared and then -statshared. If statshared reveals a high volume of orphans (unused) files, they can be cleaned up using the -sweepshared option. For information about the usage and syntax for the rembo command, see the set of command-line interface topics in the Reference section.

Note: Use the -packshared option only when disk space must be recovered for other applications because it truncates the shared repository file to the smallest possible size. This might lead to more fragmentation of the shared repository in the future thereby decreasing performance.
Notices

All statements regarding IBM’s future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

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

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

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

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.
Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.
COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM’s application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information in softcopy form, the photographs and color illustrations may not display.

**Trademarks**

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at [Copyright and trademark information](http://www.ibm.com/legal/copytrade.shtml)

Adobe is either a registered trademark or trademark of Adobe Systems Incorporated in the United States, other countries, or both.

Intel, Intel logo, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

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

UNIX is a registered trademark of The Open Group in the United States, other countries, or both.

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