DRIVER AUTOMATION TOOL

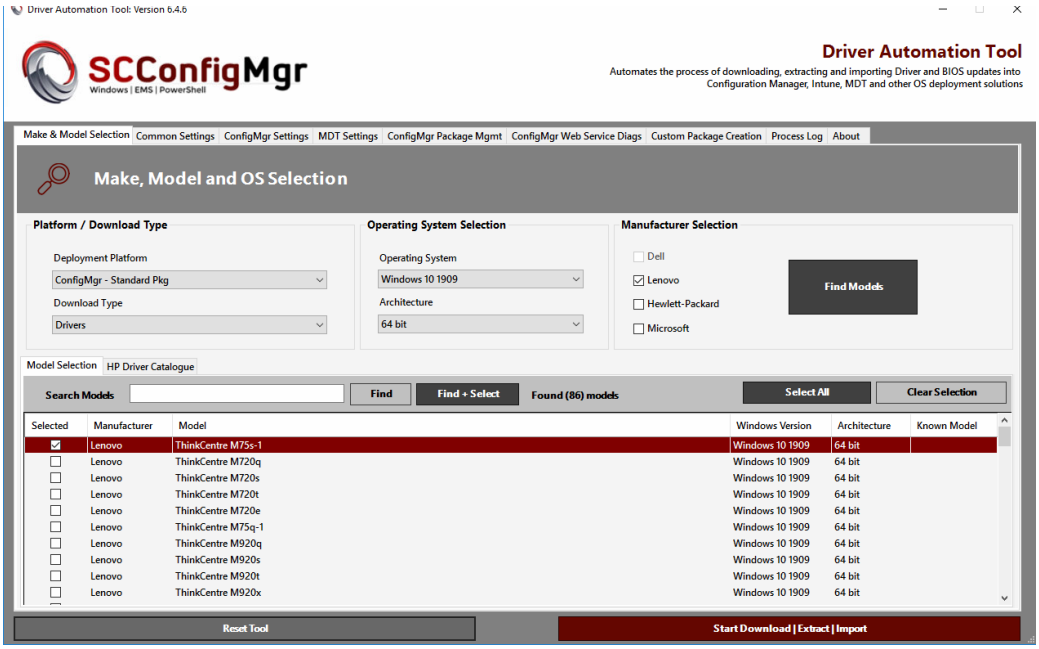


TABLE OF CONTENTS

1 INTRODUCTION

The Driver Automation Tool is an open source PowerShell script which uses WinForms to render an intuitive graphical user interface for downloading driver and BIOS packages into Systems Center Configuration Manager, MDT and other WinPE based disk imaging platforms.

The script uses source feeds from many vendors including Dell, HP, Lenovo and Microsoft to present a list of their client systems. Once model selection has taken place and the OS has been picked, the script automates the following processes:

- Driver/BIOS file download(s)
- Driver/BIOS file extraction
- Packaging of the extracted files
- Importation of the package into Configuration Manager / MDT
- Distribution of the package to distribution points (Configuration Manager)

2 PREREQUISITES

The following prerequisites are required for successful running of the Driver Automation Tool;

- PowerShell v5.0 or greater
- Internet access
- Configuration Manager Requirements;
    - Security rights to the Systems Center Configuration Manager environment
    - Remote access to your SCCM site server
    - SCCM PowerShell module for SCCM downloads
- MDT PowerShell module for MDT downloads
- Full write access to all storage locations used by the tool
- .Net Internet Explorer components are required. These can be installed by installing the Visual Studio 2015 isolated shell, downloadable from - https://visualstudio.microsoft.com/vs/older-downloads/isolated-shell/

Where Visual Studio 2015 isolated shell is not installed, the downloads site will be launched in Internet Explorer automatically. Simply download and install the isolated shell prior to continuing.

Note: For Microsoft Surface known model support, please refer to Appenix A as additional hardware classes are required.

3 RUNNING THE TOOL

Simply install and launch the tool from the Start Menu, alternatively launch the DriverAutomationTool.ps1 script directly from the Source Code folder in %ProgramFiles%\SCConfigMgr\Source Code in an administrative PowerShell console.

New in version 6.2.0 is the ability to launch the tool either as an application with no PowerShell feedback, or as a verbose output version. You will see these options within the Start Menu folder created during installation.

Note: When launching the script from the source code directory, it should be run from an elevated PowerShell window using the -ExecutionPolicy Bypass switch to avoid security warnings, example;

PowerShell.exe -ExecutionPolicy Bypass -File C:\Tools\DriverAutomationTool.ps1

3.1 COMMAND LINE SWITCHES

When running the script, the following command line options are available:

- **-NoXMLOutput**
  Variable type: Boolean
  This option allows you to skip the XML settings export process for ad-hoc operations

- **-RunSilent**
  Variable type: Boolean
  This option allows for a once off silent running of the script without scheduling

Note: The Run-DriverAutomationToolSVC.ps1 file requires configuration output from the GUI so it should not be run directly
.

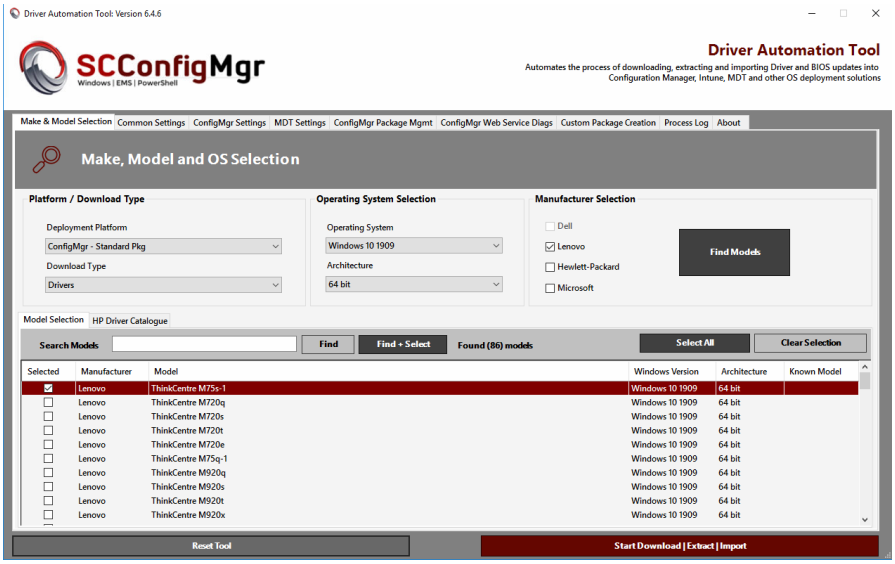
4 MIGRATING SETTINGS

If you have previously run the tool, or if you wish to copy your settings from one system to another, simply copy the entire “Settings” sub folder to the installed location. Then launch the tool and you should have your previous selections included.

6 GUI MODE

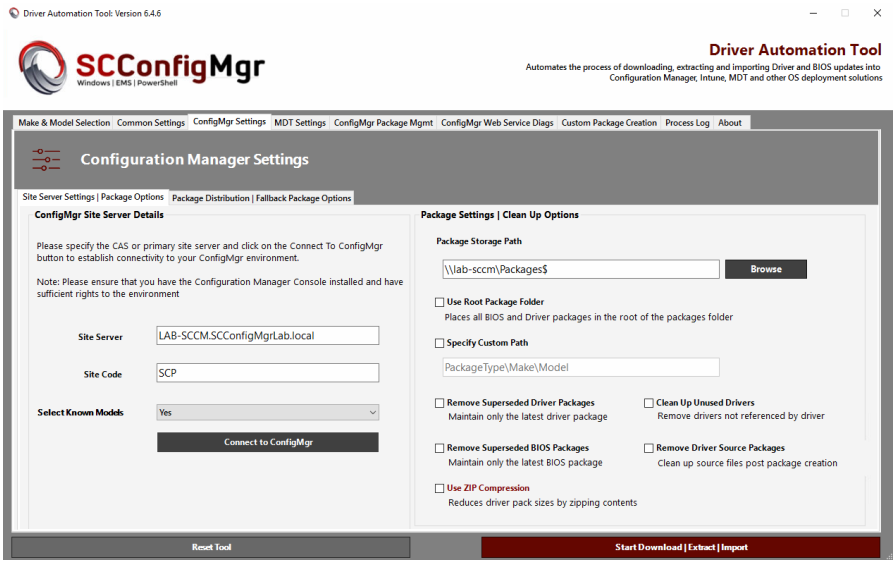
By default, when you run the DriverAutomationTool.ps1 script it will launch in full GUI mode. This is where the process of connecting to your SCCM environment begins, except for using the tool for MDT/download only.

On the initial launch, you will have a GUI like the one pictured below.



6.1 CONNECTING TO YOUR SCCM ENVIRONMENT

To connect to your SCCM environment you must specify details on the ConfigMgr Settings tab.



The name of your site server in the site server text box and click on the “Connect to ConfigMgr” button. When you do so, several processes will take place in the background to ensure that you have access to the required PS cmdlets and the site server, while also attempting to discover the site code.

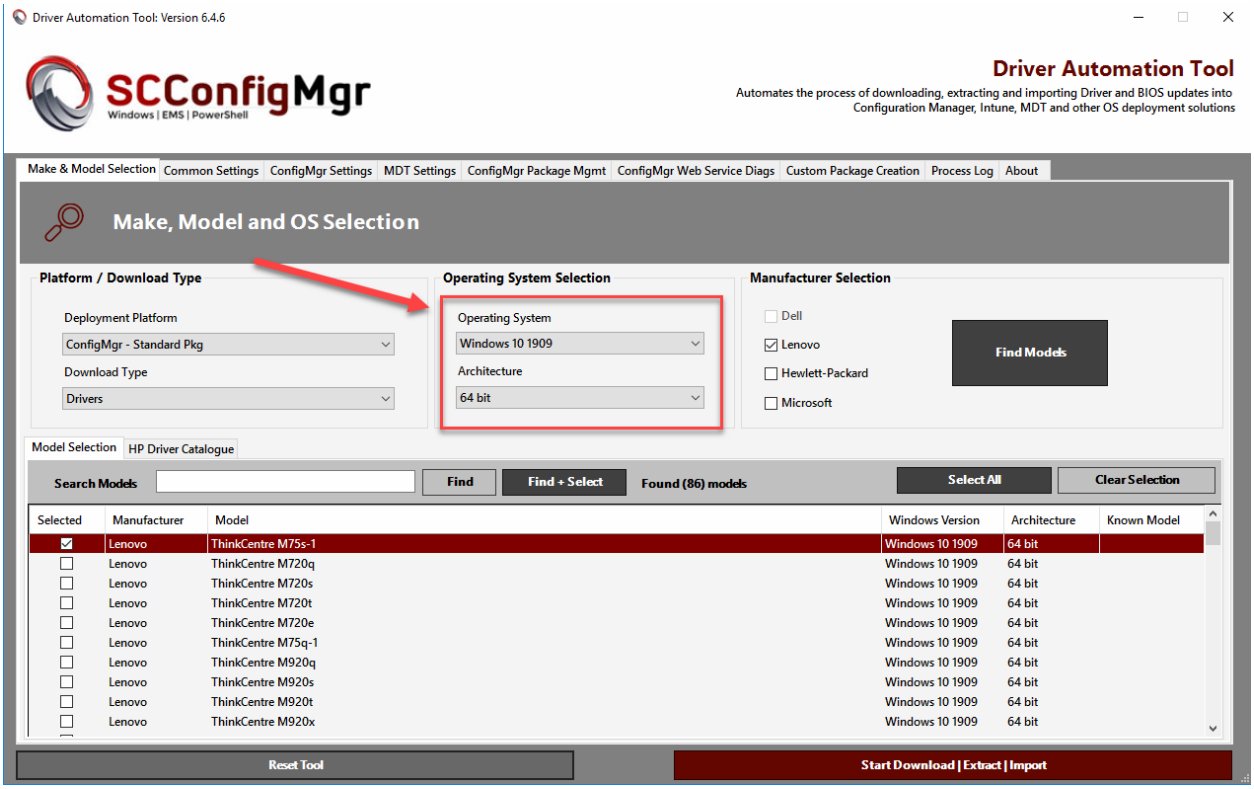
6.2 DEPLOYMENT PLATFORM

After connecting to your SCCM environment or alternatively if you are just using this tool for MDT or XML based matching, you can progress to making selections for the import process, such as the type of download and the OS which to match model listings against.

- **Deployment Platform**
    - **ConfigMgr – Driver Pkg**
      In this mode driver imports will use the Driver Package method, with each of the INF’s being individually imported and presented in the GUI
    - **ConfigMgr – Standard Pkg**
      In this mode driver imports will use the standard program type package method. This method can then be used with our Modern Driver Management process and our Web Service for dynamic deployment of drivers
    - **ConfigMgr – Standard Pkg (Pilot)**
      In this mode it provides the same functionality as the Standard Pkg option but names the packages with a “Pilot” naming scheme. The package can then be used for testing purposes before moving out to production
    - **MDT**
      Used for MDT import jobs
    - **Both – SCCM Driver / Standard Pkg**
      Used for imports both into SCCM and MDT
    - **Download Only**
      Drivers will be downloaded but not imported
    - **Download & XML Model Generation**
      The same as the download option with the exception that an XML list is created containing details of all models contained within the download path. This option can be used for third party disk imaging systems such as Novell ZenWorks

6.3 OPERATING SYSTEM SELECTION

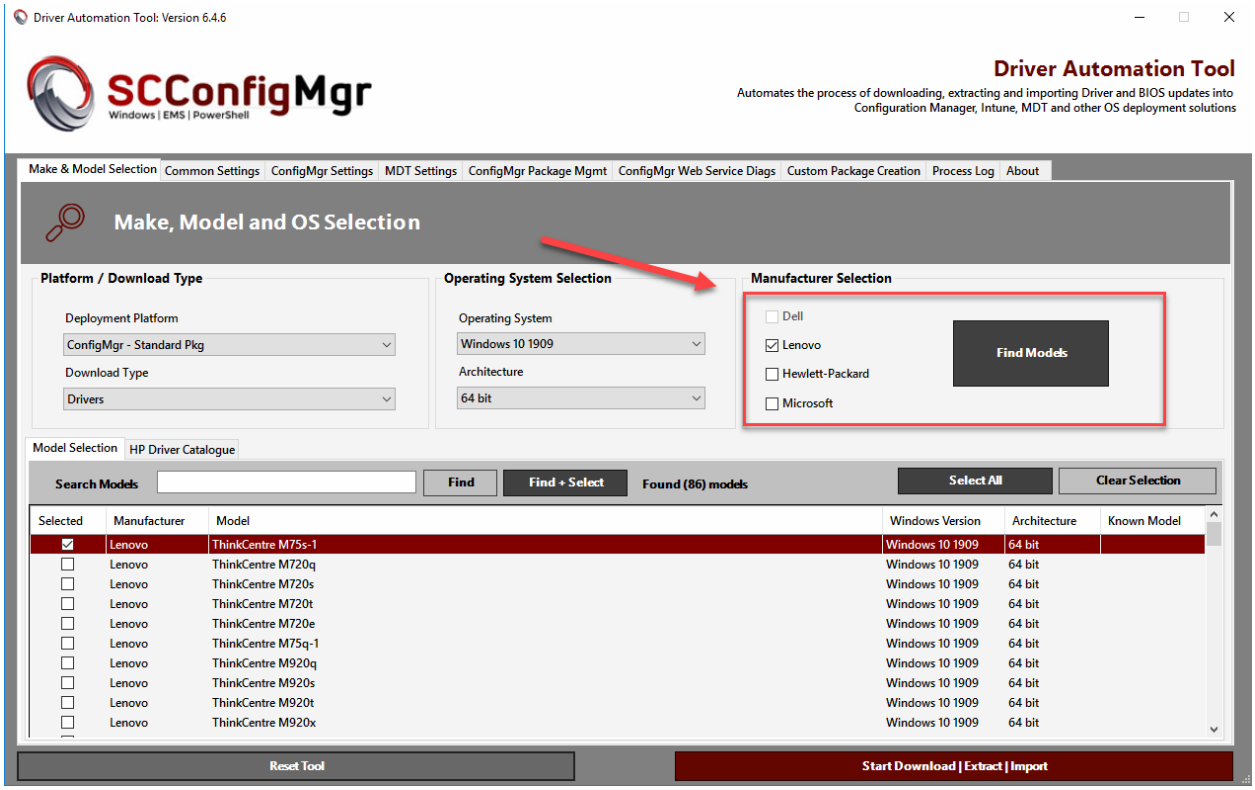
You must select an operating system and architecture for the script to run in either normal or silent mode.



NEW: In version 6.1.0 is the availability of Windows 10 version specific driver packages from Lenovo. Given the addition of Lenovo, packages have been updated to use the Windows 10 version as opposed the build number. Please review section 4.9 of this document if you have driver packages created for HP using the old build number method.

6.4 MANUFACTURER SELECTION

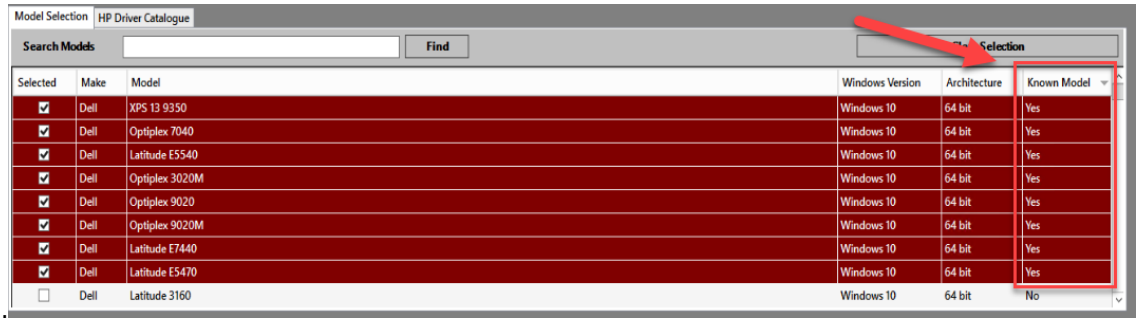
The manufacturer selection is dynamic and based upon OS support from each of the supported vendors. For example, HP is the only manufacturer supporting Windows 10 build numbers within their XML feed, hence HP will be greyed out if the Operating System selected is “Windows 10”, and all other vendors will be greyed out where “Windows 10 xxxx” is selected as the Operating System.



Simply select the vendor you wish to download files from and click on the “Find Models” button for a full list of models matching support on the Operating System selected, alternatively click “Find + Select” to also select these models..

ConfigMgr Import

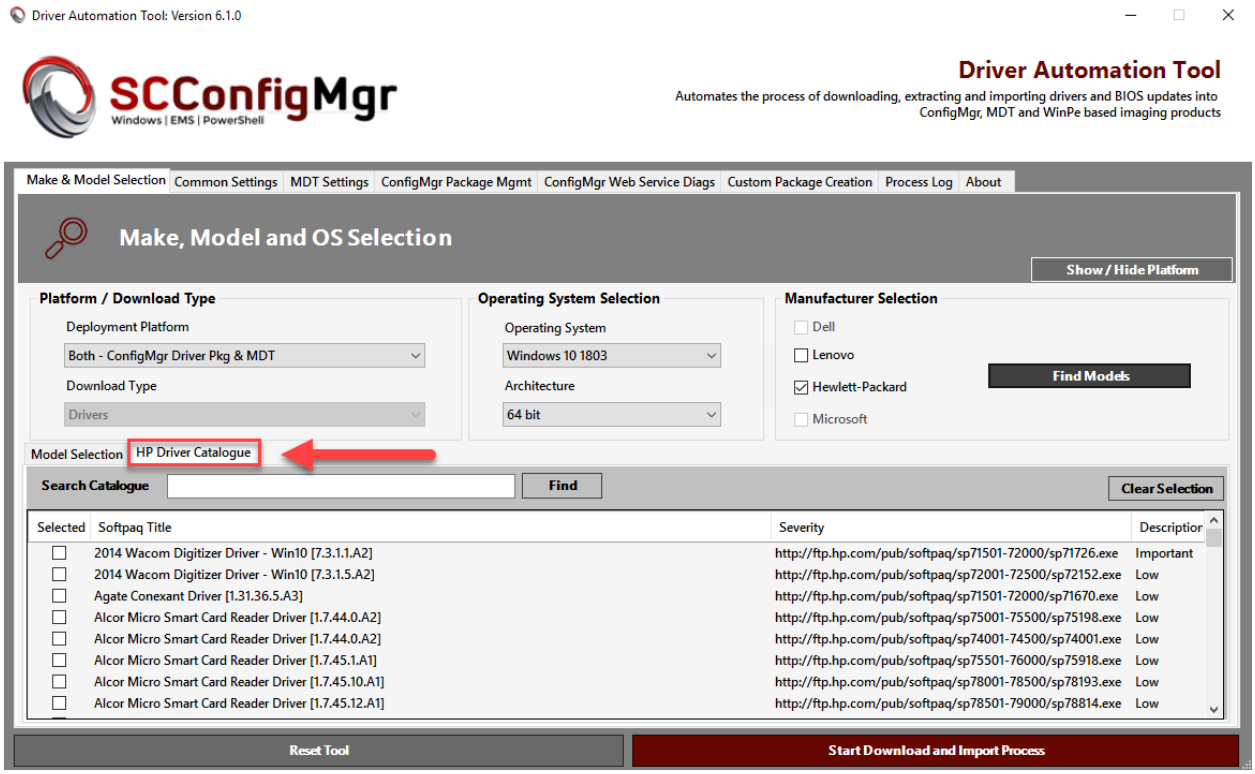
When using the tool with ConfigMgr, you have the option to allow the tool to automatically import Dell, HP and Lenovo models known in WMI. This option can be turned off and on to prevent your selected models list re-populating.



6.5 HP DRIVER SELECTION

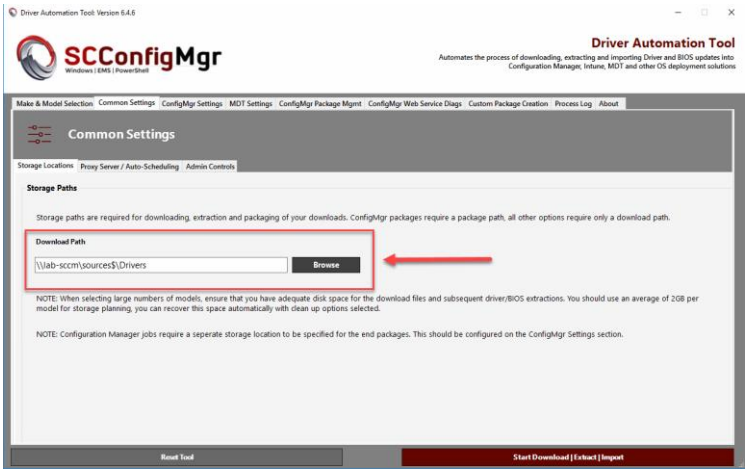
New in version 6.0.0 is the ability to select individual HP drivers based on the OS you have selected. The drivers are displayed in a new tab called “HP Driver Catalogue” and will only display if Hewlett-Packard is selected from the manufacturer list.
The list like the model selection is now also searchable. Enter a keyword and click on the Find button. The matching entries will be moved to the top of the list and a text confirmation of the number of items will appear.

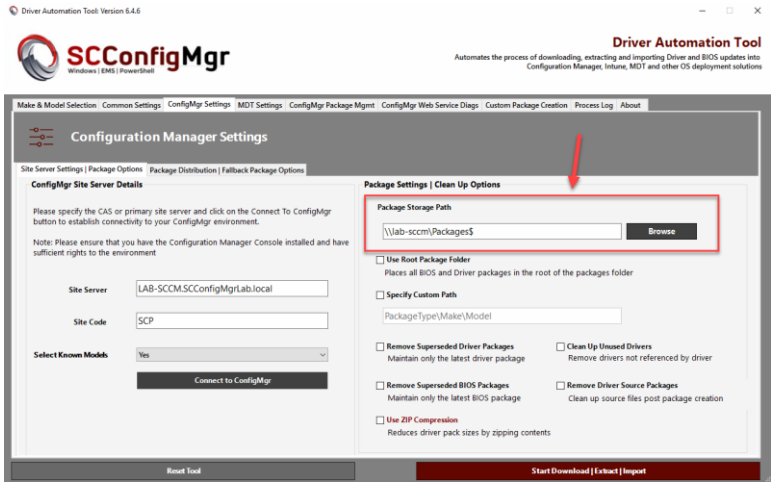
Example of HP Drivers found using key word “Hotkey”:



6.6 DRIVER STORAGE SELECTION

Storage paths are dependent on the deployment platform selected in the tool. They can be found on the Common Settings (Download Path) and on the ConfigMgr Settings tab (Package Path) ;



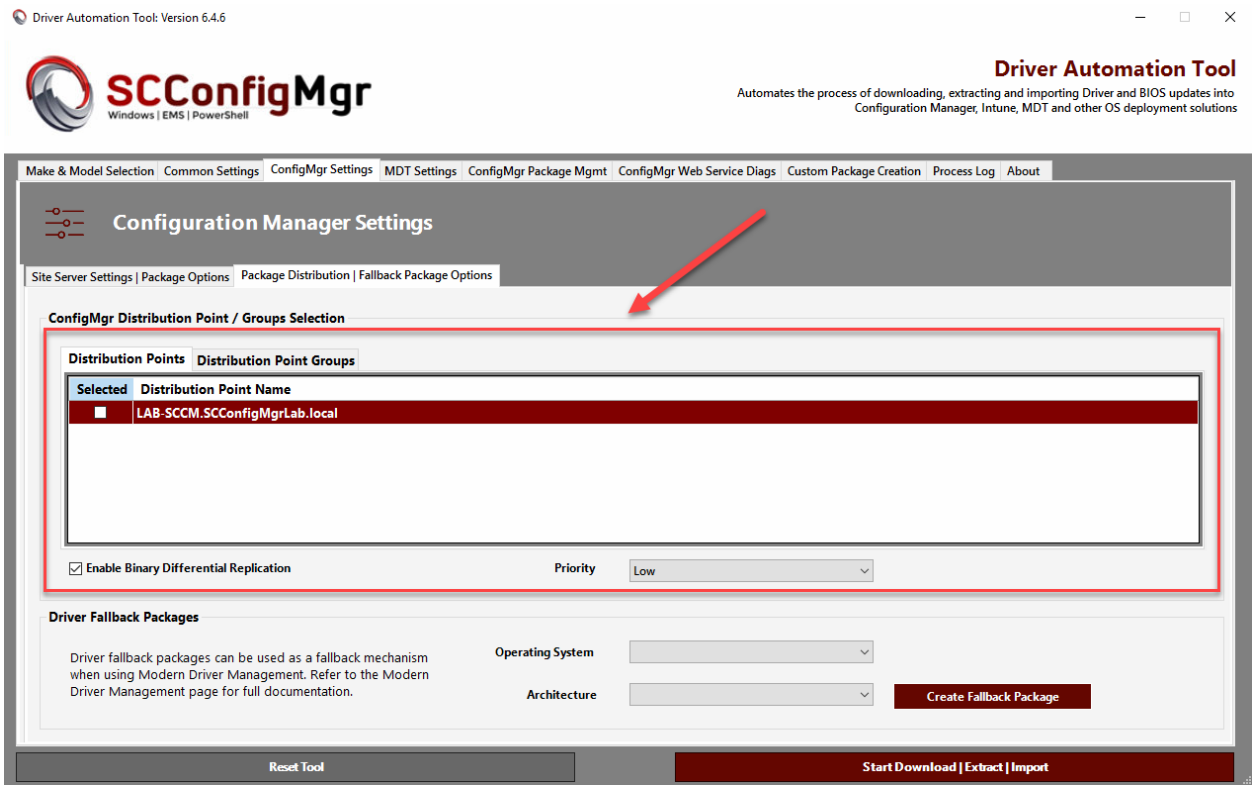


- **Download Path**
  This path is used for temporary storage of packages and driver download cabs / extracted drivers
- **Package Path**
  This path is used for the storage of BIOS and driver packages post extraction

Important
Both the Repository and Package paths should be specified and in different share locations. There is no need to create subfolders within the UNC share for the manufacturers or models as this is undertaken by the script at run time.

6.7 DISTRIBUTION

Selection of individual SCCM distribution points or distribution point groups is available on the ConfigMgr Settings / Package Distribution | Fallback Package tab.

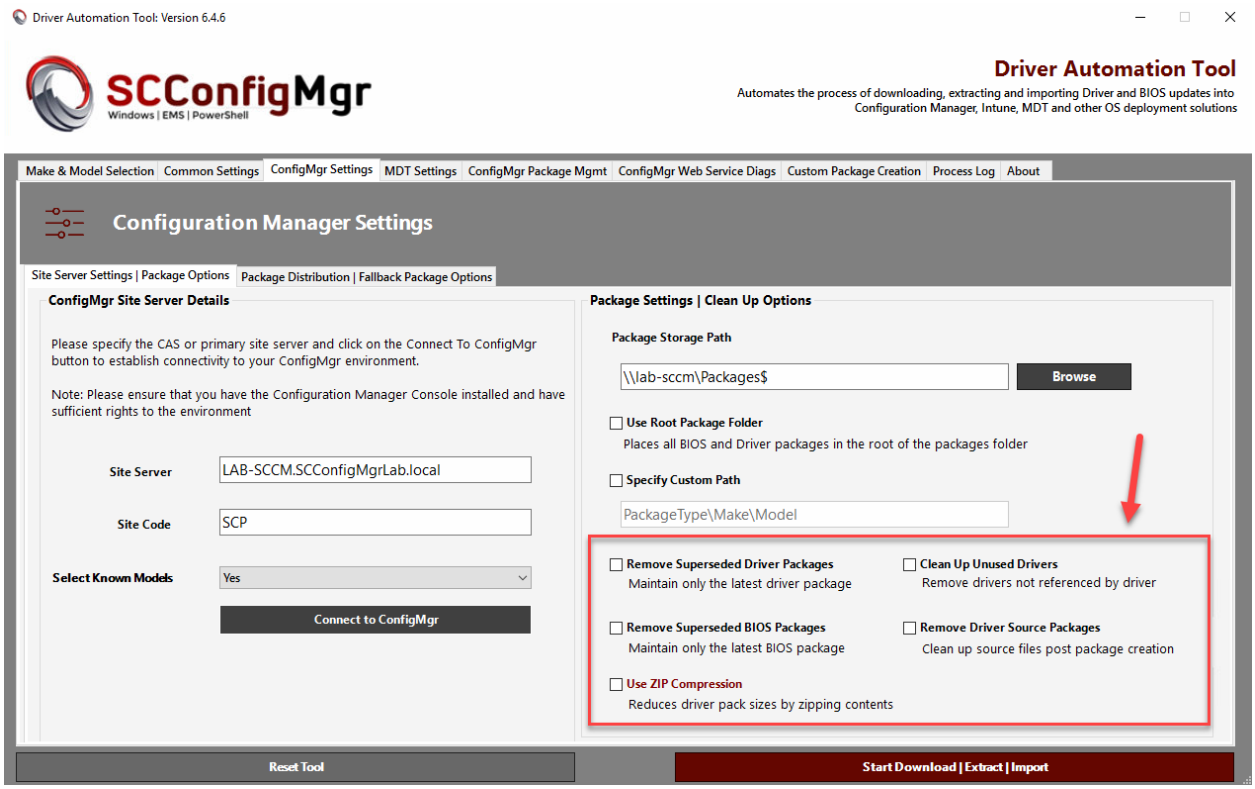


Binary differential replication is enabled by default, thus enabling you to change individual driver package contents and re-distribute the package with the minimal amount of network traffic.

A use case scenario would be where a driver was causing issues in your environment and no updated driver package was available from the OEM. In this instance simple browse to the source of the package, find the offending driver, remove the files / folder and add in a stable / new driver. Redistribute the package and you now have a stable driver set.

6.8 DRIVER CLEAN UP OPTIONS

On the Common Settings / Package Options tab you will find various options to clean up content post run time.



It is recommended that the source files are cleaned up as part of your storage maintenance. Downloading driver packages can have a significant impact on the amount of available storage, especially in environments with a diverse number of makes and models.

To facilitate automated clean up on each run of the tool, there are a number of options, including.

1. **Clean Up Unused Drivers**
   Used for ConfigMgr driver packages, particularly useful when moving away from driver packages to legacy packages containing drivers as used in the SCConfigMgr modern driver management solution.

2. **Remove Superseded Driver Packages**
   Removes the previous package version where a newer package exists

3. **Remove Driver Source Packages**
   Removes all files used in the “Download” path used to download and extract drivers prior to packaging.

6.9 PACKAGE MANAGEMENT

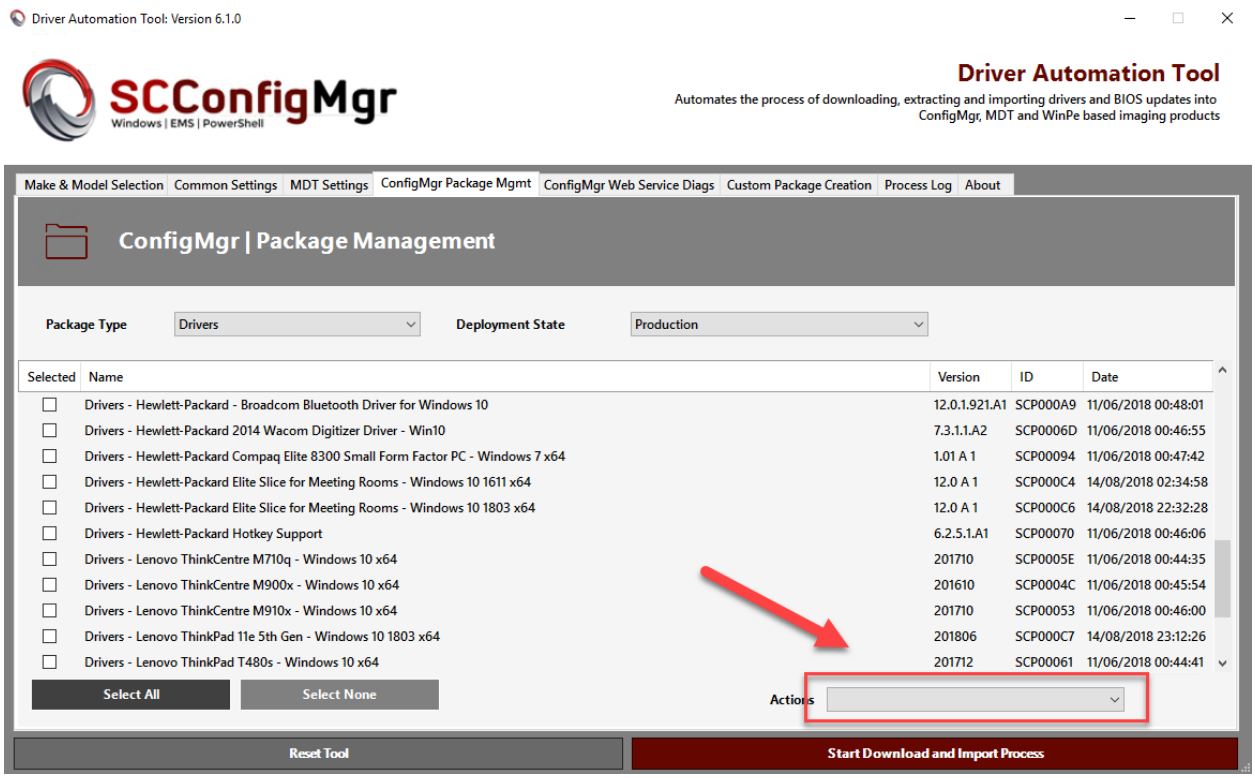
The package management section allows you to quickly move packages between pre-production (pilot), production and retired states. New in 6.1.0 is the ability to also move packages between Windows 10 versions.

- **Production**
  The production state is self-explanatory, here you will find packages which you have deemed suitable for mass deployment in your environment

- **Pilot**
  The pilot state if for pre-production testing. Combined with the filter option on our MDM solution you can specify “Drivers Pilot” or “BIOS Update Pilot” to return packages in this state for test purposes

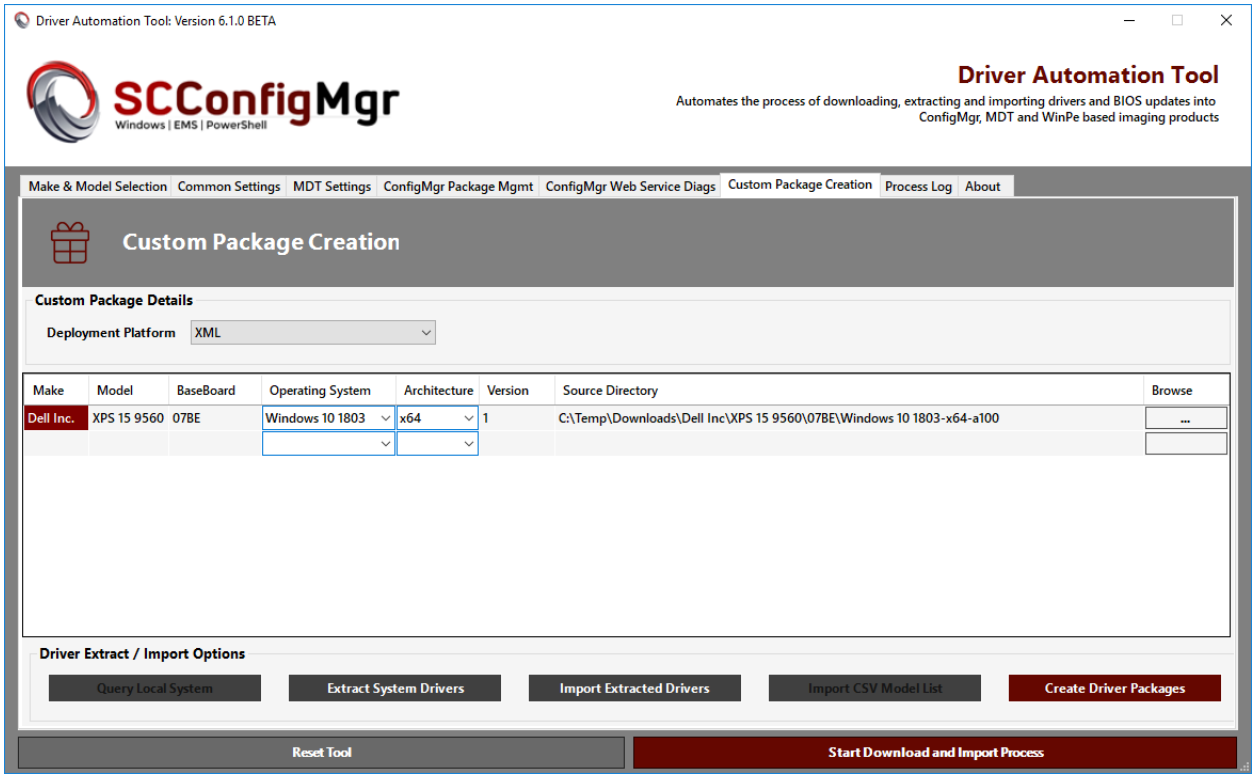
- **Retire**
  The retire state simply allows you to flag packages which are retired and can be removed by your ConfigMgr admin, or to make it easier to bulk remove these packages via PowerShell

- **NEW: Move to Windows 10 xxxx**
  This option was introduced to provide a quick means of converting pre 6.1.0 packages that use the Windows build number to the new Windows version number format



Packages can be moved between any of these states by selecting the packages and using the “Actions” drop down list.

6.10 CUSTOM PACKAGES



New to 5.0.1 is the ability to create driver packages for any make or model, allowing for use with our modern driver management solution.

To create a package, you will first of all need to download and extract the vendor drivers. After you have extracted the drivers you will need to obtain the baseboard product value from WMI, this will be used as the unique identifier to match packages.

To obtain the baseboard product value, you can use the following single line of PowerShell code:

(Get-CIMInstance -ClassName MS_SystemInformation -NameSpace root\WMI).BaseBoardProduct

Example *– Fujitsu Celsius H730 -* ***FJNB274***

Alternatively, you could also report on these values in ConfigMgr by extending the hardware collection values. To do so follow the process in Appendix A at the end of this document.

Connecting to ConfigMgr / MDT Environments

As the import process allows for a mix of MDT and ConfigMgr imports, you must connect to these environments using the ConfigMgr Settings and MDT Settings tabs prior to pressing the “Create Driver Packages” button.

Adding / Importing Custom Makes / Models

There are two options when adding custom makes/models, you can either input all of the details manually or you can use a CSV file and import the contents. Included in the ZIP download is a blank CSV for editing.

Field Values

Below is a list of the value types accepted:

FIELD NAME	*TYPE*	*OPTIONS*
MAKE	Text	
MODEL	Text	
BASEBOARD	Text	
PLATFORM	List – Single Select	ConfigMgr MDT
OPERATING SYSTEM	List – Single Select	Windows 10 1709 Windows 10 1703 Windows 10 1610 Windows 10 Windows 8.1 Windows 8 Windows 7
ARCHITECTURE	List – Single Select	x86 x64
VERSION	Text	
SOURCE DIRECTORY	Text	

CSV Format Example

Make,Model,BaseBoard,Platform,Operating System,Architecture,Version,Source Directory

Toshiba,PORTEGE Z10T-A,PT141E,ConfigMgr,Windows 10,x64,1,E:\Toshiba\Portege Z10T-A\Win1064

Creating Driver Packages

Once you have filled in all of the required values or imported your completed CSV, clicking the **Create Driver Packages** will start the process of creating ConfigMgr packages or importing drivers into MDT.

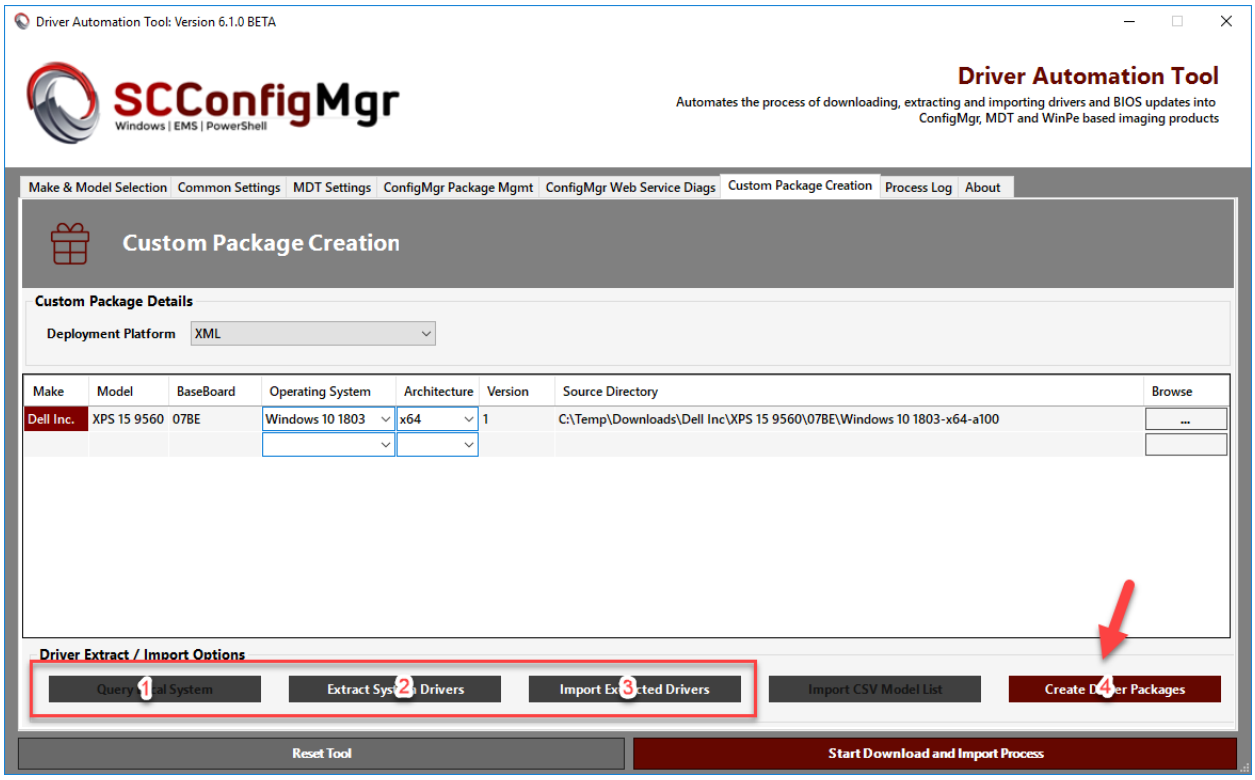
If ConfigMgr is the selected platform packages will also distribute according to the selected distribution point(s) / distribution point group(s) on the Common Settings tab.

6.11 LOCAL SYSTEM DRIVER PACKAGES

New in 6.1.0 is the ability to run the Driver Automation Tool on a end user machine, query the local system to retrieve matching variable information and export all of the currently installed drivers to create a “OEM” style package.

The idea is if you have a machine which does not officially have a driver package available, you can run the tool on the OEM shipped image, export the drivers to a UNC share and then run the tool to import the files and model details. Using this method you will create a package that contains all of the machine variables* for the machine.

* Note that baseboard variable is subject to manufacturer and might need to be manually set



To export local drivers, simple use the following four step process having selected your desired deployment platform;

1. Launch the Driver Automation Tool on a source system and go to the Custom Package Creation tab. Click on the Query Local System (1) button and details will be obtained from the local machine. A source directory will be set where the drivers will be extracted to, in order to be imported later.

   NOTE: The version number should be updated to provide a value to compare against at run time.

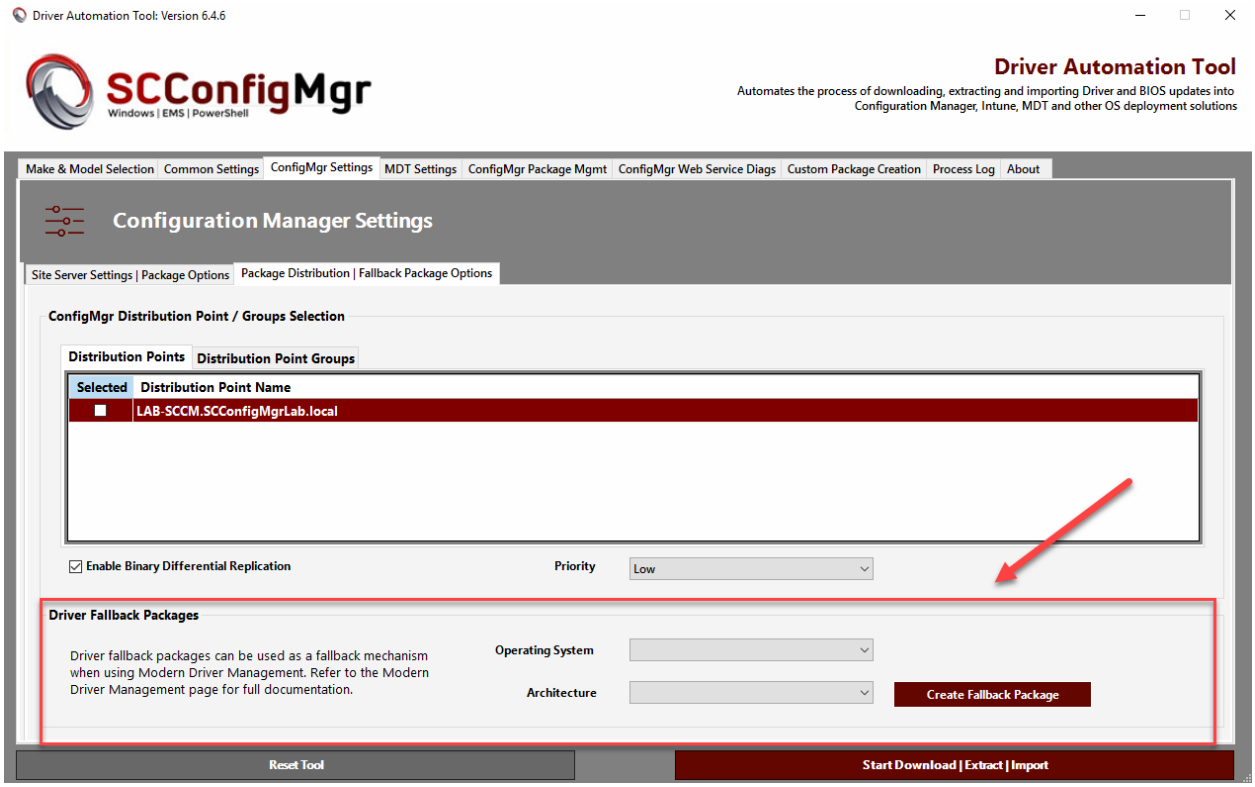
2. Click on the Extract System Drivers (2). The process of driver extraction will begin, it is recommended to use a UNC path on the target server

3. Launch the Driver Automation Tool on your ConfigMgr / MDT server and click on the Import Extracted Drivers (3) button. You will now need to browse to select the XML details file created in your “Source Directory”

4. Click on the Create Driver Packages (4) button and your custom driver package will be created in ConfigMgr / imported into MDT

6.12 DRIVER FALLBACK PACKAGES

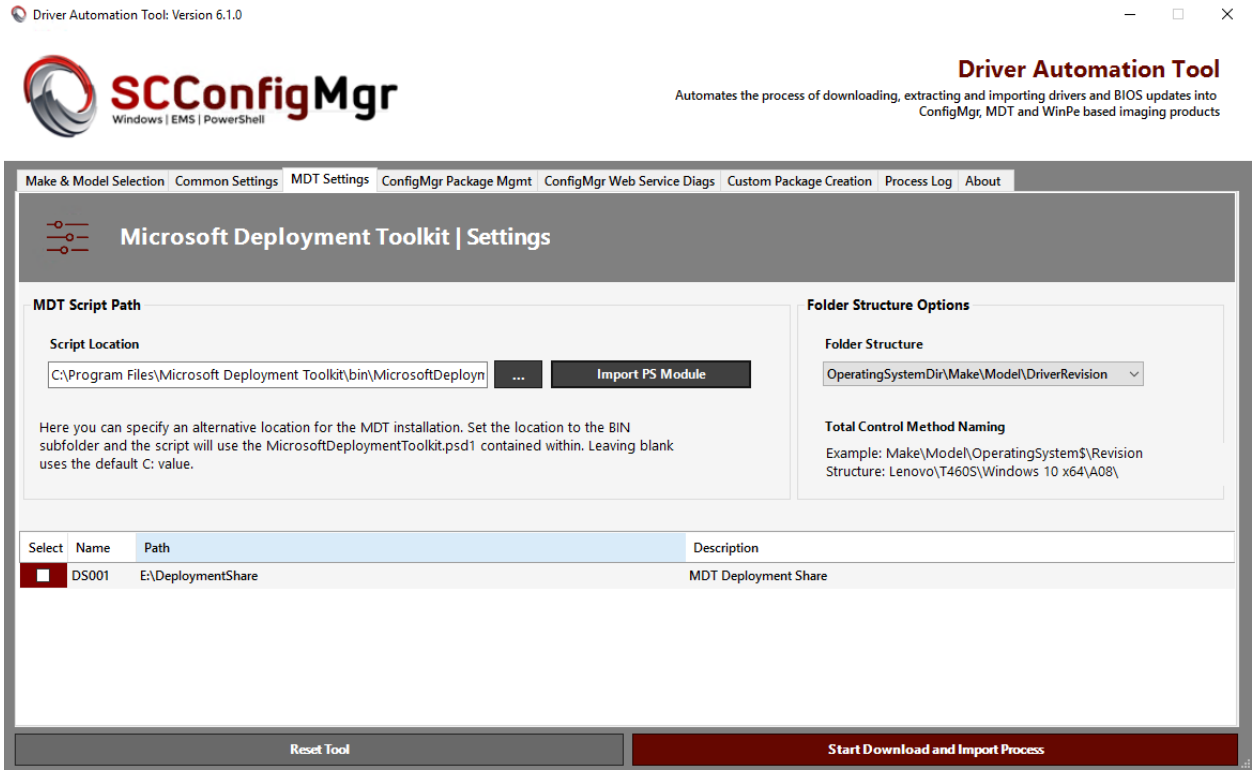
Driver fallback packages are intended for generic ad-hoc deployments where a number of driver files can be placed into a fallback package and called at deployment time.



Packages are created on a per operating system basis, based on the OS version and architecture. Drivers should then be extracted and dropped into the package source location.

6.13 MDT SETTINGS

Added in 5.0.0 is the ability to specify an alternative location for your MDT installation without having to edit the script source. Simply browse or paste in the past and click the “Import PS Module” button to verify the PowerShell cmdlets are imported.



Other MDT additions include;

- **Deployment Share Selection**

  Now you can select single or multiple deployment shares to distribute content to during the download and import process

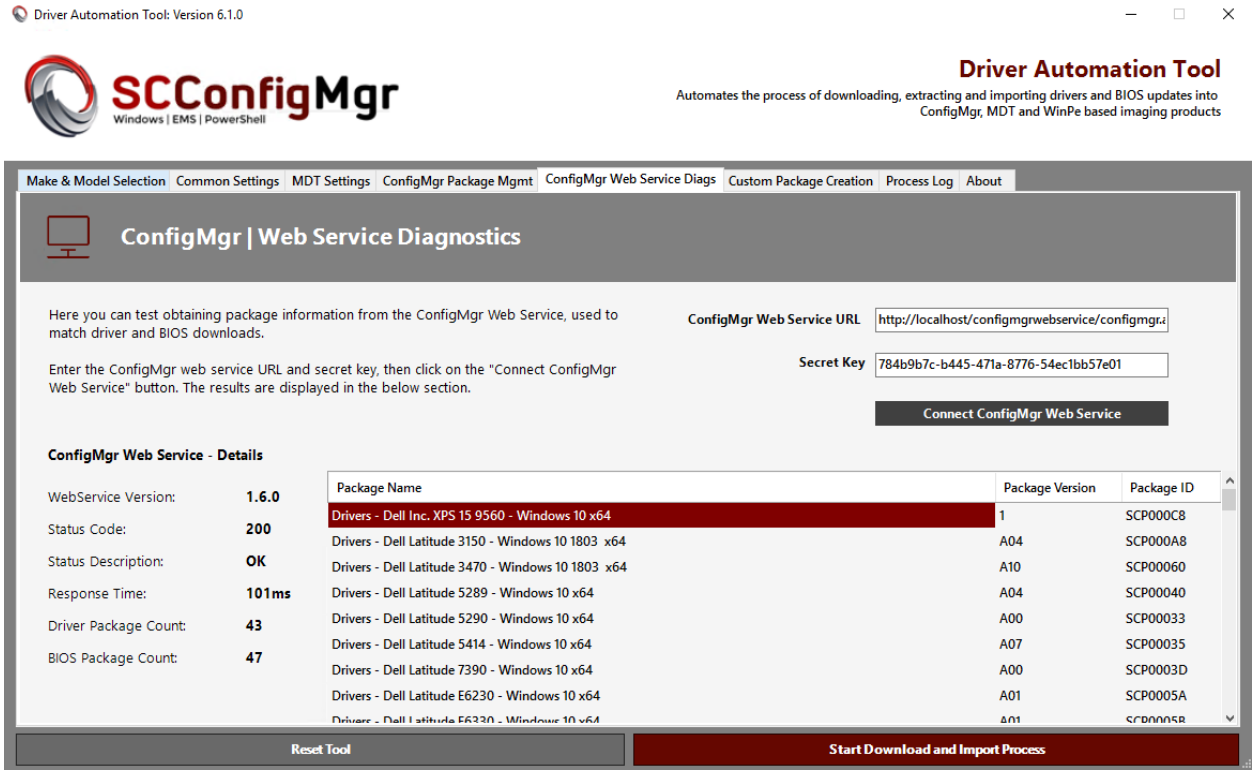
- **Driver Structure**

  Here you can select different folder naming structures for your MDT total control method. At present two are included;

  OperatingSystemDir\Make\Model\DriverRevision
  Make\Model\OperatingSystemDir\DriverRevision

7 MODERN DRIVER/BIOS MANAGEMENT DIAGNOSTICS

For those of you who are using the full Modern Driver/BIOS Management solution, in version 5.1.1 a new tab has been added to provide quick diagnostic information when querying the ConfigMgr WebService.



On the “ConfigMgr Web Service Diags” tab simply enter the ConfigMgr WebService URL and Secret Key, then press the “Connect ConfigMgr Web Service” button to attempt communications.

Upon successful connection to the ConfigMgr WebService the following details will be displayed;

- **WebService Version**
- **Status Code**
- **Status Description**
- **Response Time**
- **Driver Package Count**
- **BIOS Package Count**
- **Returned Package List**

8 MODEL XML GENERATION

New in version 6.1.0 is the ability to support third party disk imaging solutions, MDT and standalone media with a dynamic solution for matching drivers without the need to talk to a backend web service.

When packages are created using this method, the drivers are extracted, and details of the drivers contained within a master XML file.

Example:

```
<?xml version="1.0"?>
<?xml-stylesheet type='text/xsl' href='style.xsl'?>
<!--Created with the SCConfigMgr Driver Automation Tool-->
<Details current="True">
  <ModelDetails>
    <Make>Dell Inc.</Make>
    <Model>XPS 15 9560</Model>
    <SystemSKU>07BE</SystemSKU>
    <OperatingSystem>XML</OperatingSystem>
    <Architecture>Windows 10 1803</Architecture>
    <Platform>XML</Platform>
  </ModelDetails>
  <ModelDetails>
    <Make>Hewlett-Packard</Make>
    <Model>Elite x2 1012 G2 Tablet</Model>
    <SystemSKU>82ca</SystemSKU>
  </ModelDetails>
  <ModelDetails>
    <Make>Lenovo</Make>
    <Model>ThinkPad X280</Model>
    <SystemSKU>20KE 20KF</SystemSKU>
  </ModelDetails>
</Details>
```

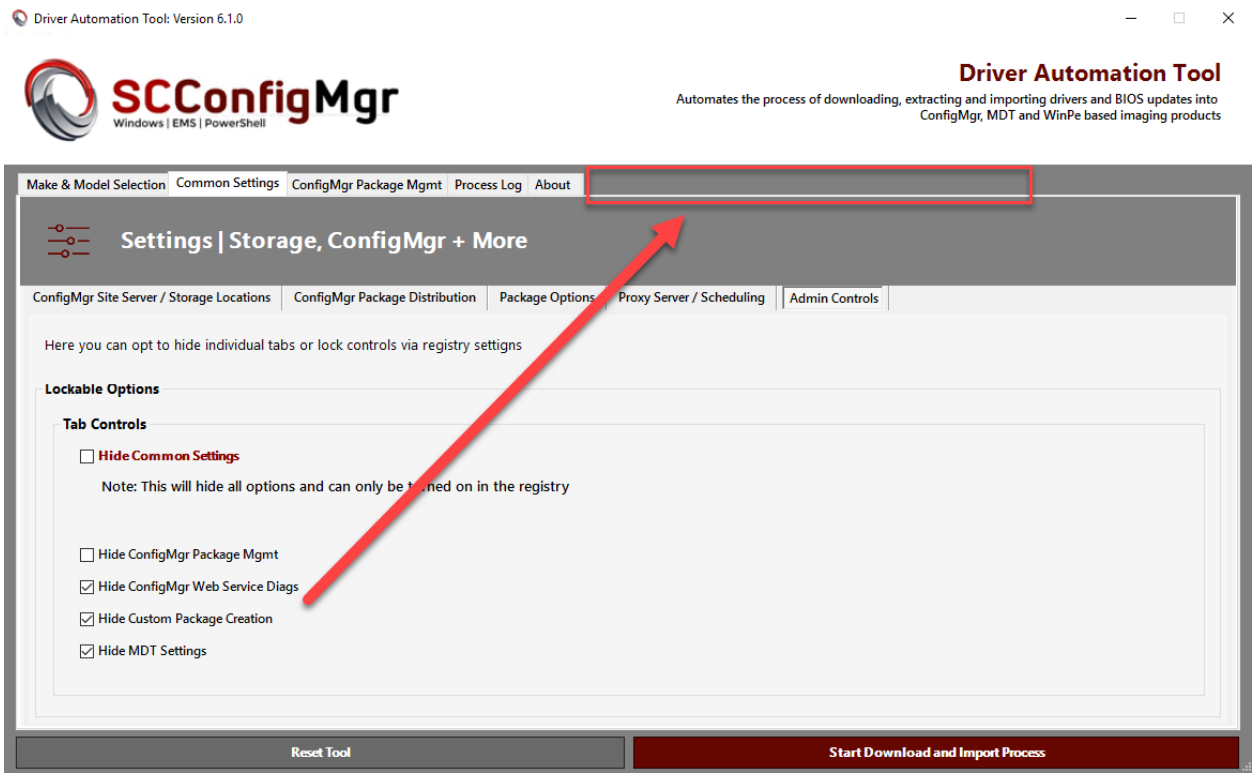
To use the XML method in our modern driver management process, a separate matching script is available on the SCConfigMgr GitHub repository – Invoke-XMLApplyDriver.ps1. The script works by calling a local or UNC storage repository containing the packaged drivers downloaded by the Driver Automation Tool.

Script Syntax Example:

Invoke-CMApplyDriverPackage.ps1 -ModelXL -DeploymentType BareMetal -DriverInstallMode Recurse -StoragePath '\\SERVER\DriverDownloads' -OS 'Windows 10' -OSArchitecture x64

9 ADMIN CONTROLS

New in version 6.1.0 is the ability to selectively hide tab pages. The use case scenario is where an admin wishes to product access to the tool to junior administrators, however they do not want them to modify settings within the tool.



Settings are maintained within the registry and therefore can be locked down by user or group. Below are the fields that are set by the tool;

Registry Path – HKLM:\Software\SCConfigMgr\DriverAutomationTool

CommonOptionsVisible
ConfigMgrPkgOptionsVisible
ConfigMgrWebSvcVisible
CustomPkgVisible
MDTSettingsVisible

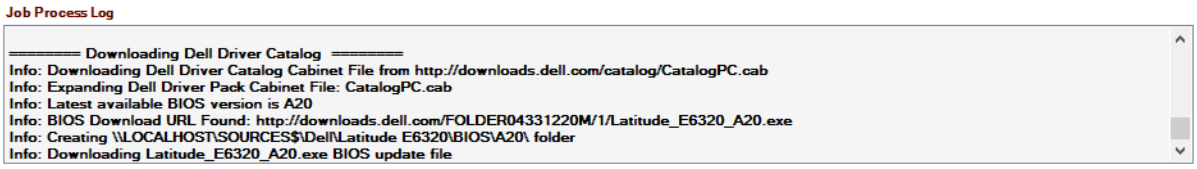
All values are Boolean, therefore setting a value of “1” will ensure that the tab is visible and “0” will hide the tab.

10 NORMAL / SILENT OPERATION

After the initial selection of your models, operating system, import type etc you have the option to either commence the download process by clicking on the **"Start Download and Import Process"** button or opt to schedule the job for silent running.

10.1 NORMAL OPERATION

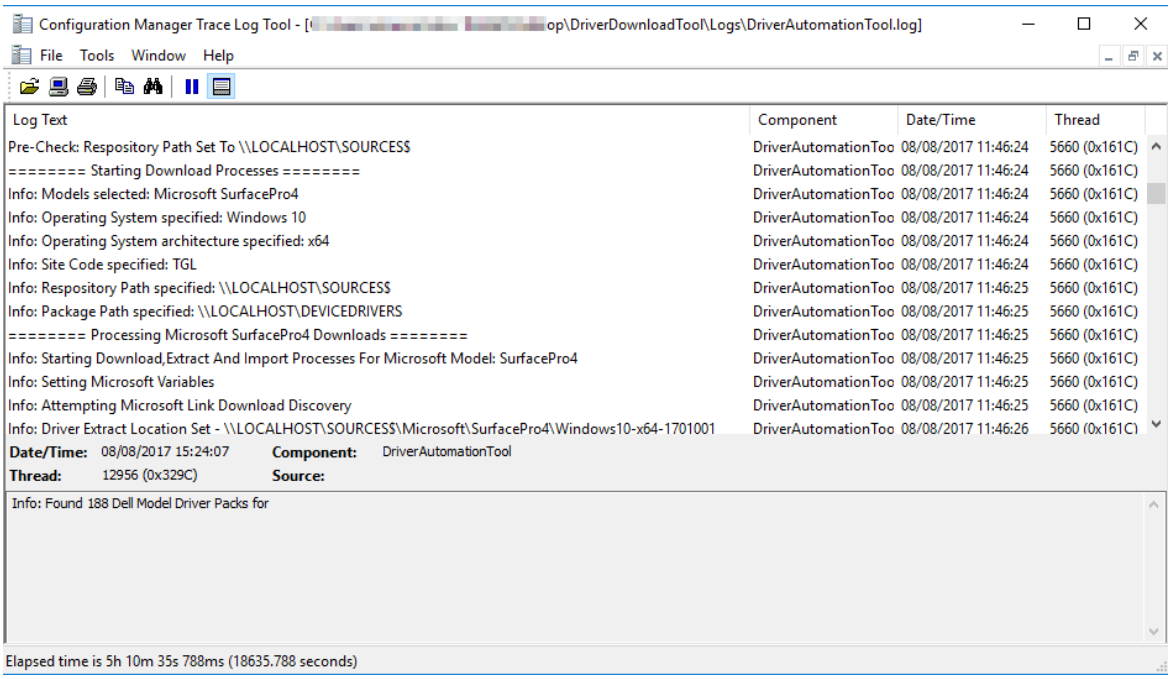
By clicking on the "Start Download and Import Process" button, all output will be passed through to the Job Process Log, keeping you up to date with the various processes running.



Contained inside the folder from which the script is launched you will notice there is now a "Logs" directory. Within you will find a verbose output log file (**DriverAutomationTool.log**) which can view with your preferred log viewer, CMTrace for example.

CMTrace is part of the Systems Center 2012 R2 Configuration Manager Toolkit and downloadable from the following URL - https://www.microsoft.com/en-us/download/details.aspx?id=50012

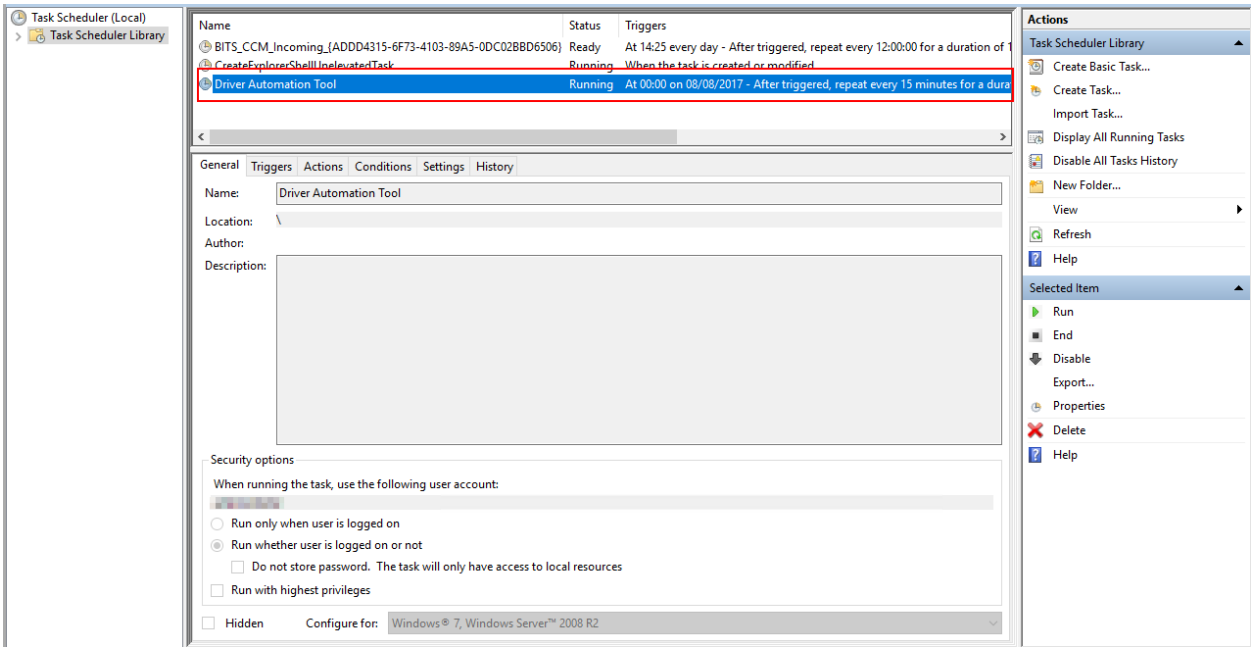
Below is an example of the contents of the log file generated;



10.2 SILENT OPERATION

For silent operation, you will need to specify a location for the script to run from along with a starting time and service account details. The service account specified should have rights to your SCCM environment.

When you click on the Schedule Job button, the script will run a function to validate your credentials against Active Directory. Once the account is valid, it will copy the Run-DriverAutomationToolSvc.ps1 script to the directory specified and set the scheduled start time.



By default, the script will run every 15 minutes, this is to cater for any time outs that occur with XML feeds, downloads etc. You can of course modify this by changing the settings in the task scheduler.

Logging of the operations is provided in a log file located in the Logs subfolder.

APPENDIX A

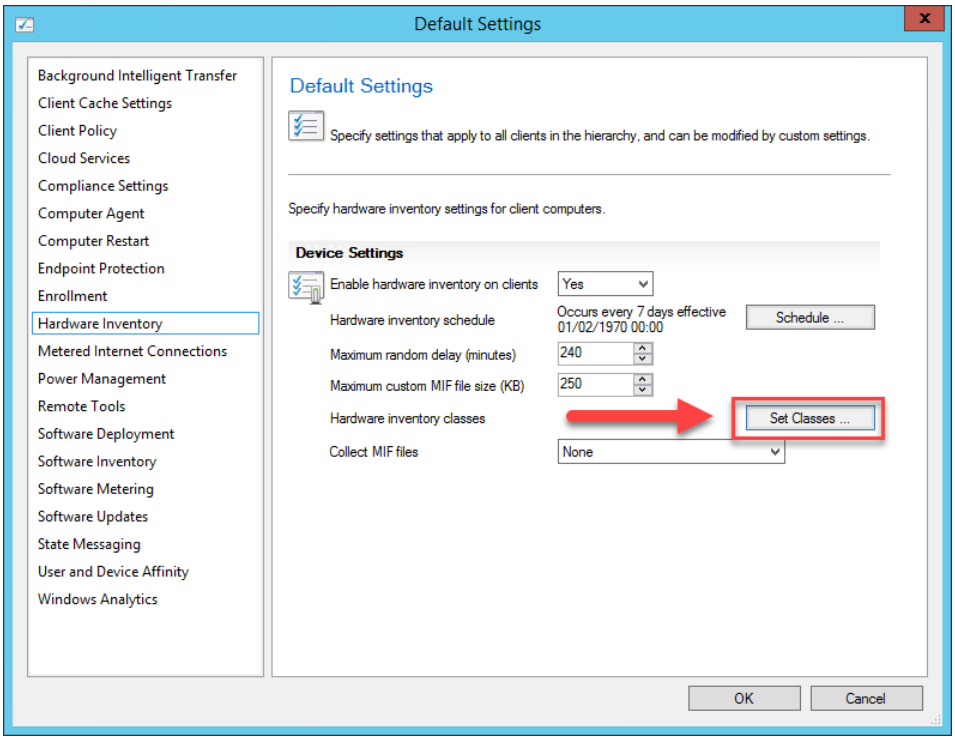
CUSTOM HARDWARE CLASSES

To identify unknown / unsupported manufacturer models when using our Modern Driver Management solution, the baseboard product value is used. This hardware inventory class is not collected by default, so it will need to be added in if you want to run reports on your environment to obtain this class information.

Simply follow the below process to perform the required steps in the Default Client Settings and then apply as required in any custom client settings you have set up.

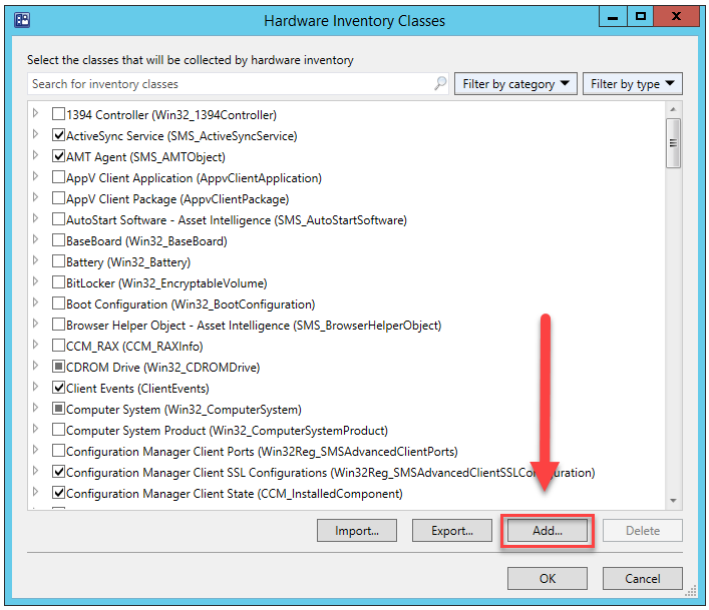
1. DEFAULT CLIENT SETTINGS

Open the client settings in the ConfigMgr console (Administration – Client Settings – Default Settings).
Once open click on the “Hardware Inventory” section and then click on the “Set Classes” button.

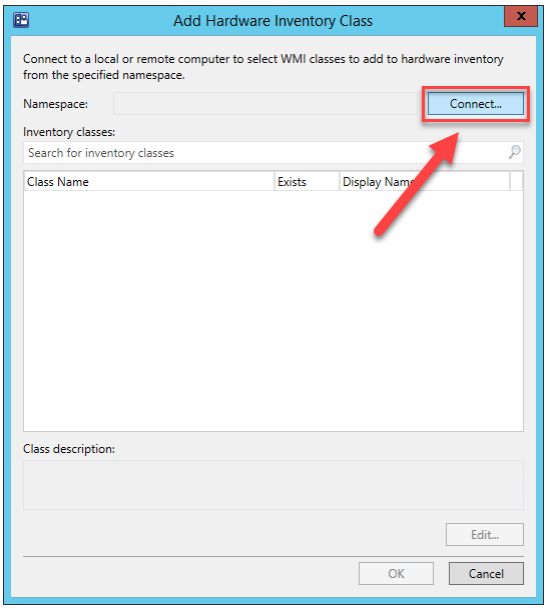


2. SET HARDWARE INVENTORY CLASSES

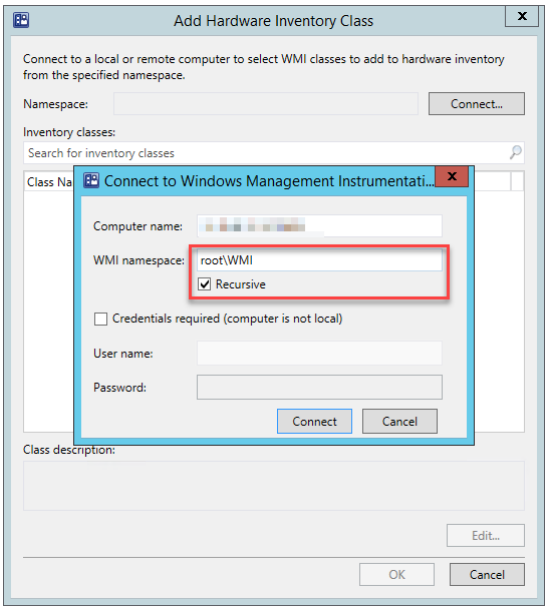
With the Hardware Inventory Classes screen open, click on the "**Add**" button.



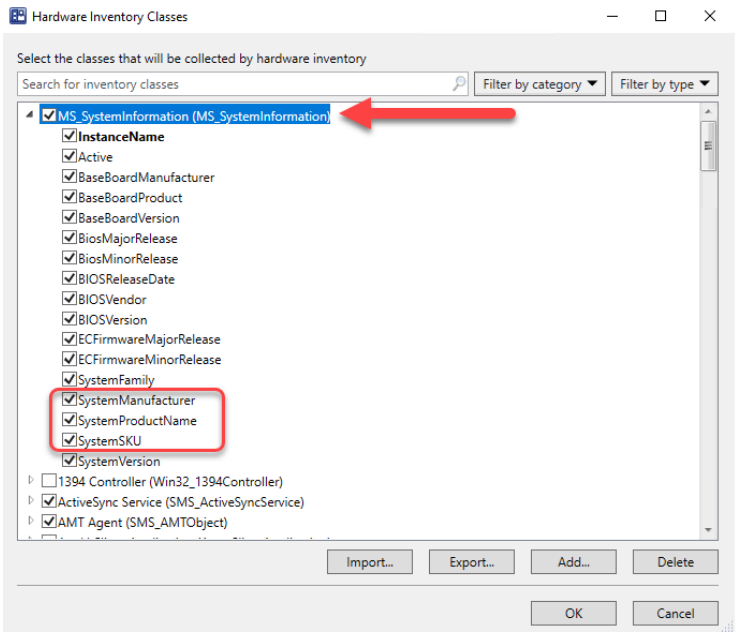
Click on the "**Connect**" button



Enter a client **Computer Name** and **WMI namespace** (Root\WMI), then tick the **Recursive** tickbox before clicking on the **Connect** button



Finally enter **MS_SystemInformation** as the filter and select the following values; “SystemManufacturer, SystemProductName, SystemSKU



DELL BIOS FLASH UTILITY – FLASH64W

For flashing the BIOS in WinPE 64-bit, Dell provide the Flash64w utility. This utility is automatically downloaded and added to BIOS packages, new in version 6.4.0, the tool will check for a new release of the Flash64w on each run. Should it find a legacy version, it will archive the legacy version and create new packages using the latest version;

Log;

```
======== Dell Latitude E5470 BIOS PROCESSING STARTED ========
Info: Reading Dell product XML file - C:\Temp\Temp\CatalogPC.xml
Info: Found BIOS URL FOLDER05863756M/1/Latitude_E5x70_Precision_3510_1.21.6.exe
Info: Latest available BIOS version is 1.21.6
Info: Checking for existing BIOS release - 1.21.6
Info: BIOS Download URL Found: https://downloads.dell.com/FOLDER05863756M/1/Latitude_E5x70_Precision_3510_1.21.6.exe
Info: Creating C:\Temp\Dell\Latitude E5470\BIOS\1-21-6\ folder
Info: Downloading Latitude_E5x70_Precision_3510_1.21.6.exe BIOS update file
Info: Existing Dell Flash 64 EXE found
Info: Unable to obtain version info from legacy Dell Flash 64 EXE
Info: Setting version info to version 1.0 for archiving purposes
Info: Unzipping latest Dell Flash64 EXE
Info: New Dell Flash 64 EXE version is 1.0
Info: Latest Dell Flash 64 EXE is 3.3.1
Info: Creating new/updated Dell Flash 64 source
Info: Archiving legacy file
Info: Promoting 1.0 release to production
Info: Copying Dell Flash64Bit EXE To C:\Temp\Dell\Latitude E5470\BIOS\1-21-6\

======== Dell Latitude E5470 BIOS PROCESSING FINISHED ========
Info: Remaining models to process: 0

======== Clean Up Driver Option Processing ========
Download: Removing driver download and extracted source driver files from C:\Temp

======== FINISHED PROCESSING ========
```

File Structure;