Table of Contents

1. Introduction .................................................................................................................. 5

2. Dell Precision Optimizer Components ........................................................................ 5
   2.1 Uninstalling Dell Precision Optimizer ................................................................. 5

3. Performance .................................................................................................................. 6
   3.1 Policy Processing Engine (PPE) ........................................................................... 6
   3.2 Profile Update Tool ............................................................................................ 7
   3.3 Profile Update Tool ............................................................................................ 7

4. Track and Analyze (TA) ............................................................................................ 7
   4.1 System Analysis Reports ................................................................................... 7
      4.1.1 Report Settings ......................................................................................... 7
   4.2 Workload Analysis ............................................................................................ 8
   4.3 CPU Intelligence Reports ................................................................................ 8
   4.4 Graphics Intelligence Reports .......................................................................... 9
   4.5 System Health Reports ................................................................................... 9
   4.6 Smart Notifications ......................................................................................... 10
   4.7 Upgrade Options ............................................................................................. 10

5. System Maintenance (SM) ....................................................................................... 10

6. User Feedback ......................................................................................................... 11

7. Improve Dell Precision Optimizer ............................................................................ 11

8. Enterprise Tools ....................................................................................................... 11
   8.1 WMI Providers .................................................................................................. 11
   8.2 DPOCMD.EXE .................................................................................................. 12
   8.3 Setup Command Line Switches ........................................................................ 14
   8.4 SCCM ................................................................................................................. 15
      8.4.1 Instructions for creating the Dell Precision Optimizer application package .... 15
      8.4.2 Instructions for Deploying the Application .................................................. 16
      8.4.3 Verifying Deployment Success in Client systems ......................................... 16
      8.4.4 Changing Dell Precision Optimizer Client Behavior using DPOCMD.EXE .... 16
      8.4.5 SSRS Reports ....................................................................................... 17
   8.5 KACE ................................................................................................................ 23
      8.5.1 Instructions for deploying Dell Precision Optimizer using KACE .................. 23
      8.5.2 Changing Dell Precision Optimizer Client Behavior using DPOCMD.EXE .... 24
1. Introduction

This document describes tools available for IT administrators to manage Dell Precision Optimizer remotely. It also contains helpful tips and recommendations to make it easier for administrators to manage Dell Precision Optimizer.

2. Dell Precision Optimizer Components

The four main components of Dell Precision Optimizer are:

- Performance
- Track & Analyze Engine (TA)
- System Maintenance (SM)
- Dell Precision Optimizer Manager CLI (dpoCmd.exe)

Each of the above subsystems is implemented as a Windows service that also acts as a COM server. The Dell Precision Optimizer installer package installs the services along with Dell Precision Optimizer support DLL(s), user interface (UI) components, kernel mode device driver(s), etc. into the POA Installation folder. Additionally, a taskbar application may be installed and launched whenever the user logs on. This application would be responsible for notifying the user of various POA events such as update completion and reboot required.

The Dell Precision Optimizer installer package will also be responsible for creating a software registry key that will be used by Dell Precision Optimizer modules. For the purpose of this document we shall assume the following defaults:

- Installation Folder: `C:\Program Files\Dell\PPO`
- Registry Path: `HKLM\Software\Dell\PPO`
- Runtime Data: `C:\ProgramData\Dell\PPO`

The installation package may also copy some default Profiles and Policies to the installation folder.

2.1 Uninstalling Dell Precision Optimizer

Dell Precision Optimizer application can be uninstalled from the system using the steps below.

The uninstall command can be fetched from the registry by reading the value of the string “UninstallString” from the following location:
[For 64bit system]
HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\Microsoft\Windows\CurrentVersion\Uninstall\{D66A3355-FEA4-4F60-8BAF-D6CBEDB396D8}

[For 32bit system]
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\{D66A3355-FEA4-4F60-8BAF-D6CBEDB396D8}

A sample value for “UninstallString” is shown below:
"C:\Program Files (x86)\InstallShield Installation Information\{D66A3355-FEA4-4F60-8BAF-D6CBEDB396D8}\setup.exe" -runfromtemp -l0x0007 -removeonly

Where -l0x0007 may be different for the system.

From a command prompt (run as administrator), typing the above command will launch the uninstall process of the Dell Precision Optimizer application. The above command can be modified to run the uninstall silently by adding the following:

-s -f1<full-path-of-iss>

E.g. If the silent response file (.iss file) is in in c:\temp folder and its name is “uninst.iss” then following command will run uninstall silently.

"C:\Program Files (x86)\InstallShield Installation Information\{D66A3355-FEA4-4F60-8BAF-D6CBEDB396D8}\setup.exe" -runfromtemp -l0x0007 –removeonly –s –f1c:\temp\uninst.iss

3. Performance

The Performance subsystem consists of the following modules:

- Policy Processing Engine (PPE)
- Profile Update Tool (profUpd.exe)
- Update Options Tool (upgradeOpt.exe)

3.1 Policy Processing Engine (PPE)

PPE is implemented as a Windows Service which starts execution as soon as the machine boots up regardless of whether the user is logged on.

PPE provides an interface which can be used to perform tasks such as the following:

- Activate or de-activate profiles
• Enumerate input and output parameters to allow new policies to be created.
• Save and retrieve profiles and policies for machines, users or third-party Dell Precision Optimizer-aware applications.

3.2 Profile Update Tool

This command line tool (profUpd.exe) will be used by Dell Precision Optimizer UI to check and update the profiles from the Dell server. A local configuration file should be used to inform Dell Precision Optimizer the Dell server address and which protocol to use (e.g. HTTP, HTTPS or FTP).

NOTE: All profiles and policies on the server are digitally signed and stored encrypted using AES-256 algorithm.

3.3 Profile Update Tool

This command line tool (upgradeOpt.exe) will be used by Dell Precision Optimizer UI to start the Dell support site URL in the default browser. The site will show lists the hardware upgrade options available for the specific system.

4. Track and Analyze (TA)

Analysis sub-system provides the ability to generate the following types of reports:

• System Analysis Reports
• Workload Analysis Reports
• CPU Intelligence Reports
• Graphics Intelligence Reports
• System Health Reports
• Smart Notifications

System Analysis Reports provide data collected by Dell Data Vault (DDV) application in an XML format.

The Workload Analysis feature allows the user to analyze their workload.

4.1 System Analysis Reports

The user can enable or disable these reports using the Dell Precision Optimizer COM interface. This interface also allows the user to configure how often System Analysis reports will be generated, to enumerate and read existing reports. The System Analysis Report XML contains the report data which is divided into <ddv_group> and <ddv_subgroup> elements. All data related to the same category will be under the same group. E.g. all thermistor 0 related data will be under DDV_GROUP called “Thermistor 0”.

4.1.1 Report Settings

Enable System Analysis
Type: ENABLE/DISABLE Checkbox
Default: DISABLED
Description: This setting will allow DDV subsystem to start its data collection. When disabled, DDV shall not be active. Once enabled, DDV reports shall periodically be generated until this setting is manually turned off. Any change in this category selection should cause all existing DDV raw data to be discarded.

Generate Report
Type: List of following options (select one):
- After 24 Hours (Daily)
- After 12 Hours
- After 8 Hours
- After 6 Hours
- After 4 Hours
Default: Daily
Description: Once enabled, DDV collects raw data and generates reports periodically. This setting controls how often the raw data is processed by DDV and converted into a new report. Any change in this category selection should cause all existing DDV raw data to be discarded.

Enable data collection
Type: Has the following checkboxes. More than one category may be checked.
- Battery
- Thermal
- Fan
- Processor
- Memory
- Storage
- Network
Default: All checkboxes are enabled
Description: This setting will allow the user to control which categories of data will be displayed in the report.

4.2 Workload Analysis
Dell Precision Optimizer 5.0 and later allows the user to characterize their workloads and determine their resource usages. When the user initiates the analysis, Dell Precision Optimizer should collect system resource usage parameters (CPU, Memory, Disk & Graphics).

4.3 CPU Intelligence Reports
Dell Precision Optimizer 5.0 and later allows the user to view enhanced Intel CPU information which includes processor information as well as live data for each logical processor. This data is displayed in the form of line graphs by Dell Precision Optimizer UI.

The UI will use the COM interface to get the following processor information:

- CPU Name
- Number of Sockets
- Number of Physical Cores
- Hyper-Threading State (Enabled/Disabled)
- L1 Cache Size (KB)
- L2 Cache Size (KB)
- L3 Cache Size (KB)
- CPU Utilization per logical processor
- CPU Active Relative Frequency per logical processor (to determine Turbo Residency)
- Processor Queue Length for the system
- Number of System Threads

4.4 Graphics Intelligence Reports

For supported Nvidia and AMD graphics adapters only, Dell Precision Optimizer 5.0 and later allows the user to view enhanced Graphics. This includes graphics adapter and software information as well as live data for each GPU. This data is displayed in the form of line graphs by Dell Precision Optimizer UI.

The UI will use Dell Precision Optimizer interface to get the following GPU information:

- Number of GPUs
- Graphics Driver Version
- Graphics Adapter Name (Active GPU 0 Only)
- Video BIOS Version (Active GPU 0 Only)
- Framebuffer Size (Active GPU 0 Only)

The Graphics live data is available from certain Nvidia & AMD adapters only and that too only when a user is logged on.

**NOTE:** On some mobile systems with AMD graphics adapters, valid live data may only be displayed when an active load is running on the AMD graphics adapter.

The following live information for each GPU shall be collected and displayed in the form of line graphs.

- GPU Utilization
- GPU Temperature
- GPU Fan #0 Speed (%)
- Video Memory Utilization

4.5 System Health Reports

Dell Precision Optimizer 5.0 and later allows the user to run System Health reports. These are standard Microsoft-provided reports such as a System Report, Battery Report, and Reliability Report. The user should be able to generate a new report or view the last report generated previously. Only users with local administrator privileges can use this option.

This feature will act as a shortcut to existing Microsoft tools. The following reports shall be available from this dashboard:

- System Diagnostic Report containing:
  - diagnostics results listing errors and warning in the system
  - resource usage overview
• System Reliability Report containing:
  o List of Application, Windows & Miscellaneous failures in the last few weeks
  o Informational Events and Warnings during that period
  o Windows Stability Index
• Battery Report containing*1
  o Installed Battery Details
  o Recent Usage & History
  o Battery Capacity & Life Estimates
*1 This feature is available only on Windows 8 and above.

4.6 Smart Notifications
Dell Precision Optimizer 4.0 and later allows the user to enable Smart Notifications. Smart Notifications allow the user to get notifications in any of the following cases:

  • Excessive CPU utilization
  • Excessive Memory utilization
  • Excessive Disk Read/Write operations

These notifications will work only if either System Analysis or Workload Analysis is turned ON. These notifications are generated by examining the data collected during analysis for the previous day.

Therefore, the user may not see any notifications if only Workload Analysis is run for 4 hours.

4.7 Upgrade Options
Dell Precision Optimizer UI provides a new link to Dell support website where a user may view and order optional/upgrade parts for his/her specific platform. The service tag of the system shall be used by the application to determine what upgrades are available. This feature is internally implemented using the upgratdeOpt.exe tool.

5. System Maintenance (SM)
Dell Precision Optimizer SM allows the user to filter updates that are seen or applied based on the following criteria derived from DCU:

  a) Criticality (Critical, Recommended, Optional)
  b) Type (Hardware Drivers, Application, BIOS, Firmware)
  c) Category (Audio, Chipset, Input, Network/Bluetooth, Storage, Video, Others)
6. User Feedback

Dell Precision Optimizer UI provides an option to the user to send feedback back to Dell. Dell Precision Optimizer UI provides a link/button which the user can click to initiate this feedback. UI will launch a URL in the browser which will allow the user to use a Dell standard form to provide feedback for Dell Precision Optimizer.

7. Improve Dell Precision Optimizer

The Dell Precision Optimizer Customer Experience Improvement Program allows Dell customers to impact the development of future Dell Precision Optimizer releases. By sharing information with Dell regarding how you use Dell Precision Optimizer, you can contribute to improvements of future versions of the product.

The Dell Precision Optimizer Customer Experience Improvement Program adheres to all of the provisions of the Dell privacy policy. Data collected will be limited to Dell Precision Optimizer usage and the workstation's service tag. No personal information of data will be collected. You may opt in or out of the Program at any time.

This feature is disabled by default.

8. Enterprise Tools

8.1 WMI Providers

Dell Precision Optimizer 5.0 and above includes a WMI provider to allow access to the following information. Please refer to APPENDIX A for MOF descriptions. The following two files are part of the Dell Precision Optimizer package:

- **Dell Precision Optimizer WMI Provider: dpoProv.mof**
- **Dell Precision Optimizer SMS MOF definition file: sms_def_dpo3.mof**

- DDV Reports
- Product Version
- Last Check For Update Time
- Last System Update Time
- Last Check For Profiles
- Profile/Policy Trigger History
- List of Active Profiles
- Smart Notifications
8.2 DPOCMD.EXE

Dell Precision Optimizer 5.0 and later provides CLI tool, dpoCmd.exe, to allow the IT administrator the following capabilities:

- Add a new Profile or Policy
- List all Profiles
- Enable or Disable a Profile
- Schedule System Analysis report(s) with specific filters
- Run Dell Precision Optimizer Dell System Update with filters
- Check for Dell Precision Optimizer Dell System Updates with filters
- Export a user created profile
- Import a user created profile
- Upgrade to Premium version
- Enable/Disable UI features using the following CLI options; these are also controlled using new command line switches in Dell Precision Optimizer installer:

<table>
<thead>
<tr>
<th>Control</th>
<th>Definition</th>
<th>Default</th>
<th>Command Line Switch</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProfileControl</td>
<td>If 0, then do not allow user to enable/disable profiles,</td>
<td>1</td>
<td>PROFCTRL</td>
</tr>
<tr>
<td>ProfileUpdate</td>
<td>If 0, then do not allow user to check for new profiles</td>
<td>1</td>
<td>PROFUPD</td>
</tr>
<tr>
<td>SystemUpdate</td>
<td>If 0, then do not allow user to check for system updates</td>
<td>1</td>
<td>SYSUPD</td>
</tr>
<tr>
<td>DDVControl</td>
<td>If 0, then do not allow user to Enable/Disable System Analysis reports</td>
<td>1</td>
<td>DDVCTRL</td>
</tr>
<tr>
<td>UserFeedback</td>
<td>If 0, then do not allow user to send Dell Precision Optimizer feedback</td>
<td>1</td>
<td>USRFB</td>
</tr>
<tr>
<td>UpgradeOptions</td>
<td>If 0, then do not allow user to check for upgrade options</td>
<td>1</td>
<td>UPGOPT</td>
</tr>
<tr>
<td>WorkloadAnalysis</td>
<td>If 0, then do not allow user to run Workload Analysis</td>
<td>1</td>
<td>WKLANL</td>
</tr>
<tr>
<td>GfxPlugins</td>
<td>If 0, then do not show GfxPlugin options to the user</td>
<td>1</td>
<td>GFXPLUGINS</td>
</tr>
<tr>
<td>ImproveDPO</td>
<td>If 0, then do not show user Improve Dell Precision Optimizer setting</td>
<td>1</td>
<td>IMPROVEDPO</td>
</tr>
<tr>
<td>ISVCertDrvr</td>
<td>If 0, then do not allow user to view/install ISV Certified graphics drivers</td>
<td>1</td>
<td>ISVCERTGFX</td>
</tr>
<tr>
<td>SmartAlerts</td>
<td>If 0, then do not allow user to enable/disable Smart Alerts</td>
<td>1</td>
<td>SMARTALERT</td>
</tr>
</tbody>
</table>
CLI Usage:

```
dpoCmd.exe -savePolicy <complete_dpx_path>
```
```
dpoCmd.exe -saveProfile <complete_dpx_path>
```
```
dpoCmd.exe -listProfiles
```
```
dpoCmd.exe -enableProfile <profile_guid>
```
```
dpoCmd.exe -disableProfile <profile_guid>
```
```
dpoCmd.exe -scheduleReports <numReports> <reportDuration> [-r <ddvSubSystem>] [-r <ddvSubSystem>] ...
```

where,

```
<reportDuration> can be one of 0, 4, 6, 8, 12
  0 means daily report
  4 means 4 hour report
  6 means 6 hour report and so on.
```

```
-r <ddvSubSystem>
```

will remove that subsystem and the data will not appear in the DDV reports that are generated. <ddvSubSystem> can be one of the following:

- Battery
- Thermal
- Fan
- Processor
- Memory
- Network
- Storage

```
dpoCmd.exe --cancelReports
```
```
dpoCmd.exe -enableFeatures <feature> [...]
```

where <feature> can be one of the following:

- PROFCTRL
- PROFUPD
- SYSUPD
- DDVCTRL
- USRFB
- UPGOPT
- WKLANL
- GFXPLUGINS
- IMPROVEDPO
- ISVCERTGFX
- SMARTALERT

```
dpoCmd.exe -disableFeatures <feature> [...]
```

where <feature> can be one of the following:

- PROFCTRL
- PROFUPD
- SYSUPD
- DDVCTRL
USRFB
UPGOPT
WKLANL
GFXPLUGINS
IMPROVEDPO
ISVCERTGFX
SMARTALERT

\texttt{dpoCmd.exe} -updateNow -criticality:CRO -filter:BDAF -device:ACMSNV <activityLogFileName>
\texttt{dpoCmd.exe} -checkForUpdatesNow -criticality:CRO -filter:BDAF -device:ACMSNV <activityLogFileName>

where -criticality: can be one or more of the following:
C => Critical
R => Recommended
O => Option

where -filter: can be one or more of the following:
B => BIOS
D => Drivers
A => Applications
F => Firmware

where -device: can be one or more of the following:
A => Audio
C => Chipset
M => Mouse/Keyboard
S => Storage
N => Network/Bluetooth
V => Video

\texttt{dpoCmd.exe} -exportProfile <profile_guid or unique_profile_name> <dpzFileName>
\texttt{dpoCmd.exe} -importProfile <dpzFileName>
\texttt{dpoCmd.exe} -upgradeToPremium <licenseKey>

where licenseKey: is an alpha numeric key and not a file that contains the key

8.3 Setup Command Line Switches
Dell Precision Optimizer 5.0 installer provides command line switches to allow IT administrator to control certain behaviors of the client package. This list is mentioned above in Section 8.2.

\texttt{Setup.exe PROFUPD=0 WKLANL=0} \textit{To install Dell Precision Optimizer, where the user will not be allowed to check for new profiles or run workload analysis.}

In addition, a new options \texttt{GUI=0}, allows the IT administrator to install the Dell Precision Optimizer client without any UI component, i.e. head less mode. The user will not be able to control the software. The IT administrator may use the new CLI tool to enable/disable other run time features.
8.4 SCCM

This is one of the methods used by the IT administrators today to centrally manage their systems and software applications. In this section we will provide examples of how IT Administrators can use SCCM tools to manage the Dell Precision Optimizer application.

NOTE: There are methods and tools other than SCCM in the industry. Use the examples below to manage Dell Precision Optimizer in those environments.

8.4.1 Instructions for creating the Dell Precision Optimizer application package

Here are some steps that you can perform to create a Dell Precision Optimizer package that can be deployed to selected client system in the Enterprise. Note: the exact steps may differ slightly based on the SCCM version that you are using.

a) Download the Dell Precision Optimizer files required for installation.
b) In Configuration Manager Console:
   - open the Software Library Page
   - Click on Overview folder
   - Click on Application Management
   - Right click Applications and select: Create Application
c) In the Create Application Wizard
   - Select Manually specify the application information
   - Give the application a name, i.e. Dell Precision Optimizer 5.00.02, click next
   - Click next in Application Catalog
   - Click add in the Deployment Types page
   - In the Create Deployment Type Wizard, select Type: script installer, click next
   - Give the deployment type a name, click next
   - Type the location of the Dell Precision Optimizer files in the Content Location
   - Type the following in Installation Program:
     - “PoaInstaller.exe” /s
   - In the Detection Methods tab click Add clause
   - The detection rule is as follows:
     - Setting Type: Registry
     - Hive: HKEY_LOCAL_MACHINE
     - Key: Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall\{D66A3355-FEA4-4F60-8BAF-D6CBEDB396D8}
   - Click OK to close the Detection Rule window, click Next In the Create Deployment Type Wizard
   - Specify the user experience as follows:
     - Installation behavior: Install for system
     - Login Requirement: Whether or not a user is logged on
     - Installation Program Visibility: Normal
   - Click Next in the Requirements tab
   - Click Next in the Dependencies tab
• Click Next in Summary, verify that the Deployment Type was created successfully and close the Create Deployment Type wizard

d) In the Create Application Wizard, click next in the Deployment Types tab, click next in the Summary Tab and confirm that the application was created successfully

8.4.2 Instructions for Deploying the Application

Once you have created the package, use the following instruction to deploy it to selected clients:

a) Right click the application to deploy and select Deploy
b) Select the device collection you would like to install Dell Precision Optimizer on
c) Make sure Automatically distribute content for dependencies is checked, click next
d) In the Content tab, click Add to select the distribution point
e) In the deployment settings tab have the following:
   • Action: Install
   • Purpose: Required
f) In the Scheduling tab click next
g) In the User Experience tab select:
   • User notifications: Display in Software Center, and only show notifications for computer restarts
h) Click next in the Alerts tab, click next in the Summary tab, and verify deployment completion

8.4.3 Verifying Deployment Success in Client systems

a) Open Software Center in the client system, verify that Dell Precision Optimizer is installed (it may take a few minutes for the installation to take place after deploying the application)
b) Go to C:\Windows\CCM\Logs and check AppDiscovery.Log, AppIntent.log, and AppEnforce.log for troubleshooting

8.4.4 Changing Dell Precision Optimizer Client Behavior using DPOCMD.EXE

Here are the steps that you can take to run Dell Precision Optimizer CLI (dpoCmd.exe) on a target system to change the behavior of Dell Precision Optimizer software on that system. The following example illustrates the use of dpoCmd.exe to enable a Dell Precision Optimizer profile (After Effects by Adobe).

Create a software package first:
1. In the Configuration Manager Console
   a. Open the Software library page
   b. Click on the Overview tab
   c. Open the Application Management tab
   d. Right click Packages and select Create new package
2. In the Create Package and Program Wizard
a. Set the Name: Enable a Dell Precision Optimizer profile  
b. Specify information about the package and click next

3. In the Program Type tab select Standard program

4. In the Standard Program tab:
   a. Name: Enable Adobe After Effects  
   b. Command line: dpoCmd.exe –enableProfile {2F066600-FA52-4F57-890D-2621D39B0BE9}
   c. Startup folder: C:\program files\dell\ppo
   d. Run: Normal
   e. Program can run: Whether or not a user is logged on  
   f. Run mode: Run with administrative rights
   g. Drive mode: Runs with UNC name

5. In the Requirements tab select ‘This program can run on any platform’

6. Click next, review the package summary and verify the package was created successfully

Deploying “Enable a Dell Precision Optimizer Profile” software package

1. In the Configuration Manager Console
   a. Open the Software library page
   b. Click on the Overview tab
   c. Open the Application Management tab
   d. Click on Packages

2. Right click the “Enable DPO profile” software package and select Deploy

3. In the Deploy Software Wizard:
   a. In the general tab ,click browse to select the device collection, click next  
   b. In the content tab, click add to add a distribution point, click next
   c. In the deployment settings tab, have the following:
      i. Action: Install
      ii. Purpose: Required
      iii. Check the ‘Send wake-up’ packets box
   d. In the scheduling tab select the time of deployment and make sure the ‘Rerun behavior is: Always rerun program’. To deploy now, click New and select ‘Assign immediately after this event: As soon as possible’
   e. In the User Experience tab make sure the following check boxes are checked
      i. Software Installation
      ii. System restart(if required to complete installation)
      iii. Commit changes at deadline or during a maintenance window (requires restarts)
   f. In the distribution points tab:
      i. Deployment options: Download content from distribution point and run locally
      ii. Deployment options: Download content from distribution point and run locally
      iii. Make sure the ‘Allow clients to share content with other clients on the same subnet’ is checked
   g. Click next and verify the deployment successfully completed

8.4.5 SSRS Reports

As a system administrator you can create various reports based on the data collected from Dell Precision Optimizer’s WMI providers. If this is desired, you can include the sms_def_dpo3.mof to extend the DB definitions and pull corresponding data from Dell Precision Optimizer client systems. You may select some or all the data elements that you will like to review. The default is set to select all Dell Precision Optimizer data elements.
**Importing the sms_def_dpo3.mof file to set hardware inventory classes**

1. In the Configuration Manager Console
   a. Open the Administration page
   b. Click the Overview tab
   c. Click the Site Configuration tab and select Client Settings
2. Right click an existing Client Setting and select properties or create a new Custom Client Setting
3. In the Hardware Inventory tab select ‘Set Classes’
4. Select Import and browse to the location of the sms_def_dpo3.mof file
5. Click Ok to import the file and close the Hardware Inventory Classes Window

Once collected data is populate d in the SQL database, you can create different type of Dell Precision Optimizer reports. A handful of samples (*.RDL) are provided with the Dell Precision Optimizer software. You can import these RDL files, connect them to your SQL database and run the reports.

**To Import an RDL file**

1. Open SQL server data tools
2. In the Solution Explorer right click the folder in which you would like to add the RDL file
   a. Select Add Existing Item
   b. Select the RDL file
3. Once the file is imported, open the file and select the Design tab

**To ensure the RDL file is using the right data source**

1. In the Report Data pane, click on Datasets and right click one of the data sets and select ‘Dataset Properties’
2. In the Dataset Properties window:
   a. Make sure ‘Use a dataset embedded in my report’ is selected
   b. Under Data source, click ‘New…’
   c. In the Data Source Properties window select ‘Use shared data source reference’ and select the correct data source
   d. Click ok
3. Repeat steps 1 and 2 for all other data sets in the Datasets folder
Here are some of the screen shots of the reports that are provided with Dell Precision Optimizer:

**Disk Information**

This report displays disk information across multiple systems using their latest system analysis report.

<table>
<thead>
<tr>
<th>Service Tag</th>
<th>Bytes Read (MB)</th>
<th>Bytes Write (MB)</th>
<th>Read Time (%)</th>
<th>Write Time (%)</th>
<th>Idle Time (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>98K94X1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>F8W4WZ1</td>
<td>0</td>
<td>460</td>
<td>0</td>
<td>0</td>
<td>98</td>
</tr>
</tbody>
</table>

**Disk Information**

This report displays disk information across multiple systems using their latest system analysis report.
## Disk Information

This report displays disk information for a single system across multiple reports.

<table>
<thead>
<tr>
<th>Date Reported</th>
<th>Bytes Read (MB)</th>
<th>Bytes Write (MB)</th>
<th>Read Time (%)</th>
<th>Write Time (%)</th>
<th>Idle Time (%)</th>
<th>Hours On</th>
</tr>
</thead>
<tbody>
<tr>
<td>2015-07-15T11:06:53-05:00</td>
<td>5</td>
<td>52</td>
<td>0</td>
<td>0</td>
<td>98</td>
<td>4</td>
</tr>
<tr>
<td>2015-07-15T07:06:53-05:00</td>
<td>16</td>
<td>60</td>
<td>0</td>
<td>1</td>
<td>98</td>
<td>4</td>
</tr>
<tr>
<td>2015-07-15T03:06:53-05:00</td>
<td>56</td>
<td>94</td>
<td>0</td>
<td>0</td>
<td>98</td>
<td>3.1</td>
</tr>
<tr>
<td>2015-07-14T23:06:53-05:00</td>
<td>6</td>
<td>55</td>
<td>0</td>
<td>0</td>
<td>98</td>
<td>4</td>
</tr>
<tr>
<td>2015-07-14T19:07:01-05:00</td>
<td>0</td>
<td>46</td>
<td>0</td>
<td>0</td>
<td>98</td>
<td>4</td>
</tr>
<tr>
<td>2015-07-14T15:06:46-05:00</td>
<td>0</td>
<td>44</td>
<td>0</td>
<td>0</td>
<td>98</td>
<td>3.9</td>
</tr>
<tr>
<td>Time</td>
<td>CPU Usage</td>
<td>GPU Usage</td>
<td>Memory Usage</td>
<td>Disk Usage</td>
<td>Network Usage</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------</td>
<td>-----------</td>
<td>--------------</td>
<td>------------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td>2015-06-22T15:24:47-05:00</td>
<td>36%</td>
<td>35%</td>
<td>0%</td>
<td>0%</td>
<td>9%</td>
<td></td>
</tr>
<tr>
<td>2014-12-16T03:26:33-05:00</td>
<td>0%</td>
<td>17%</td>
<td>0%</td>
<td>0%</td>
<td>9%</td>
<td></td>
</tr>
<tr>
<td>2014-12-15T23:26:33-05:00</td>
<td>0%</td>
<td>17%</td>
<td>0%</td>
<td>0%</td>
<td>9%</td>
<td></td>
</tr>
<tr>
<td>2014-12-15T19:26:34-05:00</td>
<td>0%</td>
<td>17%</td>
<td>0%</td>
<td>0%</td>
<td>9%</td>
<td></td>
</tr>
</tbody>
</table>
Switch

Disk Information

This report displays disk information for a single system across multiple reports.
8.5 KACE

This is one of the methods used by the IT administrators today to centrally manage their systems and software applications. In this section we will provide examples of how IT Administrators can use KACE Appliance to manage the Dell Precision Optimizer application.

NOTE: The following steps were verified on KACE Appliance 6.4.120756 K1000. If you are using a different version of KACE, then the actual steps may vary slightly.

8.5.1 Instructions for deploying Dell Precision Optimizer using KACE

An IT administrator could use the following procedure to deploy Dell Precision Optimizer application on select client systems in their domain.

Create an installation script

From your KACE Appliance console:

- Navigate to Scripting->Scripts->Choose Action->New
- On the Script Detail page enter the following information:

  Name -> Install DPO
  Enabled -> Check the box
  Type -> Online K-Script
  Description -> This script will install DPO client software...
  Deploy -> None
  Operating Systems -> Uncheck Select Specific Operating Systems and select Microsoft Windows
  (Alternate) Operating Systems -> Select specific Windows OSes for deployment
  Windows Run As -> Local System
  Notify -> None
  Schedule -> None
  Dependencies -> Add all DPO package files as new dependencies
  Tasks -> Select New Task

Verifying deployment:

- Verify -> Click Add, then select “Launch a program”, enter the following data:
  - Directory -> $(KACE_DEPENDENCY_DIR)
  - File -> Poainstaller.exe
  - Wait for Completion -> CHECKED
  - Visible -> UNCHECKED
  - Parameters -> LOGFILE=c:\temp\dpo.log /s
  - <Save Changes>

- On Success -> None
- Remediation -> None
- On Remediation Success -> None
- One Remediation Failure -> None
- Tasks -> Select New Task

Verifying file existence:

- Verify -> Click Add, then select “Verify a file exists”, enter the following data:
  - Directory -> C:\Program Files\Dell\DPO
Run installation script on select systems
From your KACE Appliance console:

- Scripting -> Run Now
- Select “Install DPO” from the Scripts drop down menu
- Under Labels, select a label of Windows devices where you wish to deploy Dell Precision Optimizer OR manually select a set of systems
- Click Run Now
- Click Save

This will initiate the deployment of Dell Precision Optimizer client software on selected systems. These steps are for reference only can could be easily customized for your need.

8.5.2 Changing Dell Precision Optimizer Client Behavior using DPOCMD.EXE
Here are the steps that you can take to run Dell Precision Optimizer CLI (dpoCmd.exe) on a target system to change the behavior of Dell Precision Optimizer software on that system. The following example illustrates the use of dpoCmd.exe to enable a Dell Precision Optimizer profile (After Effects by Adobe).

Create an installation script
From your KACE Appliance console:

- Navigate to Scripting->Scripts->Choose Action->New
- On the Script Detail page enter the following information:
  
  **Name** -> Enable DPO Profile After Effects
  **Enabled** -> Check the box
  **Type** -> Online K-Script
  **Description** -> This script will enable After Effects profile under DPO client software...
  **Deploy** -> None
  **Operating Systems** -> Uncheck Select Specific Operating Systems and select Microsoft Windows (Alternate) Operating Systems -> Select specific Windows OSes for deployment
  **Windows Run As** -> Local System
  **Notify** -> None
  **Schedule** -> None
  **Dependencies** -> None
  **Tasks** -> Select New Task
  **Verify** -> Click Add, then select “Launch a program”, enter the following data:
Directory C:\Program Files\Dell\PPO
File dpoCmd.exe
Wait for Completion CHECKED
Visible UNCHECKED
Parameters -enableProfile {2F066600-FA52-4F57-890D-2621D39B0BE9}

On Success -> None
Remediation -> None
On Remediation Success -> None
One Remediation Failure -> None
Tasks -> Select New Task
Verify -> Click Add, then select “Verify a file exists”, enter the following data:
  Directory C:\Program Files\Dell\PPO
  File dpoCmd.exe
  <Save Changes>
On Success -> None
Remediation -> None
On Remediation Success -> None
One Remediation Failure -> None
Save

Run this script on select systems
From your KACE Appliance console:

- Scripting -> Run Now
- Select “Enable DPO Profile After Effects” from the Scripts drop down menu
- Under Labels, select a label of Windows devices where you wish to deploy Dell Precision Optimizer OR manually select a set of systems
- Click Run Now

8.5.3 Custom Reports
Here are some examples on how you can collect some data from Dell Precision Optimizer clients using its WMI classes and create custom reports. Dell Precision Optimizer provides a large set of WMI classes to allow an IT administrator to create a huge variety of reports. This following illustrates the basic steps to create a Dell Precision Optimizer report. An IT administrator can customize what data needs to be collected, how frequently and finally how to present that data.

Create Custom Inventory Rules
From your KACE Appliance console:

- Inventory -> Software
- Choose Action -> New
- Software Details page, enter the following information
Name: DPO Sample Inventory
Version: v1
Publisher: Dell
Supporting Operating Systems: Select OSes
Custom Inventory Rule:
   ShellCommandTextReturn(wmic /namespace:root\cimv2 DPO Path DPO_Profiles get /ALL)

Save
- Click back into the new custom inventory record and hover over the record just created. Note the ID# at the end of the URL. URL with the ID# is visible at the lower left hand corner of the page. You will need this later for creating the report.

Force Inventory Collection
From your KACE Appliance console:
- Inventory -> Devices
- Select the device(s) where Dell Precision Optimizer is installed (you could use a SmartLabel for this purpose)
- Choose Action -> Force Inventory
- Once the inventory cycle has completed, navigate into one of the selected devices that was online
- Device Detail page -> Software -> expand Custom Inventory Fields -> This should show a list of profiles and their current state

NOTE: Now that you have Script and Custom Inventory setup, and completed a Custom Inventory cycle on all desired systems, it is time to take advantage of your K1000s reporting capabilities! While you can definitely pull the Dell Precision Optimizer information out of the K1000 using a Wizard based report, we are going to use a custom SQL report to process and filter our information into a useful report.

Create Report
From your KACE Appliance console:
- Reporting -> Reports
- Choose Action -> New (SQL)
- Report Detail page -> Enter the following data

<table>
<thead>
<tr>
<th>Title</th>
<th>Dell Precision Optimizer Sample Profile Report</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>This is a sample Dell Precision Optimizer report ...</td>
</tr>
<tr>
<td>Category</td>
<td>&lt;any&gt; or New Category -&gt; DPO Reports</td>
</tr>
<tr>
<td>SQL</td>
<td>SELECT MACHINE.NAME AS Name, MACHINE.IP AS IP, MACHINE.USER_LOGGED AS LoggedUser, MACHINE.CS_MANUFACTURER AS Manufacturer, MACHINE.CS_MODEL AS Model, MACHINE_CUSTOM_INVENTORY.STR_FIELD_VALUE AS MACHINE_CUSTOM_INVENTORY_XXXX, COUNT(MACHINE_CUSTOM_INVENTORY.STR_FIELD_VALUE) AS Total_Devices</td>
</tr>
</tbody>
</table>
FROM MACHINE_CUSTOM_INVENTORY
JOIN MACHINE ON MACHINE.ID = MACHINE_CUSTOM_INVENTORY.ID
WHERE MACHINE_CUSTOM_INVENTORY.SOFTWARE_ID = XXXX
GROUP BY MACHINE_CUSTOM_INVENTORY.STR_FIELD_VALUE
ORDER BY MACHINE.CS_MANUFACTURER ASC, MACHINE.CS_MODEL ASC

Replace XXXX with the ID# of your custom inventory which was collected above when we created the Custom Inventory Rule.

- Save

**Run Report**

From your KACE Appliance console:

- Reporting -> Reports
- Search -> Enter “DPO”, this will list your reports
- Select the desired report, e.g. DPO Sample Profile Report, and click on report format you want, say HTML
APPENDIX A - dpoCmd.exe Exit Codes

typedef enum {
    EXIT_CODE_SUCCESS = (int) 0,
    EXIT_CODE_ERROR_GET_COMP_NAME = (int) 1,
    EXIT_CODE_COINIT_FAILED = (int) 2,
    EXIT_CODE_PROFILE_NOT_FOUND = (int) 3,
    EXIT_CODE_ERROR = (int) 4,
    EXIT_CODE_ERROR_GET_COMP_SID = (int) 5,
    EXIT_CODE_COINIT_SECURITY_FAILED = (int) 6,
    EXIT_CODE_MISSING_COM_INTERFACE = (int) 7,
    EXIT_CODE_PROFILE_GETSTATE_FAILED = (int) 8,
    EXIT_CODE_PROFILE_SETSTATE_FAILED = (int) 9,
    EXIT_CODE_MISSING_STORE = (int) 40,
    EXIT_CODE_NULL_STORE = (int) 41,
    EXIT_CODE_READFILE_FAILED = (int) 42,
    EXIT_CODE_WRITEFILE_FAILED = (int) 43,
    EXIT_CODE_OUT_OF_MEM = (int) 44,
    EXIT_CODE_SAVE_STORE_FAILED = (int) 45,
    EXIT_CODE_ENCRYPTION_FAILED = (int) 46,
    EXIT_CODE-DDV_REPORTS_ALREADY_SCHEDULED = (int) 60,
    EXIT_CODE_ENABLE_DDV_FAILED = (int) 61,
    EXIT_CODE_SET_DDV_FILTERS_FAILED = (int) 62,
    EXIT_CODE_INVALID_DDV_REPORT_DURATION = (int) 63,
    EXIT_CODE_SET_REPORT_FREQ_FAILED = (int) 64,
    EXIT_CODE_SET_REPORT_NUM_FAILED = (int) 65,
    EXIT_CODE_DISABLE_DDV_FAILED = (int) 66,
    EXIT_CODE_ERROR_ENUM_DDV_SUBSYSTEMS = (int) 67,
    EXIT_CODE_DO_UPDATE_FAILED = (int) 70,
    EXIT_CODE_PREV_CHECK_FAILED = (int) 71,
    EXIT_CODE_PREV_UPDATE_ACTION_IN_PROGRESS = (int) 72,
    EXIT_CODE_REGISTER_EVENTS_FAILED = (int) 73,
    EXIT_CODE_CHECK_UPDATE_FAILED = (int) 74,
    EXIT_CODE_SET_FEATURE_FAILED = (int) 80,
    EXIT_CODE_UI_IS_RUNNING = (int) 98,
    EXIT_CODE_USAGE_ERROR = (int) 99
} EXIT_CODE;
APPENDIX B - WMI Class Definition Files

/*****************************************************************************
 * DPOProv.mof
 * Last Updated: 03/06/2015
 * This file defines the classes exposed by "dpoProv".
 *****************************************************************************

#pragma autorecover
#pragma namespace("\\\root\cimv2")
instance of __Namespace
{
    Name = "DPO";
}
#pragma namespace("\\\root\cimv2\DPO")

/*****************************************************************************
 * DPO_HardwareInfo
 * There is one instance of this class for each summary file present on the system.
 * The instance will contain all the hardware data and the statistics from the summary file.
 * HardwareInfoGUID is the unique ID from the summary file.
 * HardwareInfoGUID associates this instance with instances of other dependent classes that may have multiple instances (eg. DPO_Monitor, DPO_BiosInternalLogs etc.)
 *****************************************************************************
[Description("An instance of this class contains all the hardware data and "
 " statistics from a summary file.")],
Dynamic,Provider("DPOProv") ]
class DPO_HardwareInfo
{
    [Description("Unique ID from the summary file.")],
    Key
    string HardwareInfoGUID;
    [Description("Revision of Dell Data Vault.")]
    string DDV_Revision;
    [Description("Date/time when the summary file was created.")]
    string File_Creation_Datetime;
    [Description("Date/time when Dell Data Vault began collecting the raw data.")]
    string Data_Beginning_Date;
    [Description("Date/time when Dell Data Vault stopped collecting the raw data and generated the statistics.")]
    string Data_Ending_Date;
    [Description("Indicates whether this summary was created on service startup, regular timer or on demand.")]
    string Summary_Type;
    [Description("Service Tag of the system obtained from the BIOS.")]
    string System_Service_Tag;
    [Description("Customer Name 1")]
    string Customer_Name_1;
    [Description("Customer Name 2")]
    string Customer_Name_2;
    [Description("Customer Name 3")]
    string Customer_Name_3;
[Description("Customer specific data 1")]
string Customer_Defined_1;

[Description("Customer specific data 2")]
string Customer_Defined_2;

[Description("Customer specific data 3")]
string Customer_Defined_3;

[Description("System Model")]
string System_Model;

[Description("ePPID of the motherboard obtained from the BIOS.")]
string Motherboard_ePPID;

[Description("Current BIOS Version.")]
string BIOS_Version;

[Description("Type of the system eg. Laptop or Desktop")]
string System_Type;

[Description("Serial number of the CPU.")]
string Processor_Serial_Number;

[Description("Processor name.")]
string Processor_Name;

[Description("Processor speed.")]
string Processor_Speed;

[Description("Average of the percentage LCD brightness when the system was on AC.")]
sint16 LCD_Avg_Brightness_AC_Pct;

[Description("Average of the percentage LCD brightness when the system was on battery.")]
sint16 LCD_Avg_Brightness_DC_Pct;

[Description("Video Controller name.")]
string Video_Controller;

[Description("Video controller memory size.")]
sint32 Video_RAM_Bytes;

[Description("Number of displays on the system.")]
sint16 Number_of_Displays;

[Description("Operating system, 32bit vs 64bit & system locale information.")]
string Operating_System;

[Description("AC adapter power (for notebooks only).")]
string AC_Adapter_Type_W;

[Description("Number of hours the system was on.")]
real32 Hours_On;

[Description("Number of hours the system was on when powered by AC.")]
real32 Hours_On_AC;

[Description("Number of hours the system was on when powered by battery (for notebooks only).")]
real32 Hours_On_Batt;

[Description("Number of times the AC adapter was inserted in the system (for notebooks only).")]
sint16 No_Of_AC_Insertions;

[Description("Number of times the primary battery was inserted into the system (for notebooks only).")]
// NameChange  sint16 Number_Of_Battery_Insertions;
sint16 Num_Battery_Insertions;

[Description("Number of times the system was running on battery (for notebooks only.").")]
sint16 Number_of_Battery_Sessions;

[Description("Number of battery sessions where the session was between 0 to 30 mins (for notebooks only.").")]
sint16 Battery_Sessions_0_30mins;

[Description("Number of battery sessions where the session was between 30 mins to 1 hr (for notebooks only.").")]
sint16 Battery_Sessions_30min_1hr;

[Description("Number of battery sessions where the session was between 1 to 2 hrs (for notebooks only.").")]
sint16 Battery_Sessions_1_2hr;

[Description("Number of battery sessions where the session was between 2 to 3 hrs (for notebooks only.").")]
sint16 Battery_Sessions_2_3hr;

[Description("Number of battery sessions where the session was between 3 to 4 hrs (for notebooks only.").")]
sint16 Battery_Sessions_3_4hr;

[Description("Number of battery sessions where the session was between 4 to 6 hrs (for notebooks only.").")]
sint16 Battery_Sessions_4_6hr;

[Description("Number of battery sessions where the session was between 6 to 8 hrs (for notebooks only.").")]
sint16 Battery_Sessions_6_8hr;

[Description("Number of battery sessions where the session was between 8 to 12 hrs (for notebooks only.").")]
sint16 Battery_Sessions_8_12hr;

[Description("Number of battery sessions where the session was greater than 12 hrs (for notebooks only.").")]
sint16 Battery_Sessions_GT12hr;

[Description("Number of system shutdowns.")."]]
sint16 S5_Requests;

[Description("Number of times the system entered Hibernate state (S4).")."]]
sint16 S4_Requests;

[Description("Total time the system was in Hibernate state (S4)")."]]
real32 S4_mins;

[Description("Number of times the system was in Hibernate state (S4) where the time in S4 was between 0 to 30 mins.")."]]
sint16 S4_Event_Bin_0_30_mins;

[Description("Number of times the system was in Hibernate state (S4) where the time in S4 was between 30 mins to 1 hr.")."]]
sint16 S4_Event_Bin_30_60_mins;

[Description("Number of times the system was in Hibernate state (S4) where the time in S4 was between 1 hr to 2 hrs.")."]]
sint16 S4_Event_Bin_60_120_mins;

[Description("Number of times the system was in Hibernate state (S4) where the time in S4 was between 2 to 4 hrs.")."]]
sint16 S4_Event_Bin_120_240_mins;

[Description("Number of times the system was in Hibernate state (S4) where the time in S4 was between 4 to 8 hrs.")."]]
sint16 S4_Event_Bin_240_480_mins;

[Description("Number of times the system was in Hibernate state (S4) where the time in S4 was between 8 to 16 hrs.")."]]
sint16 S4_Event_Bin_480_960_mins;

[Description("Number of times the system was in Hibernate state (S4) where the time in S4 was greater than 16 hrs.")."]]
sint16 S4_Event_Bin_GT960_mins;

[Description("Number of times the system entered Standby/Sleep state (S3).")."]]
sint16 S3_Requests;

[Description("Total time the system was in Standby/Sleep state (S3).")."]]
real32 S3_mins;

sint16 S3_Event_Bin_0_30_mins;

sint16 S3_Event_Bin_30_60_mins;

sint16 S3_Event_Bin_60_120_mins;

sint16 S3_Event_Bin_120_240_mins;

sint16 S3_Event_Bin_240_480_mins;

sint16 S3_Event_Bin_480_960_mins;

sint16 S3_Event_Bin_GT_960_mins;

real32 Avg_CPU_Consumption;

sint16 CPU_0_Pct;

sint16 CPU_1_20_Pct;

sint16 CPU_20_40_Pct;

sint16 CPU_40_60_Pct;

sint16 CPU_60_80_Pct;

sint16 CPU_80_100_Pct;

real32 Avg_CPU_Throttle;

sint16 Throttle_0_Pct;

sint16 Throttle_1_25_Pct;

sint16 Throttle_25_50_Pct;

sint16 Throttle_50_75_Pct;

sint16 Throttle_75_100_Pct;

sint16 C1_State_Pct;

sint16 C2_State_Pct;
sint16 C2_State_Pct;

[Description("Percentage of time the processor (all processors combined) was in C3 state.")] sint16 C3_State_Pct;

[Description("Percentage of time the processor (all processors combined) was in C0 state.")] sint16 C0_State_Pct;

[Description("Number of LID transitions. One open-close is considered as one transition.")] sint16 Lid_Transitions;

[Description("Number of hours the system was ON with LID open.")] real32 Lid_Hours_Open;

[Description("Number of hours the system was ON with LID closed.")] real32 Lid_Hours_Closed;

[Description("Number of dock events.")] sint16 Number_Dock_Events;

[Description("Total system RAM memory.")] string System_RAM_Bytes;

[Description("Total system RAM memory in GB.")] real32 System_RAM_GB;

[Description("Percentage of time the system had to access hard disk to resolve page faults.")] sint16 pgs_per_sec_pct;

[Description("Minimum number of pages read from or written to the disk to resolve hard page faults.")] sint32 min_pgs_per_sec;

[Description("Maximum number of pages read from or written to the disk to resolve hard page faults.")] sint32 max_pgs_per_sec;

[Description("Average number of pages read from or written to the disk to resolve hard page faults.")] real32 avg_pgs_per_sec;

[Description("Percentage of time the system had between 0 to 256 MB of free physical memory.")] real32 FreeMem_0_256MB_Pct;

[Description("Percentage of time the system had between 256 MB to 512 MB of free physical memory.")] real32 FreeMem_256_512MB_Pct;

[Description("Percentage of time the system had between 512 MB to 768 MB of free physical memory.")] real32 FreeMem_512_768MB_Pct;

[Description("Percentage of time the system had between 768 MB to 1024 MB of free physical memory.")] real32 FreeMem_768_1024MB_Pct;

[Description("Percentage of time the system had between 1024 MB to 1280 MB of free physical memory.")] real32 FreeMem_1024_1280MB_Pct;

[Description("Percentage of time the system had between 1280 MB to 1536 MB of free physical memory.")] real32 FreeMem_1280_1536MB_Pct;

[Description("Percentage of time the system had between 1536 MB to 1792 MB of free physical memory.")] real32 FreeMem_1536_1792MB_Pct;

[Description("Percentage of time the system had between 1792 MB to 2048 MB of free physical memory.")] real32 FreeMem_1792_2048MB_Pct;

[Description("Percentage of time the system had between 2048 MB to 2304 MB of free physical memory.")] real32 FreeMem_2048_2304MB_Pct;

[Description("Percentage of time the system had between 2304 MB to 2560 MB of free physical memory.")]
real32 FreeMem_2304_2560MB_Pct;
[Description("Percentage of time the system had between 2560 MB to 2816 MB of free physical memory.")]}
real32 FreeMem_2560_2816MB_Pct;
[Description("Percentage of time the system had between 2816 MB to 3072 MB of free physical memory.")]}
real32 FreeMem_2816_3072MB_Pct;
[Description("Percentage of time the system had more than 3072 MB of free physical memory.")]}
real32 FreeMem_GT3072MB_Pct;
[Description("Percentage of time the system had between 0 to 256 MB of physical memory available to processes running on the computer.")]}
real32 AvailMem_0_256MB_Pct;
[Description("Percentage of time the system had between 256 MB to 512 MB of physical memory available to processes running on the computer.")]}
real32 AvailMem_256_512MB_Pct;
[Description("Percentage of time the system had between 512 MB to 768 MB of physical memory available to processes running on the computer.")]}
real32 AvailMem_512_768MB_Pct;
[Description("Percentage of time the system had between 768 MB to 1024 MB of physical memory available to processes running on the computer.")]}
real32 AvailMem_768_1024MB_Pct;
[Description("Percentage of time the system had between 1024 MB to 1280 MB of physical memory available to processes running on the computer.")]}
real32 AvailMem_1024_1280MB_Pct;
[Description("Percentage of time the system had between 1280 MB to 1536 MB of physical memory available to processes running on the computer.")]}
real32 AvailMem_1280_1536MB_Pct;
[Description("Percentage of time the system had between 1536 MB to 1792 MB of physical memory available to processes running on the computer.")]}
real32 AvailMem_1536_1792MB_Pct;
[Description("Percentage of time the system had between 1792 MB to 2048 MB of physical memory available to processes running on the computer.")]}
real32 AvailMem_1792_2048MB_Pct;
[Description("Percentage of time the system had between 2048 MB to 2304 MB of physical memory available to processes running on the computer.")]}
real32 AvailMem_2048_2304MB_Pct;
[Description("Percentage of time the system had between 2304 MB to 2560 MB of physical memory available to processes running on the computer.")]}
real32 AvailMem_2304_2560MB_Pct;
[Description("Percentage of time the system had between 2560 MB to 2816 MB of physical memory available to processes running on the computer.")]}
real32 AvailMem_2816_2560MB_Pct;
[Description("Percentage of time the system had between 2816 MB to 3072 MB of physical memory available to processes running on the computer.")]}
real32 AvailMem_2816_3072MB_Pct;
[Description("Percentage of time the system had more than 3072 MB of physical memory available to processes running on the computer.")]}
real32 AvailMem_GT3072MB_Pct;
[Description("Average Processor Queue Length.")]}
real32 Average_PQL;
[Description("Minimum Processor Queue Length.")]}
sint16 Min_PQL;
# DPO_Monitor

This has the monitor information from a summary log. There may be multiple instances of this class for each summary file.

```c
/******************************************************************************
* DPO_Monitor
* This has the monitor information from a summary log. There may be multiple
* instances of this class for each summary file.
*******************************************************************************/

[Description("Monitor information from the summary log file. This information is extracted from the EDID data in the registry")]
Dynamic, Provider("DPOProv")

class DPO_Monitor
{
    [Description("Unique ID from the summary file.")],
    Key
    ] string HardwareInfoGUID;

    [Description("Monitor index number, starting from 0.")],
    Key
    ] sint16 Index;

    void DeleteInstance ();
};
```
[Description("Type of monitor (Dell or Non-Dell.")
string Monitor_Type;

[Description("Model name of the monitor.")
string Model_Name;

[Description("Serial number of the monitor.")
string Serial;

[Description("Any vendor specific monitor data.")
string Vendor_Specific_Data;
]

};

/**************************************************************/
* DPO_HardwareInfoToMonitor
* This class associates DPO_Monitor instance(s) with an
* instance of DPO_HardwareInfo.
**************************************************************/
[Association : ToInstance,
  Description("This class associates DPO_Monitor instance(s) 
  with an instance of DPO_HardwareInfo.")]
dynamic:ToInstance,
  PROVIDER("DPOProv"):ToInstance
]
class DPO_HardwareInfoToMonitor
{
  [key] DPO_HardwareInfo REF Antecedent;
  [key] DPO_Monitor REF Dependent;
};

/**************************************************************/
* DPO_BiosInternalLogs
* This has the BIOS logs information from a summary log. There
* may be multiple instances of this class for each summary file.
**************************************************************/
[Description("BIOS logs: All system logs such as BIOS, Diagnostics, IPMI, SMBIOS, SPD logs etc."),
  Dynamic.Provider("DPOProv")]
class DPO_BiosInternalLogs
{
  [Key]
    string HardwareInfoGUID;
    string Name;
    string Time;
    string LogType;
    string EventCode;
    string Descr;
};

/**************************************************************/
* DPO_HardwareInfoToBiosInternalLogs
* This class associates DPO_HardwareInfoToBiosInternalLogs
* instance(s) with an instance of DPO_HardwareInfo.
**************************************************************/
 Association : ToInstance,
   Description("This class associates DPO_HardwareInfoToBiosInternalLogs
   " instance(s) with an instance of DPO_HardwareInfo.").
   dynamic:ToInstance,
   PROVIDER("DPOProv"):ToInstance
 ]
class DPO_HardwareInfoToBiosInternalLogs
 |
 [ [key] DPO_HardwareInfo REF Antecedent;
   [key] DPO_BiosInternalLogs REF Dependent;
 ];

*****************************************************************************
* DPO_WWAN
* This has the Wireless WAN adapter information from a summary
* log. There may be multiple instances of this class for each
* summary file.
*****************************************************************************
[Description("Wireless WAN adapter information."),
 Dynamic,Provider("DPOProv") ]
class DPO_WWAN
 |
 [ Description("Unique ID from the summary file.").
   Key
 ] string HardwareInfoGUID;

   [Description("Device name.")]
   string Device_Name;

   [Description("IMEI number.")]
   string IMEI;
 ];

*****************************************************************************
* DPO_HardwareInfoToWWAN
* This class associates DPO_WWAN instance(s) with an
* instance of DPO_HardwareInfo.
*****************************************************************************
[ Association : ToInstance,
   Description("This class associates DPO_WWAN instance(s) with "
   " an instance of DPO_HardwareInfo.").
   dynamic:ToInstance,
   PROVIDER("DPOProv"):ToInstance
 ]
class DPO_HardwareInfoToWWAN
 |
 [ [key] DPO_HardwareInfo REF Antecedent;
   [key] DPO_WWAN REF Dependent;
 ];

*****************************************************************************
* DPO_Battery
* This has the battery information from a summary log. There
* may be multiple instances of this class for each summary file.
*****************************************************************************
[Description("Battery information including charge, discharge and dwell statistics.").
 Dynamic,Provider("DPOProv") ]
class DPO_Battery
 |
 [ Description("Unique ID from the summary file.").
   Key
 ]
string HardwareInfoGUID;

[Description("Index number of the battery device starting from 1.")]
Key sint16 Index;

[Description("Manufacture date.")]
string Manufacture_Date;

[Description("Serial number.")]
string Serial_Number;

[Description("Chemistry.")]
string Chemistry;

[Description("Design Capacity in mAH.")]
string Design_Capacity_mAH;

[Description("Battery name.")]
string Name;

[Description("Manufacturer's name.")]
string Mfg_Name;

[Description("Full charge capacity of the battery.")]
sint32 FullChargeCapacity;

[Description("Battery cycle count.")]
sint16 Cycle_Count;

[Description("Total time (in minutes) the battery was in discharge state.")]
real32 Discharge_Time_mins;

[Description("Number of times the discharge depth was between 0 to 5%.")]
sint16 Discharge_Depth_0_5_Pct;

[Description("Number of times the discharge depth was between 5 to 10%.")]
sint16 Discharge_Depth_5_10_Pct;

[Description("Number of times the discharge depth was between 10 to 20%.")]
sint16 Discharge_Depth_10_20_Pct;

[Description("Number of times the discharge depth was between 20 to 40%.")]
sint16 Discharge_Depth_20_40_Pct;

[Description("Number of times the discharge depth was between 40 to 60%.")]
sint16 Discharge_Depth_40_60_Pct;

[Description("Number of times the discharge depth was between 60 to 80%.")]
sint16 Discharge_Depth_60_80_Pct;

[Description("Number of times the discharge depth was between 80 to 100%.")]
sint16 Discharge_Depth_80_100_Pct;

[Description("Number of times the start of discharge was between 100 to 94%.")]
//NameChange sint16 Discharge_Start_Point_100_94_Pct;
//NameChange sint16 Discharge_StartPt_GT_94_Pct;

[Description("Number of times the start of discharge was between 94 to 70%.")]
//NameChange sint16 Discharge_Start_Point_94_70_Pct;
//NameChange sint16 Discharge_StartPt_94_70_Pct;

[Description("Number of times the start of discharge was between 70 to 50%.")]
//NameChange sint16 Discharge_Start_Point_70_50_Pct;
//NameChange sint16 Discharge_StartPt_70_50_Pct;
sint16 Discharge_Start_Point_50_30_Pct;
  Discharge_StartPt_50_30_Pct;

// NameChange sint16
sint16 Discharge_Start_Point_30_10_Pct;
  Discharge_StartPt_30_10_Pct;

// NameChange sint16
sint16 Discharge_Start_Point_0_10_Pct;
  Discharge_StartPt_0_10_Pct;

// NameChange sint16
sint16 Discharge_Sessions_With_End_10_15;
  Discharge_Sess_End_10_15;

// NameChange sint16
sint16 Discharge_Sessions_With_End_5_10;
  Discharge_Sess_End_5_10;

// NameChange sint16
sint16 Discharge_Sessions_With_End_LT_5;
  Discharge_Sess_End_LT_5;

real32 Discharge_Temp_Avg;

// NameChange real32
real32 Discharge_Temp_Std_Dev;

// NameChange real32
real32 Discharge_Temp_Max;

// NameChange real32
real32 Discharge_Temp_Min;

real32 Discharge_mA_Avg;

// NameChange real32
real32 Discharge_mA_Std_Dev;

// NameChange real32
real32 Discharge_mA_Max;

// NameChange real32
real32 Discharge_mA_Min;

real32 Discharge_mV_Avg;

// NameChange real32
real32 Discharge_mV_Std_Dev;

// NameChange real32
real32 Discharge_mV_Max;

// NameChange real32
real32 Discharge_mV_Min;

real32 Discharge_Power_W_Avg;

// NameChange real32
real32 Discharge_Power_W_Std_Dev;

// NameChange real32
real32 Discharge_Power_W_Max;

// NameChange real32
real32 Discharge_Power_W_Min;
real32  Discharge_Power_W_Std_Dev;

sint32  Discharge_Power_W_Max;

sint32  Discharge_Power_W_Min;

sint16  Discharge_Power_0_5W_Pct;

sint16  Discharge_Power_5_10W_Pct;

sint16  Discharge_Power_10_15W_Pct;

sint16  Discharge_Power_15_20W_Pct;

sint16  Discharge_Power_20_25W_Pct;

sint16  Discharge_Power_25_30W_Pct;

sint16  Discharge_Power_30_40W_Pct;

sint16  Discharge_Power_40_50W_Pct;

sint16  Discharge_Power_50_60W_Pct;

sint16  Discharge_Power_GT60W_Pct;

real32  Charge_Time_mins;

sint16  Charge_Number_Full_Charge_Sessions;

sint16  Num_Full_Charge_Sessions;

sint16  Charge_Number_Partial_Charge_Sessions;

sint16  Num_Partial_Charge_Sessions;

real32  Charge_Temp_Avg;

real32  Charge_Temp_Std_Dev;

sint16  Charge_Temp_Max;

sint16  Charge_Temp_Min;

real32  Charge_mA_Avg;

real32  Charge_mA_Std_Dev;


```c
sint32 Charge_mA_Max;
sint32 Charge_mA_Min;
real32 Charge_mV_Avg;
real32 Charge_mV_Std_Dev;
sint32 Charge_mV_Max;
sint32 Charge_mV_Min;
real32 Charge_Power_W_RSOC_LE_60_Avg;
real32 Charge_Power_W_RSOC_LE_60_Min;
real32 Charge_Power_W_RSOC_LE_60_Std_Dev;
real32 Charge_Power_W_RSOC_LE_60_Max;
sint16 Charge_Pwr_RSOC_LE_60_Max;
sint16 Charge_Pwr_RSOC_LE_60_Min;
real32 Charge_Pwr_RSOC_LE_60_Std_Dv;
real32 Charge_Pwr_RSOC_LE_60_Avg;
real32 Dwell_Temp_Avg;
real32 Dwell_Temp_Std_Dev;
sint32 Dwell_Time_mins;
real32 Dwell_Avg_RSOC_Level;
real32 Dwell_Temp_Avg;
sint16 Dwell_Temp_Max;
```
42 Dell Precision Optimizer Administrator’s Guide

--------

```c
#include "Dell_HardwareInfoToBattery.h"

/* DPO_HardwareInfoToBattery */

/* This class associates DPO_Battery instance(s) with an */
/* instance of DPO_HardwareInfo. */

class DPO_HardwareInfoToBattery {

    [key]  DPO_HardwareInfo  REF  Antecedent;
    [key]  DPO_Battery       REF  Dependent;
};

/* DPO_NBFan */

/* This has the notebook fan information from a summary log. There */
/* may be multiple instances of this class for each summary file. */

class DPO_NBFan {

    [Key] string HardwareInfoGUID;
    [Key] sint16 Index;
    [Key] string Location;
    sint16 Fan_Duty_Cycle_Pct;
    sint32 RPM;
    sint32 Peak_Fan_RPM;
    real32 Average_Fan_RPM;
    sint16 RPM_0_Pct;
    sint16 RPM_0_1000_Pct;
    sint16 RPM_1000_1700_Pct;
    sint16 RPM_1700_2200_Pct;
};
```

---
sint16 RPM_1700_2200_Pct;

[Description("Percentage of time the fan speed was between 2200 and 2600 RPMs.")] sint16 RPM_2200_2600_Pct;

[Description("Percentage of time the fan speed was between 2600 and 2900 RPMs.")] sint16 RPM_2600_2900_Pct;

[Description("Percentage of time the fan speed was between 2900 and 3100 RPMs.")] sint16 RPM_2900_3100_Pct;

[Description("Percentage of time the fan speed was between 3100 and 3300 RPMs.")] sint16 RPM_3100_3300_Pct;

[Description("Percentage of time the fan speed was between 3300 and 3600 RPMs.")] sint16 RPM_3300_3600_Pct;

[Description("Percentage of time the fan speed was between 3600 and 3900 RPMs.")] sint16 RPM_3600_3900_Pct;

[Description("Percentage of time the fan speed was between 3900 and 4200 RPMs.")] sint16 RPM_3900_4200_Pct;

[Description("Percentage of time the fan speed was between 4200 and 4600 RPMs.")] sint16 RPM_4200_4600_Pct;

[Description("Percentage of time the fan speed was between 4600 and 5100 RPMs.")] sint16 RPM_4600_5100_Pct;

[Description("Percentage of time the fan speed was between 5100 and 5600 RPMs.")] sint16 RPM_5100_5600_Pct;

[Description("Percentage of time the fan speed was between 5600 and 6200 RPMs.")] sint16 RPM_5600_6200_Pct;

[Description("Percentage of time the fan speed was between 6200 and 7000 RPMs.")] sint16 RPM_6200_7000_Pct;

[Description("Percentage of time the fan speed was more than 7000 RPMs.")] sint16 RPM_GT7000_Pct;

};

/*************************************************************/
* DPO_HardwareInfoToNBFan
* This class associates DPO_HardwareInfo instance(s) with an
* instance of DPO_NBFan.
*************************************************************/
class DPO_HardwareInfoToNBFan
{
  [key] DPO_HardwareInfo REF Antecedent;
  [key] DPO_NBFan REF Dependent;
};

/*****************************/
* DPO_DTFan
* This has the desktop fan information from a summary log. There
* may be multiple instances of this class for each summary file.
*****************************/
[Description("Desktop fan speed statistics.")],

Dell Precision Optimizer Administrator’s Guide
Dynamic,Provider("DPOProv") ]
class DPO_DTFan
{
    ["Unique ID from the summary file."],
    Key
    ] string HardwareInfoGUID;

    ["Desktop fan index number starting from 0.",
    Key
    ] sint16 Index;

    ["Location where the fan is present in the system."]]
    string Location;

    ["Percentage of time fan rpm was non-zero."]]
    sint16 Fan_Duty_Cycle_Pct;

    ["Fan speed when the summary log was generated."]]
    sint32 RPM;

    ["Peak fan speed."]]
    sint32 Peak_Fan_RPM;

    ["Average fan speed."]]
    real32 Average_Fan_RPM;

    ["Percentage of time the fan speed was between 0 and 500 RPMs."]]
    sint16 RPM_0_500_Pct;

    ["Percentage of time the fan speed was between 500 and 900 RPMs."]]
    sint16 RPM_500_900_Pct;

    ["Percentage of time the fan speed was between 900 and 1100 RPMs."]]
    sint16 RPM_900_1100_Pct;

    ["Percentage of time the fan speed was between 1100 and 1300 RPMs."]]
    sint16 RPM_1100_1300_Pct;

    ["Percentage of time the fan speed was between 1300 and 1600 RPMs."]]
    sint16 RPM_1300_1600_Pct;

    ["Percentage of time the fan speed was between 1600 and 1900 RPMs."]]
    sint16 RPM_1600_1900_Pct;

    ["Percentage of time the fan speed was between 1900 and 2300 RPMs."]]
    sint16 RPM_1900_2300_Pct;

    ["Percentage of time the fan speed was between 2300 and 2700 RPMs."]]
    sint16 RPM_2300_2700_Pct;

    ["Percentage of time the fan speed was between 2700 and 3100 RPMs."]]
    sint16 RPM_2700_3100_Pct;

    ["Percentage of time the fan speed was between 3100 and 3500 RPMs."]]
    sint16 RPM_3100_3500_Pct;

    ["Percentage of time the fan speed was between 3500 and 4000 RPMs."]]
    sint16 RPM_3500_4000_Pct;

    ["Percentage of time the fan speed was between 4000 and 4500 RPMs."]]
    sint16 RPM_4000_4500_Pct;

    ["Percentage of time the fan speed was between 4500 and 5000 RPMs."]]
    sint16 RPM_4500_5000_Pct;
```csharp
{sint16 RPM_5000_5500_Pct;

}sint16 RPM_5500_6000_Pct;

}sint16 RPM_GT6000_Pct;

};

/* DPO_HardwareInfoToDTFan
 * This class associates DPO_DTFan instance(s) with an instance of DPO_HardwareInfo.
 ************************************************************************************/
class DPO_HardwareInfoToDTFan
{
    [key] DPO_HardwareInfo REF Antecedent;
    [key] DPO_DTFan REF Dependent;
};

/* DPO_Thermistor
 * This has the thermal information from a summary log. There may be multiple instances of this class for each summary file.
 ************************************************************************************/
class DPO_Thermistor
{
    [Description("Unique ID from the summary file.")] string HardwareInfoGUID;

    [/Description("Thermistor index number starting from 0.")] sint16 Index;

    [Description("Thermistor location eg CPU, Memory etc.")] string Location;

    [Description("Temperature read from the thermistor when the summary log was generated.")] sint16 Temp;

    [Description("Maximum temperature read from the thermistor.")] sint16 Peak_Temp;

    [Description("Average temperature read from the thermistor.")] real32 Avg_Temp;

    [Description("Minimum temperature read from the thermistor.")] sint16 Min_Temp;

    [Description("Standard deviation of temperature read from the thermistor.")] real32 Std_Dev_Temp;

    [Description("Percentage of time the temperature read was between 0 to 30C.")]}
```
sint16  Temp_0_30C_Pct;

[Description("Percentage of time the temperature read was between 30 to 40C.")]  
sint16  Temp_30_40C_Pct;

[Description("Percentage of time the temperature read was between 40 to 50C.")]  
sint16  Temp_40_50C_Pct;

[Description("Percentage of time the temperature read was between 50 to 60C.")]  
sint16  Temp_50_60C_Pct;

[Description("Percentage of time the temperature read was between 60 to 70C.")]  
sint16  Temp_60_70C_Pct;

[Description("Percentage of time the temperature read was between 70 to 80C.")]  
sint16  Temp_70_80C_Pct;

[Description("Percentage of time the temperature read was between 80 to 90C.")]  
sint16  Temp_80_90C_Pct;

[Description("Percentage of time the temperature read was between 90 to 100C.")]  
sint16  Temp_90_100C_Pct;

[Description("Percentage of time the temperature read was more than 100C.")]  
sint16  Temp_GT100C_Pct;

};

/**************************************************************/
/* DPO_HardwareInfoToThermistor */
/* This class associates DPO_Thermistor instance(s) with an */
/* instance of DPO_HardwareInfo. */
/**************************************************************/

Association : ToInstance,
Description("This class associates DPO_Thermistor instance(s) 
" " with an instance of DPO_HardwareInfo"),
dynamic:ToInstance,
PROVIDER("DPOProv"):ToInstance
]
class DPO_HardwareInfoToThermistor
{
  [key] DPO_HardwareInfo REF  Antecedent;
  [key] DPO_Thermistor  REF  Dependent;
};

/**************************************************************/
/* DPO_Logical_Processor */
/* This has the logical processor information from a summary log. */
/* There may be multiple instances of this class for each summary */
/* file. */
/**************************************************************/

[Description("Logical processors statistics.")],
Dynamic,Provider("DPOProv")
]  
class DPO_Logical_Processor
{
  [  
    Description("Unique ID from the summary file.").
    Key
  ]
  string  HardwareInfoGUID;

  [  
    Description("Index of logical processor starting from 0.").
    Key
  ]
  sint16  Index;
}
```csharp
// DPO_HardwareInfoToLogical_Processor
// This class associates DPO_Logical_Processor instance(s) with an instance of DPO_HardwareInfo.

public class DPO_HardwareInfoToLogical_Processor
{
    public DPO_HardwareInfo HardwareInfo;
    public List<DPO_Logical_Processor> LogicalProcessors;

    // Constructor
    public DPO_HardwareInfoToLogical_Processor(DPO_HardwareInfo hardwareInfo)
    {
        HardwareInfo = hardwareInfo;
        LogicalProcessors = new List<DPO_Logical_Processor>();
    }

    // Method to add a logical processor
    public void AddLogicalProcessor(DPO_Logical_Processor processor)
    {
        LogicalProcessors.Add(processor);
    }

    // Method to remove a logical processor
    public void RemoveLogicalProcessor(DPO_Logical_Processor processor)
    {
        LogicalProcessors.Remove(processor);
    }

    // Method to get the logical processor list
    public List<DPO_Logical_Processor> GetLogicalProcessors()
    {
        return LogicalProcessors;
    }

    // Method to get the hardware info
    public DPO_HardwareInfo GetHardwareInfo()
    {
        return HardwareInfo;
    }
}
```

```c++

};//

/****************************
* DPO_HardwareInfoToDisk
* This class associates DPO_Disk instance(s) with an
* instance of DPO_HardwareInfo.
**************************************************************/
[Association : ToInstance,
Description("This class associates DPO_Disk instance(s) with 
an instance of DPO_HardwareInfo"),
dynamic:ToInstance,
PROVIDER("DPOProv"):ToInstance
]
class DPO_HardwareInfoToDisk
{
    [key] DPO_HardwareInfo REF Antecedent;
    [key] DPO_Disk REF Dependent;
};

/****************************
* DPO_Partition
* This has the logical partition information from a summary log.
* There may be multiple instances of this class for each summary
* file.
**************************************************************/
[Association : ToInstance,
Description("Information for each partition found on a disk."),
Dynamic, Provider("DPOProv")]
class DPO_Partition
{
    [Description("Unique ID from the summary file.")],
    Key
    string HardwareInfoGUID;

    [Description("Unique ID assigned to the physical disk instance to which this partition belongs.")],
    Key
    string DiskGUID;

    [Description("Partition index number starting from 0.")],
    Key
    sint16 Index;

    [Description("Partition name, eg C:.")]
    string Name;

    [Description("Total size of the partition in MBs.")]
    sint32 Size_MB;
};

/****************************
* DPO_DiskToPartition
* This class associates DPO_Partition instance(s) with an
* instance of DPO_Disk.
**************************************************************/
```
class DPO_DiskToPartition {
    [key] DPO_Disk REF Antecedent;
    [key] DPO_Partition REF Dependent;
};

class DPO_LanAdapter {
    [Description("LAN adapter information and statistics.")]
    Dynamic, Provider("DPOProv")
    [Key]
    string HardwareInfoGUID;
    [Key]
    sint16 Index;
    [Description("LAN adapter name.")]
    string Name;
    [Description("LAN adapter's MAC address.")]
    string MAC;
    [Description("Percentage of time the adapter was busy when the system was on AC.")]
    sint16 ActivityAC_Pct;
    [Description("Percentage of time the adapter was busy when the system was on battery.")]
    sint16 ActivityDC_Pct;
};

class DPO_HardwareInfoToLanAdapter {
    [key] DPO_HardwareInfo REF Antecedent;
    [key] DPO_LanAdapter REF Dependent;
};

class DPO_HardwareInfoToLanAdapter {
    [key] DPO_HardwareInfo REF Antecedent;
    [key] DPO_LanAdapter REF Dependent;
};

class DPO_WlanAdapter
This has the wlan adapter information from a summary log. There
may be multiple instances of this class for each summary file.

[Description("Wireless LAN adapter information and statistics.")],
Dynamic,Provider("DPOProv")

class DPO_WlanAdapter
{
    [Description("Unique ID from the summary file.")],
    Key
    string HardwareInfoGUID;

    [Description("Wireless LAN adapter index number starting from 0.")],
    Key
    sint16 Index;

    [Description("Wireless LAN adapter name.")]
    string Name;

    [Description("Wireless LAN adapter's MAC address.")]
    string MAC;

    [Description("Percentage of time the radio was off when the system was on AC.")]
    sint16 WlanRadioOffAC_Pct;

    [Description("Percentage of time the WLAN adapter was connected when the system was on AC.")]
    sint16 WlanConnectedAC_Pct;

    [Description("Percentage of time the adapter was not connected when the system was on AC.")]
    sint16 WlanDisconnectedAC_Pct;

    [Description("Percentage of time the radio was off when the system was on battery.")]
    sint16 WlanRadioOffDC_Pct;

    [Description("Percentage of time the WLAN adapter was connected when the system was on battery.")]
    sint16 WlanConnectedDC_Pct;

    [Description("Percentage of time the adapter was not connected when the system was on battery.")]
    sint16 WlanDisconnectedDC_Pct;
};

******************************************************************************

*DPO_HardwareInfoToWlanAdapter
* This class associates DPO_WlanAdapter instance(s) with an
* instance of DPO_HardwareInfo.
******************************************************************************

[Association : ToInstance,
Description("This class associates DPO_WlanAdapter instance(s) 
" " with an instance of DPO_HardwareInfo").
Dynamic,ToInstance,
PROVIDER("DPOProv"):ToInstance
]
class DPO_HardwareInfoToWlanAdapter
{
[key] DPO_HardwareInfo REF Antecedent;
[key] DPO_WlanAdapter REF Dependent;
};

******************************************************************************

*DPO_Smart
* This has the SMART information from a summary log. There
* may be multiple instances of this class for each summary file.
******************************************************************************
class DPO_Smart {

    [Description("Unique ID from the summary file.")]
    Key string HardwareInfoGUID;

    [Description("Smart data index number starting from 0.")]
    Key sint16 Index;

    [Description("Name eg, SMART0.")]
    string Name;

    [Description("Disk Model number.")]
    string Model;

    [Description("Average disk temperature read using SMART.")]
    real32 Temp_Avg;

    [Description("Standard deviation of disk temperature read using SMART.")]
    real32 Temp_Std_Dev;

    [Description("Minimum disk temperature read using SMART.")]
    sint16 Temp_Min;

    [Description("Maximum disk temperature read using SMART.")]
    sint16 Temp_Max;

    [Description("Percentage of time disk temperature read using SMART was between 0 to 30C.")]
    sint16 Temp_0_30_Pct;

    [Description("Percentage of time disk temperature read using SMART was between 30 to 40C.")]
    sint16 Temp_30_40_Pct;

    [Description("Percentage of time disk temperature read using SMART was between 40 to 50C.")]
    sint16 Temp_40_50_Pct;

    [Description("Percentage of time disk temperature read using SMART was between 50 to 60C.")]
    sint16 Temp_50_60_Pct;

    [Description("Percentage of time disk temperature read using SMART was between 60 to 70C.")]
    sint16 Temp_60_70_Pct;

    [Description("Percentage of time disk temperature read using SMART was between 70 to 80C.")]
    sint16 Temp_70_80_Pct;

    [Description("Percentage of time disk temperature read using SMART was more than 80C.")]
    sint16 Temp_GT_80_Pct;

    [Description("Shock events.")]
    sint32 Shock_Events;

    [Description("Shock events (normalized value).")]
    uint8 Shock_Events_Normalized;

    [Description("Shock events (worst value).")]
    uint8 Shock_Events_Worst;

    [Description("Shock events (threshold value).")]
    uint8 Shock_Events_Threshold;
}
**Description:** Total blocks read from the disk.
`sint64` Blks_Read;

**Description:** Total blocks read from the disk (normalized value).
`uint8` Blks_Read_Normalized;

**Description:** Total blocks read from the disk (worst value).
`uint8` Blks_Read_Worst;

**Description:** Total blocks read from the disk (threshold value).
`uint8` Blks_Read_Threshold;

**Description:** Total blocks written to the disk.
`sint64` Blks_Written;

**Description:** Total blocks written to the disk (normalized value).
`uint8` Blks_Written_Normalized;

**Description:** Total blocks written to the disk (worst value).
`uint8` Blks_Written_Worst;

**Description:** Total blocks written to the disk (threshold value).
`uint8` Blks_Written_Threshold;

**Description:** Start stop count.
`sint64` Start_Stop_Count;

**Description:** Start stop count (normalized value).
`uint8` Start_Stop_Count_Normalized;

**Description:** Start stop count (worst value).
`uint8` Start_Stop_Count_Worst;

**Description:** Start stop count (threshold value).
`uint8` Start_Stop_Count_Threshold;

**Description:** Load unload cycle count.
`sint64` Load_Unload_Cycle_Count;

**Description:** Load unload cycle count (normalized value).
`uint8` Load_Unload_Cycle_Count_Normalized;

**Description:** Load unload cycle count (worst value).
`uint8` Load_Unload_Cycle_Count_Worst;

**Description:** Load unload cycle count (threshold value).
`uint8` Load_Unload_Cycle_Count_Threshold;

**Description:** Total power on hours.
`sint64` Power_On_Hours;

**Description:** Total power on hours (normalized value).
`uint8` Power_On_Hours_Normalized;

**Description:** Total power on hours (worst value).
`uint8` Power_On_Hours_Worst;

**Description:** Total power on hours (threshold value).
`uint8` Power_On_Hours_Threshold;

**Description:** Realloc sector count.
`sint64` ReAlloc_Sector_Count;

**Description:** Realloc sector count (normalized value).
`uint8` ReAlloc_Sector_Count_Normalized;
uint8 ReAlloc_Sector_Count_Worst;
uint8 ReAlloc_Sector_Count_Threshold;
sint64 Head_Flying_Hours;
uint8 Head_Flying_Hours_Normalized;
uint8 Head_Flying_Hours_Worst;
uint8 Head_Flying_Hours_Threshold;
sint64 Raw_Read_Error_Rate;
uint8 Raw_Read_Error_Rate_Normalized;
uint8 Raw_Read_Error_Rate_Worst;
uint8 Raw_Read_Error_Rate_Threshold;
sint64 Spin_Up_Time;
uint8 Spin_Up_Time_Normalized;
uint8 Spin_Up_Time_Worst;
uint8 Spin_Up_Time_Threshold;
sint64 Free_Fall_Count;
uint8 Free_Fall_Count_Normalized;
uint8 Free_Fall_Count_Worst;
uint8 Free_Fall_Count_Threshold;
sint64 Power_Cycle_Count;
uint8 Power_Cycle_Count_Normalized;
uint8 Power_Cycle_Count_Worst;
uint8 Power_Cycle_Count_Threshold;
Program fail count:
sint64 Program_Fail_Count;

Program fail count (normalized value):
uint8 Program_Fail_Count_Normalized;

Program fail count (worst value):
uint8 Program_Fail_Count_Worst;

Program fail count (threshold value):
uint8 Program_Fail_Count_Threshold;

Erase fail count:
sint64 Erase_Fail_Count;

Erase fail count (normalized value):
uint8 Erase_Fail_Count_Normalized;

Erase fail count (worst value):
uint8 Erase_Fail_Count_Worst;

Erase fail count (threshold value):
uint8 Erase_Fail_Count_Threshold;

Wear leveling count:
sint64 Wear_Leveling_Count;

Wear leveling count (normalized value):
uint8 Wear_Leveling_Count_Normalized;

Wear leveling count (worst value):
uint8 Wear_Leveling_Count_Worst;

Wear leveling count (threshold value):
uint8 Wear_Leveling_Count_Threshold;

User reserved block count:
sint64 User_Rsvd_Block_Count;

User reserved block count (normalized value):
uint8 User_Rsvd_Block_Count_Normalized;

User reserved block count (worst value):
uint8 User_Rsvd_Block_Count_Worst;

User reserved block count (threshold value):
uint8 User_Rsvd_Block_Count_Threshold;

User reserved block count (SSD Total):
sint64 User_Rsvd_Block_Count_Total;

User reserved block count (SSD Total) (normalized value):
uint8 User_Rsvd_Block_Count_Total_Normalized;

User reserved block count (SSD Total) (worst value):
uint8 User_Rsvd_Block_Count_Total_Worst;

User reserved block count (SSD Total) (threshold value):
uint8 User_Rsvd_Block_Count_Total_Threshold;

Unused reserved block count:
sint64 Unused_Rsvd_Block_Count;

Unused reserved block count (normalized value):
uint8 Unused_Rsvd_Block_Count_Normalized;
Unused_Rsvd_Block_Count_Worst;

Unused_Rsvd_Block_Count_Threshold;

Program_Fail_Count_Total;

Program_Fail_Count_Total_Normalized;

Program_Fail_Count_Total_Worst;

Program_Fail_Count_Total_Threshold;

Erase_Fail_Count_Total;

Erase_Fail_Count_Total_Normalized;

Erase_Fail_Count_Total_Worst;

Erase_Fail_Count_Total_Threshold;

Uncorrectable_Error_Count;

Uncorrectable_Error_Count_Normalized;

Uncorrectable_Error_Count_Worst;

Uncorrectable_Error_Count_Threshold;

Ecc_Rate;

Ecc_Rate_Normalized;

Ecc_Rate_Worst;

Ecc_Rate_Threshold;

DPO_HardwareInfoToSmart
* This class associates DPO_Smart instance(s) with an
* instance of DPO_HardwareInfo.
*********************************************************************

[Association : ToInstance,
Description("This class associates DPO_Smart instance(s) with" " an instance of DPO_HardwareInfo"),
dynamic:ToInstance,
PROVIDER("DPOProv"):ToInstance]
class DPO_HardwareInfoToSmart
{
  [key] DPO_HardwareInfo REF Antecedent;
  [key] DPO_Smart REF Dependent;
};

/*************************************************************/
* DPO_DIMM
* This has the DIMM information from a summary log. There
* may be multiple instances of this class for each summary file.
*************************************************************/
[Description("DIMM information for all DIMMs reported by the BIOS.")]
Dynamic,Provider("DPOProv")
class DPO_DIMM
{
  [Description("Unique ID from the summary file.")],
  Key
  string HardwareInfoGUID;

  [Description("DIMM index number starting from 0.")],
  Key
  sint16 Index;

  [Description("DIMM name.")]
  string Name;

  [Description("DIMM manufacturer's name.")]
  string Manufacturer;

  [Description("DIMM part number.")]
  string Part;

  [Description("DIMM location.")]
  string Location;

  [Description("DIMM serial number.")]
  string Serial;
};

/*************************************************************/
* DPO_HardwareInfoToDIMM
* This class associates DPO_DIMM instance(s) with an
* instance of DPO_HardwareInfo.
*************************************************************/
[Association : ToInstance,
  Description("This class associates DPO_DIMM instance(s) with 
  "an instance of DPO_HardwareInfo"),
  dynamic:ToInstance,
  PROVIDER("DPOProv"):ToInstance
]
class DPO_HardwareInfoToDIMM
{
  [key] DPO_HardwareInfo REF Antecedent;
  [key] DPO_DIMM REF Dependent;
};

/*************************************************************/
* DPO_Logical_Drive_Info_New
* This has the new logical drive information from a summary log. There
* may be multiple instances of this class for each summary file.
*************************************************************/
class DPO_Logical_Drive_Info_New {
    [Description("Logical drive information for all logical drives found on the system.")],
    Dynamic, Provider("DPOProv")
    string HardwareInfoGUID;
    Key
    sint16 Index;
    string Name;
    sint64 Size_MB;
    sint64 Freespace_MB;
};

class DPO_HardwareInfoToLogicalDriveInfoNew {
    [Association : ToInstance, Dynamic, Provider("DPOProv")]
    class DPO_HardwareInfoToLogicalDriveInfoNew {
        [key] DPO_HardwareInfo REF Antecedent;
        [key] DPO_Logical_Drive_Info_New REF Dependent;
    }
};

class DPO_CrashInfo {
    Dynamic, Provider("DPOProv")
    string HardwareInfoGUID;
    Key
    sint16 Index;
    string Name;
    sint64 Size_MB;
    sint64 Freespace_MB;
};
```csharp
sint16 Index;

[Description("Local Time stamp (with time zone) of the date/time the crash was generated.")] string BugCheck_Time;

[Description("Information string from Windows Event Log.")] string BugCheck_String;

[Description("MiniDump File Name.")] string Minidump_FileName;

[Description("MiniDump File Data Length")] uint32 Minidump_DataLen;

[Description("MiniDump File Binary Data")] uint8 Minidump_Data[];

[Description("Bug check stack frame 1")] string BugCheck_Stack1;

[Description("Bug check stack frame 2")] string BugCheck_Stack2;

[Description("Bug check stack frame 3")] string BugCheck_Stack3;

[Description("Bug check stack frame 4")] string BugCheck_Stack4;

[Description("Bug check stack frame 5")] string BugCheck_Stack5;

};

/**********************************************************
* DPO_HardwareInfoToCrashInfo
* This class associates DPO_CrashInfo instance(s) with an
* instance of DPO_HardwareInfo.
**********************************************************/

[Association: ToInstance,
 Description("This class associates DPO_CrashInfo instance(s) with "
 "an instance of DPO_HardwareInfo"),
 dynamic:ToInstance,
 PROVIDER("DPOProv"):ToInstance
]

class DPO_HardwareInfoToCrashInfo
{
  [key] DPO_HardwareInfo REF Antecedent;
  [key] DPO_CrashInfo REF Dependent;
};

/**********************************************************
* DPO_FreeFall
* This has Free fall information from a summary
* log. Right now, there is only one instance of this class for
* each summary file but that may change in the future.
**********************************************************/

[Description("Free fall information from the summary log file."),
 Dynamic.Provider("DPOProv") ]

class DPO_FreeFall
{
  [ Description("Unique ID from the summary file."),
    Key
  ]
```
```java
string HardwareInfoGUID;

    [Description("Number of times free fall condition was detected since last summary file was generated.")]
sint16 FreeFallCount;

};//

/************************************************************
 * DPO_HardwareInfoToFreeFall
 * This class associates DPO_FreeFall instance(s) with an
 * instance of DPO_HardwareInfo.
 ************************************************************/
[Association : ToInstance,
Description("This class associates DPO_FreeFall instance(s) with 
"an instance of DPO_HardwareInfo"),
dynamic:ToInstance,
PROVIDER("DPOProv"):ToInstance
]
class DPO_HardwareInfoToFreeFall
{
    [key] DPO_HardwareInfo REF Antecedent;
    [key] DPO_FreeFall REF Dependent;
};

/************************************************************
 * DPO_Cable
 * This has the cable log information from a summary log. There
 * may be multiple instances of this class for each summary file.
 ************************************************************/
[Description("Cable logs: List of all cables attached, required but not connected in the system."),
Dynamic,Provider("DPOProv") ]
class DPO_Cable
{
    [Description("Unique ID from the summary file.")],
    Key string HardwareInfoGUID;

    [Description("Cable index number, starting from 0.")],
    Key sint16 Index;

    [Description("Name of cable.")]
    string Name;

    [Description("Cable's connection status.")]
    string Status;
};

/************************************************************
 * DPO_HardwareInfoToCableLogs
 * This class associates DPO_HardwareInfoToCable
 * instance(s) with an instance of DPO_HardwareInfo.
 ************************************************************/
[Association : ToInstance,
Description("This class associates DPO_HardwareInfoToCable 
"instance(s) with an instance of DPO_HardwareInfo.") ,
dynamic:ToInstance,
PROVIDER("DPOProv"):ToInstance
]
class DPO_HardwareInfoToCable
{
[key] DPO_HardwareInfo REF Antecedent;
[key] DPO_Cable REF Dependent;

/*******************************************
* DPO_CableChangeHistory
* This has the cable change history information from a summary log.
* There may be multiple instances of this class for each summary
* file.
*******************************************/
[Description("Information for status change for a cable."),
Dynamic,Provider("DPOProv") ]
class DPO_CableChangeHistory
{
  [Description("Name of cable.")]
  string Name;

  [Description("Timestamp when the change in cable status was noted.")]
  string Timestamp;

  [Description("Cable's connection status.")]
  string Status;
};

/*******************************************
* DPO_CableToCableChangeHistory
* This class associates DPO_CableChangeHistory instance(s) with an
* instance of DPO_Cable.
*******************************************/
[Association : ToInstance,
Description(" This class associates DPO_CableChangeHistory instance(s) ",
" with an instance of DPO_Cable."),
dynamic:ToInstance,
PROVIDER("DPOProv"):ToInstance ]
class DPO_CableToCableChangeHistory
{
  [key] DPO_Cable REF Antecedent;
  [key] DPO_CableChangeHistory REF Dependent;
};

/*******************************************
* DPO_BTMODULE
* This has the bluetooth module information from a summary log. There
* may be multiple instances of this class for each summary file.
*******************************************/
[Description("Bluetooth module information and statistics.")],
Dynamic,Provider("DPOProv") ]
class DPO_BTModule
{
  [Description("Unique ID from the summary file.")],
  Key
  string HardwareInfoGUID;

  [Description("Bluetooth module index number starting from 0.")],
  Key
  sint16 Index;

  [Description("Bluetooth module name.")]
  string Name;
string Address;

sint16 BTRadioOnAC_Pct;

sint16 BTConnectedAC_Pct;

sint16 BTDconnectedAC_Pct;

sint16 BTRadioOnDC_Pct;

sint16 BTConnectedDC_Pct;

sint16 BTDconnectedDC_Pct;

};

/*************************************************************/
/* DPO_HardwareInfoToBTModule */
/* This class associates DPO_BTModule instance(s) with an */
/* instance of DPO_HardwareInfo. */
/*************************************************************/
[Association : ToInstance,
Description("This class associates DPO_BTModule instance(s) with an DPO_HardwareInfo.
"),
dynamic:ToInstance,
PROVIDER("DPOProv"):ToInstance]
class DPO_HardwareInfoToBTModule
{
[key] DPO_HardwareInfo REF Antecedent;
[key] DPO_BTModule REF Dependent;
};

/*************************************************************/
/* DPO_IntelPerf */
/* This has the Intel performance information from a summary log. There */
/* may be multiple instances of this class for each summary file. */
/*************************************************************/
[Description("Intel performance information and statistics.")],
Dynamic.Provider("DPOProv") ]
class DPO_IntelPerf
{
[ Description("Unique ID from the summary file.")],
Key
]
string HardwareInfoGUID;

[ Description("Processor number starting from 0.")],
Key
]
sint16 Index;

[ Description("Minimum active relative frequency of the processor.")]
real32 Min_ActiveRelativeFreq;

[ Description("Maximum active relative frequency of the processor.")]

real32 Max_ActiveRelativeFreq;
[Description("Average active relative frequency of the processor.")]
real32 Avg_ActiveRelativeFreq;
[Description("Percentage of time the processor was in turbo mode when the system was on AC.")]
real32 TurboResidencyACPct;
[Description("Percentage of time the processor was in turbo mode when the system was on battery.")]
real32 TurboResidencyDCPct;
};

/********************
*************************************************************/
* DPO_HardwareInfoToIntelPerf
* This class associates DPO_IntelPerf instance(s) with an
* instance of DPO_HardwareInfo.
*******************************************************************************/
[Association : ToInstance,
Description("This class associates DPO_IntelPerf instance(s) "
" with an instance of DPO_HardwareInfo"),
dynamic:ToInstance,
PROVIDER("DPOProv") : ToInstance]
class DPO_HardwareInfoToIntelPerf
{
[key] DPO_HardwareInfo REF Antecedent;
[key] DPO_IntelPerf REF Dependent;
};

/*******************************************************************************/
* DPO_Graphics
* This has the graphics information from a summary log. There
* may be multiple instances of this class for each summary file.
*******************************************************************************/
[Description("Graphics performance information and statistics.")],
Dynamic, Provider("DPOProv") ]
class DPO_Graphics
{
    [Description("Unique ID from the summary file.")],
    Key
    string HardwareInfoGUID;

    [Description("GPU number starting from 0.")],
    Key
    sint16 Index;

    [Description("Minimum GPU utilization.")]
    sint16 Min_GpuUtilization;

    [Description("Maximum GPU utilization.")]
    sint16 Max_GpuUtilization;

    [Description("Average GPU utilization.")]
    real32 Avg_GpuUtilization;

    [Description("Percentage of time GPU was at 0% utilization.")]
    real32 GpuUtilization_0_Pct;

    [Description("Minimum graphics memory utilization.")]
    sint16 Min_MemUtilization;

    [Description("Maximum graphics memory utilization.")]
}
```c
sint16 Max_MemUtilization;

[Description("Average graphics memory utilization.")] real32 Avg_MemUtilization;

[Description("Percentage of time graphics memory was at 0% utilization.")] real32 MemUtilization_0_Pct;

[Description("Minimum graphics engine utilization.")] sint16 Min_EngineUtilization;

[Description("Maximum graphics engine utilization.")] sint16 Max_EngineUtilization;

[Description("Average graphics engine utilization.")] real32 Avg_EngineUtilization;

[Description("Percentage of time graphics engine was at 0% utilization.")] real32 EngineUtilization_0_Pct;

[Description("Minimum graphics bus utilization.")] sint16 Min_BusUtilization;

[Description("Maximum graphics bus utilization.")] sint16 Max_BusUtilization;

[Description("Average graphics bus utilization.")] real32 Avg_BusUtilization;

[Description("Percentage of time graphics bus was at 0% utilization.")] real32 BusUtilization_0_Pct;

[Description("Minimum graphics fan speed. The fan speed is reported in percentage.")] sint16 Min_FanSpeedPct;

[Description("Maximum graphics fan speed. The fan speed is reported in percentage.")] sint16 Max_FanSpeedPct;

[Description("Average graphics fan speed. The fan speed is reported in percentage.")] real32 Avg_FanSpeedPct;

[Description("Percentage of time graphics fan was at 0% speed.")] real32 FanSpeedPct_0_Pct;

[Description("Minimum GPU temperature.")] sint16 Min_Temperature;

[Description("Maximum GPU temperature.")] sint16 Max_Temperature;

[Description("Average GPU temperature.")] real32 Avg_Temperature;
```

/****************************
* DPO_HardwareInfoToGraphics
* This class associates DPO_Graphics instance(s) with an
* instance of DPO_HardwareInfo.
******************************/

```c
[Association : ToInstance,
  Description("This class associates DPO_Graphics instance(s) with an instance of DPO_HardwareInfo.")]
dynamic:ToInstance,
PROVIDER("DPOProv"):ToInstance
]
class DPO_HardwareInfoToGraphics
```
Current DPO version
Features Enabled/Disabled (e.g., GUI control listed above)
Time of last Check for Profiles

Time of last System Update
Time of last Check for Updates
Profile trigger history (time, profile, policy)

/*********************************************
* DPO_Info
*********************************************
[Description("DPO Info"),
 Dynamic.Provider("DPOProv") ]
class DPO_Info
{
    [ Description("Product version"),
    key ]
    string ProductVersion;

    [ Description("Features enabled") ]
    int *

    [ Description("Date/Time of last check for system updates") ]
    string LastCheckForUpdateTime;

    [ Description("Date/Time of last system update") ]
    string LastSystemUpdateTime;

    [ Description("Date/Time of last check for updated profiles") ]
    string LastCheckForProfiles;
};

/*********************************************
* DPO_TriggeredProfiles
*********************************************
[Description("DPO Profiles that have triggered"),
 Dynamic.Provider("DPOProv") ]
class DPO_TriggeredProfiles
{
[  
  Description("Unique ID of profile"),
  Key
]  
string ProfileGUID;

[  
  Description("Name of profile")
]  
string ProfileName;

[  
  Description("Unique ID of policy that triggered"),
  Key
]  
string PolicyGUID;

[  
  Description("Name of policy that triggered")
]  
string PolicyName;

[  
  Description("Date/Time of trigger"),
  key
]  
string TriggeredAt;

};

/*****************************/
// DPO_Profiles
/*****************************/
[Description("DPO Profiles"),
  Dynamic, Provider("DPOProv") ]
class DPO_Profiles
{
  [  
    Description("Unique ID"),
    Key
   ]
  string ProfileGUID;

  [  
    Description("Name")
  ]  
  string ProfileName;

  [  
    Description("Active")
  ]  
  string Active;

};

/*****************************/
// DPO_SmartAlerts
/*****************************/
[Description("DPO Smart Alerts"),
  Dynamic, Provider("DPOProv") ]
class DPO_SmartAlerts
{
  [  
    Description("Unique ID of alert"),
    Key
   ]
}
string AlertGUID;

[  Description("Alert Message")
]
string AlertMessage;

[  Description("Alert Description"),
]
string AlertDescr;

[  Description("Guidance"),
]
string AlertGuidance;

[  Description("Local date/time of alert"),
]
string AlertGeneratedAt;

};

/**************************************************************
* Creat an instance of the provider
// Setting the HostingModel to Decoupled:Com registers the provider as a decoupled com provider,
// lowers RPC_C_IMP_LEVEL_IMPERSONATE and RPC_C_IMP_LEVEL_DELEGATE impersonation levels to
// RPC_C_IMP_LEVEL_IDENTIFY before calling into provider:

// Setting the HostingModel to Decoupled:Com;FoldIdentity(FALSE} allows original client
// impersonation level through to provider.
// This lets a decoupled provider impersonate the client and hence
// act in the role of that client. This poses a potential security risk for the client
// if the decoupled provider security identity has less rights than the original client.
// Use a strong security descriptor when using this option:
**************************************************************/

instance of __Win32Provider as $P
{
  Clsid = "{C4ABD5F1-1260-4192-BF0B-11909C172043}";
   Name = "DPOProv";
   HostingModel = "NetworkServiceHost";
};

instance of __InstanceProviderRegistration
{
   Provider = $P;
   SupportsGet = TRUE;
   SupportsPut = FALSE;
   SupportsDelete = FALSE;
   SupportsEnumeration = TRUE;

   // we want WMI to do query parsing
   QuerySupportLevels = NULL;
};

instance of __MethodProviderRegistration
{
   Provider = $P;
};