Exemple : MDT avec SQL

Dans mon lab, j'ai déployé un MDT avec DFS pour les applications. Voici l'exemple complet :

Fichier customsettings.ini

```
[Settings]
Priority=DefaultGateway, TaskSequenceID, Default, MAKE, CSettings
Properties=TargetGroup

[Default]
OSInstall=Y

;Computer Details
SkipComputerName=YES
SkipDomainMembership=YES
JoinDomain=ad.khroners.fr
DomainAdmin=mdt_admin
DomainAdminDomain=ad.khroners.fr
DomainAdminPassword=Motdepasse

;Skip Task Sequence
SkipTaskSequence=NO
;TaskSequenceID=W10PRO-21H2-X64

;Drivers
DriverGroup001=WinPE x64\%Make%

;WSUS
TargetGroup=MDT

;User Data
SkipUserData=YES
```

```
;Computer Backup
SkipComputerBackup=YES

;Product Key
SkipProductKey=YES

;Language Packs
SkipPackageDisplay=YES

;Locale and Time
SkipLocaleSelection=YES
SkipTimeZone=YES
KeyboardLocale=040c:0000040c
KeyboardLocalePE=040c:0000040c
UserLocale=fr-FR
UILanguage=fr-FR
TimeZoneName=Romance Standard Time

;Roles and Features
SkipRoles=YES

;Applications
SkipApplications=YES

;Administrator Password
SkipAdminPassword=YES
AdminPassword=local-admin-password

;Local Administrators
SkipAdminAccounts=YES

;Capture Image
SkipCapture=YES

;Bitlocker
SkipBitLocker=YES

;Ready to begin
SkipSummary=YES
```

```
;Operating system deployment completed successfully
SkipFinalSummary=YES

FinishAction=REBOOT
WSUSServer=http://RN-SRV-WS-AAD01.ad.khroners.fr:8530
EventService=http://RN-SRV-WDS01:9800

[DefaultGateway]
10.29.0.254=Brest
10.35.100.254=Rennes

[Brest]
SLShare=\\BR-SRV-WDS01.ad.khroners.fr\DeploymentShare$\Logs
EventService=http://BR-SRV-WDS01:9800

[Rennes]
SLShare=\\RN-SRV-WDS01.ad.khroners.fr\DeploymentShare$\Logs
EventService=http://RN-SRV-WDS01:9800

[W10-22H2-ADM-35]
MachineObjectOU=OU=Administratifs,OU=Workstations,OU=RENNES,OU=Sites,DC=ad,DC=khroners,DC=fr

[W10-22H2-FOR-35]
MachineObjectOU=OU=Formateurs,OU=Workstations,OU=RENNES,OU=Sites,DC=ad,DC=khroners,DC=fr

[CSettings]
SQLServer=RN-SRV-WDS01.ad.khroners.fr
Instance=SQLEXPRESS
Database=MDT
Netlib=DBMSSOCN
DBID=sql_mdt
DBPWD=Motdepasse
Table=ComputerSettings
Parameters=UUID, AssetTag, SerialNumber, MacAddress
ParameterCondition=OR
```

Fichier bootstrap.ini

```
[Settings]
Priority=DefaultGateway, Default

[DefaultGateway]
10.29.0.254=Brest
10.35.100.254=Rennes

[Brest]
DeployRoot=\\BR-SRV-WDS01\DeploymentShare$

[Rennes]
DeployRoot=\\RN-SRV-WDS01\DeploymentShare$

[Default]
UserDomain=AD
UserID=mdt_admin
UserPassword=Motdepasse
SkipBDDWelcome=YES
```

Scripts

DeployWiz_SelectTS.vbs

J'ai modifié ce script pour utiliser la property "TaskSequenceID" dans le customsettings.ini.

```
' // ***************************************************************************
' //
' // Copyright (c) Microsoft Corporation.  All rights reserved.
' //
' // Microsoft Deployment Toolkit Solution Accelerator
' //
' // File:      DeployWiz_Initialization.vbs
' //
' // Version:   6.3.8456.1000
' //
' // Purpose:   Main Client Deployment Wizard Initialization routines
' //
```

```
' // ***************************************************************************


Option Explicit


''''''''''''''''''''''''''''''''''''''
'  Image List
'

Dim g_AllOperatingSystems

Function AllOperatingSystems


	Dim oOSes

	If isempty(g_AllOperatingSystems) then
	
		set oOSes = new ConfigFile
		oOSes.sFileType = "OperatingSystems"
		oOSes.bMustSucceed = false
		
		set g_AllOperatingSystems = oOSes.FindAllItems
		
	End if

	set AllOperatingSystems = g_AllOperatingSystems

End function


Function InitializeTSList
	Dim oItem, sXPathOld
	
	If oEnvironment.Item("TaskSequenceID") <> "" and oProperties("TSGuid") = "" then
		
		sXPathOld = oTaskSequences.xPathFilter
		for each oItem in oTaskSequences.oControlFile.SelectNodes( "/*/*[ID = '" &
```

```
oEnvironment.Item("TaskSequenceID")&"']")
□□□oLogging.CreateEntry "TSGuid changed via TaskSequenceID = " &
oEnvironment.Item("TaskSequenceID"), LogTypeInfo
□□□oEnvironment.Item("TSGuid") = oItem.Attributes.getNamedItem("guid").value
□□□exit for
□□next
□□
□□oTaskSequences.xPathFilter = sXPathOld
□□
□End if

□TSListBox.InnerHTML = oTaskSequences.GetHTMLEx ( "Radio", "TSGuid" )
□
□PopulateElements
□TSItemChange

End function


Function TSItemChange

□Dim oInput
□ButtonNext.Disabled = TRUE
□
□for each oInput in document.getElementsByName("TSGuid")
□□If oInput.Checked then
□□□oLogging.CreateEntry "Found CHecked Item: " & oInput.Value, LogTypeVerbose
□□
□□□ButtonNext.Disabled = FALSE
□□□exit function
□□End if
□next

End function


''''''''''''''''''''''''''''''''''''''''
'  Validate task sequence List
'
```

```
Function ValidateTSList

	Dim oTS
	Dim sCmd
	'ajoute
	Dim oItem
	Set oShell = createObject("Wscript.shell")
	'
	set oTS = new ConfigFile
	oTS.sFileType = "TaskSequences"

	SaveAllDataElements

	If Property("TSGuid") = "" then
		oLogging.CreateEntry "No valid TSGuid found in the environment.", LogTypeWarning
		ValidateTSList = false
	End if

	oLogging.CreateEntry "TSGuid Found: " & Property("TSGuid"), LogTypeVerbose

	If oTS.FindAllItems.Exists(Property("TSGuid")) then
		oEnvironment.Item("TaskSequenceID") =
oUtility.SelectSingleNodeString(oTS.FindAllItems.Item(Property("TSGuid")),"./ID")
	End if



	' Set the related properties

	oUtility.SetTaskSequenceProperties oEnvironment.Item("TaskSequenceID")

	If oEnvironment.Item("OSGUID") <> "" and oEnvironment.Item("ImageProcessor") = "" then
		' There was an OSGUID defined within the TS.xml file, however the GUID was not found
		' within the OperatingSystems.xml file. Which is a dependency error. Block the wizard.
		ValidateTSList = False
		ButtonNext.Disabled = True
		Bad_OSGUID.style.display = "inline"
	Else
		ValidateTSList = True
```

```
□□ButtonNext.Disabled = False
□□Bad_OSGUID.style.display = "none"
□End if
' ajoute
□sCmd = "wscript.exe """ & oUtility.ScriptDir & "\ZTIGather.wsf"""
□oItem = oShell.Run(sCmd, , true)
'
End Function
```

ZTIWindowsUpdate.wsf

J'ai modifié ce script pour ajouter la property "TargetGroup" pour cibler les PC lors du déploiement pour WSUS.

> Il faut donc ajouter un groupe de PC dans WSUS.

```
<job id="ZTIWindowsUpdate">
□<script language="VBScript" src="ZTIUtility.vbs"/>
□<script language="VBScript">
' // ***************************************************************************
' //
' // Copyright (c) Microsoft Corporation.  All rights reserved.
' //
' // Microsoft Deployment Toolkit Solution Accelerator
' //
' // File:      ZTIWindowsUpdate.wsf
' //
' // Version:   6.3.8456.1000
' //
' // Purpose:   Installs all needed updates (drivers, patches, service packs,
' //            etc.) from the Windows Update/Microsoft Update site or WSUS
' //            server, rebooting as required until no more updates are needed.
' //
' // Usage:     cscript.exe [//nologo] ZTIWindowsUpdate.wsf [/debug:true]
' //
' // ***************************************************************************

Option Explicit
```

```
RunNewInstance


'//---------------------------------------------------------------------------
'//  Global Constants
'//---------------------------------------------------------------------------

Const MSIT_WU_REBOOT_MAX = 7
Const MAX_UPDATES = 100


'//---------------------------------------------------------------------------
'//  Main Class
'//---------------------------------------------------------------------------

Class ZTIWindowsUpdate

'//---------------------------------------------------------------------------
'//  Class instance variable declarations
'//---------------------------------------------------------------------------

Public globalVariable
Private privateVariable


'//---------------------------------------------------------------------------
'//  Constructor to initialize needed global objects
'//---------------------------------------------------------------------------

Private Sub Class_Initialize

' No initialization is required

End Sub


'//---------------------------------------------------------------------------
'//  Main routine
'//---------------------------------------------------------------------------
```

```
□Function Main
□
□□Dim iRetVal
□□Dim Item
□□Dim MSIT_WU_Count
□□Dim MSIT_LogType
□□Dim ServiceManager
□□Dim bFoundMU
□□Dim NewUpdateService
□□Dim strCabPath
□□Dim iResult
□□Dim oProgress
□□Dim bFailure, bReboot

□□Main = Success

□□' Validate that are not restarting from a failed install.

□□If ucase(oEnv("SystemDrive")) = "X:" Then
□□□oLogging.CreateEntry "Environment Error: ManualRetry (From ZTIWindowsUpdate).", LogTypeInfo

□□□oEnvironment.Item("LTISuspend") = "LiteTouch is trying to install Windows Updates." & _
□□□□vbNewLine & "This cannot be performed in Windows PE." & _
□□□□vbNewLine & "If booting from a USB Flash Disk, please remove all drives before Retrying." &
□□□□vbNewLine & "Otherwise, ensure the hard disk is selected first in the boot order of the BIO
□□□oEnvironment.Item("SMSTSRebootRequested") = "true"
□□□oEnvironment.Item("SMSTSRetryRequested") = "true"
□□□Main = SUCCESS
□□□exit function
□□End if



□□'//----------------------------------------------------------------------------
□□'//  Initialization
□□'//----------------------------------------------------------------------------

□□MSIT_WU_Count = oEnvironment.Item("MSIT_WU_Count")
□□If not IsNumeric(MSIT_WU_Count) then
```

```
    MSIT_WU_Count = 0
  End if
  oLogging.CreateEntry "Begin Windows Update. Reboot=[" &
oEnvironment.Item("SMSTSRebootRequested") & "]  Retry=[" &
oEnvironment.Item("SMSTSRetryRequested") & "]  Count = " & MSIT_WU_Count , LogTypeInfo

  MSIT_WU_Count = MSIT_WU_Count + 1
  oEnvironment.Item("MSIT_WU_Count") = MSIT_WU_Count



  If oEnvironment.Item("WsusServer") = "" then
    oLogging.ReportProgress "Initializing Windows Update process (pass " & MSIT_WU_Count & ")",
  Else
    oLogging.ReportProgress "Initializing WSUS update process (pass " & MSIT_WU_Count & ")", 0
  End if



  If oEnvironment.Item("SMSTSRebootRequested") <> "" then
    oEnvironment.Item("SMSTSRebootRequested") = ""
  End if
  If oEnvironment.Item("SMSTSRetryRequested") <> "" then
    oEnvironment.Item("SMSTSRetryRequested") = ""
  End if

  If MSIT_WU_Count > MSIT_WU_REBOOT_MAX then
    oLogging.ReportFailure "ZTIWindowsUpdate has run and failed too many times. Count = " &
MSIT_WU_Count, 9902
  End if



  ' Make sure the necessary agent is in place

  iRetVal = VerifyWUA
  If iRetVal = 3010 then

    ' Initiate a reboot and ask that we be re-executed

    oEnvironment.Item("SMSTSRebootRequested") = "true"
    oEnvironment.Item("SMSTSRetryRequested") = "true"
```

```
□□□Exit Function

□□ElseIf iRetVal <> 0 then

□□□oLogging.ReportFailure "Unexpected issue installing the updated Windows Update Agent, rc = "
iRetVal, 9903

□□End if



□□' Opt-In to the Microsoft Update Agent

□□On Error Resume Next
□□□Item = oFSO.GetFileVersion ( ees("%SystemRoot%\System32\WUAUENG.DLL" ) )
□□□oLogging.CreateEntry "Ready to Opt-In to Microsoft Update: WUA Version: " & Item , LogTypeIn
□□□Set ServiceManager = nothing
□□□Set ServiceManager = CreateObject("Microsoft.Update.ServiceManager")
□□On Error Goto 0

□□If ServiceManager is nothing then
□□□oLogging.CreateEntry "Failed to Create Object: Microsoft.Update.ServiceManager" ,
LogTypeWarning
□□Else
□□□ServiceManager.ClientApplicationID = "ZTIWindowsUpdate " & Version
□□□bFoundMU = False
□□□For each Item in ServiceManager.Services
□□□□WScript.Echo "Registered Update Service: " & Item.ServiceID & "   " & Item.Name
□□□□If Item.ServiceID = "7971f918-a847-4430-9279-4a52d1efe18d" then
□□□□□bFoundMU = True
□□□□End if
□□□Next

□□□oLogging.CreateEntry "Microsoft Update Service:  Enabled = " & bFoundMU, LogTypeInfo
□□□If not bFoundMU then
□□□□On Error Resume Next
□□□□□Err.clear
□□□□□If Err.Number <> 0 then
□□□□□□oLogging.CreateEntry "There was an error getting Windows Update to opt into Microsoft Upd
Please verify you are running the latest version of Windows Update Agent." , LogTypeWarning
```

□□□□□End if
□□□□□
□□□□□If oEnvironment.Item("WsusServer") = "" then

□□□□□□'//---
□□□□□□'// Try to find the standalone muauth.cab file and install from it
□□□□□□'//---
□□□□□□'
□□□□□□' From http://download.windowsupdate.com/v9/microsoftupdate/redir/muauth.cab
□□□□□□'
□□□□□□' Place this file in the Distribution\Tools folder so this script can find them.
□□□□□□iResult = oUtility.FindFile("muauth.cab", strCabPath)

□□□□□□If iResult <> Success then
□□□□□□□'// "" will force a internet search for cab file
□□□□□□□strCabPath = ""
□□□□□□End if

□□□□□□oLogging.CreateEntry " about to begin add service ["+ strCabPath +"]", LogTypeInfo
□□□□□
□□□□□□Set NewUpdateService = ServiceManager.AddService2("7971f918-a847-4430-9279-4a52d1efe18d",6,strCabPath)
□□□□□□oLogging.CreateEntry " Status: " & NewUpdateService.RegistrationState, LogTypeInfo
□□□□□End if

□□□□On error goto 0
□□□End if
□□End if

□□'//---
□□'// Process the command line
□□'//---

□□Dim IsRegistered, Query_Only, UpdateCommand, BadKBArticlesList
□□Dim BadGUIDList

□□Query_Only   = FALSE or WScript.Arguments.Named.Exists("QUERY")
□□IsRegistered = FALSE

```
If WScript.Arguments.Unnamed.Count > 0 then
UpdateCommand = WScript.Arguments.Unnamed.Item(0)
Elseif Ucase(oEnvironment.Item("DoCapture")) = "YES" or Ucase(oEnvironment.Item("DoCapture"))
= "PREPARE" then
UpdateCommand = "IsInstalled = 0 and IsHidden = 0 and Type = 'Software'"
Else
UpdateCommand = "IsInstalled = 0 and IsHidden = 0"
End if


' Check to see if this version of Windows has been registered

IsRegistered = FALSE
On Error Resume Next
For each Item in objWMI.InstancesOf("Win32_WindowsProductActivation")
IsRegistered = Item.ActivationRequired = 0
Exit for
Next
On Error Goto 0

oLogging.CreateEntry "Command Line Procesed Query=" & QUery_Only & " Registered=" &
IsRegistered & "  UpdateCommand=[" & UpdateCommand & "]" , LogTypeInfo


Set BadKBArticlesList = oEnvironment.ListItem("WUMU_ExcludeKB")
Set BadGUIDList = oEnvironment.ListItem("WUMU_ExcludeID")


'//----------------------------------------------------------------------------
'//  Search Windows Update
'//----------------------------------------------------------------------------

oLogging.ReportProgress "Searching for updates", 0

Dim UpdateSession, searchResults, updatesToDownload
Dim Downloader, Installer, UpdateResult
Dim kbArticle, bInstall, kb, iSize
Dim i
```

```
□□On Error Resume Next
□□Set updateSession = CreateObject("Microsoft.Update.Session")
□□Set updatesToDownload = CreateObject("Microsoft.Update.UpdateColl")
□□On Error Goto 0

□□If updateSession is nothing then
□□□oLogging.ReportFailure "Failed to Create Object: Microsoft.Update.Session.", 9904
□□End if
□□If updatesToDownload is nothing then
□□□oLogging.ReportFailure "Failed to Create Object: Microsoft.Update.UpdateColl.", 9905
□□End if
□□updateSession.ClientApplicationID = "ZTIWindowsUpdate " & Version

□□oLogging.CreateEntry "Start Search..." , LogTypeInfo
□□On Error Resume Next
□□Set searchResults = updateSession.CreateupdateSearcher().Search(UpdateCommand)
□□If Err then
□□□If Err.Number = &h8024402c then
□□□□oLogging.CreateEntry "Error searching for updates: Not Connected to Internet? (" & Err.Numb
& ")", LogTypeInfo
□□□□Main = Success
□□□ElseIf Err.Number = &h80072ee2 then
□□□□oLogging.CreateEntry "Error searching for updates: ERROR_INTERNET_TIMEOUT: Retry! (" &
Err.Number & ")", LogTypeInfo
□□□□oEnvironment.Item("SMSTSRebootRequested") = "true"
□□□□oEnvironment.Item("SMSTSRetryRequested") = "true"
□□□ElseIf Err.Number = &h80244010 then
□□□□oLogging.CreateEntry "Timeout Error WU_E_PT_EXCEEDED_MAX_SERVER_TRIPS : Retry! (" & Err.Num
& ")", LogTypeInfo
□□□□' See: http://blogs.technet.com/sus/archive/2008/09/18/wsus-clients-fail-with-warning-
syncserverupdatesinternal-failed-0x80244010.aspx
□□□□oEnvironment.Item("SMSTSRebootRequested") = "false"
□□□□oEnvironment.Item("SMSTSRetryRequested") = "true"

□□□Else
□□□□TestAndLog err = 0, "Windows Update, search for updates."
□□□□Main = Failure
□□□End if
```

```
□□□CleanupWhenDone
□□□Exit Function
□□End if
□□On Error Goto 0

□□oLogging.ReportProgress "Processing " & searchResults.Updates.Count & " updates.", 0
□□For each item in searchResults.Updates

□□□bInstall = TRUE

□□□On Error Resume Next

□□□item.AcceptEula

□□□If item.InstallationBehavior.CanRequestUserInput then
□□□□bInstall = FALSE ' Do NOT install anything that can Request User Input!
□□□End if

□□□For each kb in Item.Categories
□□□□if ucase(kb.Name) = "DRIVERS" then
□□□□□bInstall = TRUE ' Some XP drivers may be marked as CanRequestUserInput. Override!
□□□□□exit for
□□□□elseif ucase(kb.Name) = "WINDOWS VISTA ULTIMATE LANGUAGE PACKS" then
□□□□□bInstall = FALSE ' Most users don't want *ALL* Language Packs. Too much. Override!
□□□□□exit for
□□□□end if
□□□Next



□□□If BadKBArticlesList.Count > 0 then
□□□□For each kbArticle in item.KBArticleIDs
□□□□□For each kb in BadKBArticlesList
□□□□□□If lcase(kb) = lcase(kbArticle) then
□□□□□□□bInstall = FALSE ' Do NOT install any patch in the Bad KB articles list!
□□□□□□End if
□□□□□Next
□□□□Next
□□□End if
```

```
□□□For each kbArticle in BadGUIDList
□□□□If lcase(item.Identity.UpdateID) = lcase(kbArticle) then
□□□□□bInstall = FALSE ' Do NOT install any patch in the Bad GUID articles list!
□□□□End if
□□□Next



□□□iSize = empty
□□□kb = ""
□□□for i = 0 to item.KBArticleIDs.Count - 1
□□□□If instr(1,Item.Title,item.KBArticleIDs(i),vbTextCompare) = 0 then
□□□□□oStrings.AddToList kb, "KB" & item.KBArticleIDs(i), " "
□□□□End if
□□□next
□□□iSize = item.MinDownloadSize
□□□If item.MaxDownloadSize > 0 then
□□□□iSize = Item.MaxDownloadSize
□□□End if
□□□If kb <> "" then
□□□□kb = "  [ " & kb & " ]"
□□□End if
□□□If iSize > 0 then
□□□□kb = kb & " - " & FormatLargeSize(iSize)
□□□End if
□□□
□□□If bInstall = TRUE and updatesToDownload.count < MAX_UPDATES then
□□□□oLogging.CreateEntry "INSTALL - " & item.Identity.UpdateID & " - " & Item.Title & kb,
LogTypeInfo
□□□□updatesToDownload.Add(Item)
□□□Else
□□□□oLogging.CreateEntry "  SKIP  - " & item.Identity.UpdateID & " - " & Item.Title & kb,
LogTypeInfo
□□□End if

□□□On Error Goto 0

□□Next

□□oLogging.CreateEntry "Scan complete, ready to install updates. Count = " &
```

```
updatesToDownload.Count, LogTypeInfo

□□If updatesToDownload.Count = 0 or Query_Only then
□□□oLogging.CreateEntry "This computer is up to date (Success)" , LogTypeInfo
□□□oEnvironment.Item("MSIT_WU_Count") = ""  ' Reset the counter

□□□CleanupWhenDone
□□□Main = Success
□□□Exit Function
□□□
□□End if


□□If MSIT_WU_Count > MSIT_WU_REBOOT_MAX - 1 then
□□□MSIT_LogType = LogTypeWarning
□□Else
□□□MSIT_LogType = LogTypeInfo
□□End if


□□'//---------------------------------------------------------------------------
□□'//  Download binaries
□□'//---------------------------------------------------------------------------

□□Set oProgress = new Progress

□□oLogging.CreateEntry "Begin Downloading...", LogTypeInfo

□□Set Downloader = updateSession.CreateUpdateDownloader()
□□Downloader.Updates = UpdatesToDownload
□□Set UpdateResult = Downloader.BeginDownload(oProgress, oProgress, vbNull)

□□On Error Resume Next
□□While not UpdateResult.IsCompleted
□□□oLogging.ReportProgress "Downloading " &
UpdatesToDownload(UpdateResult.GetProgress.CurrentUpdateIndex).Title,
UpdateResult.GetProgress.PercentComplete
□□□WScript.Sleep 500
□□WEnd
```

```
    On Error Goto 0

    For item = 0 to UpdatesToDownload.Count - 1
        If not UpdatesToDownload.Item(item).IsDownloaded then
            oLogging.CreateEntry "    Failed to download: " &
UpdatesToDownload.Item(item).Identity.UpdateID & _
                "  result(" & UpdateResult.GetProgress.GetUpdateResult(item).ResultCode & ") : " &
UpdatesToDownload.Item(item).Title, MSIT_LogType
        End if
    Next

    On Error Resume Next
    Downloader.EndDownload UpdateResult
    On Error Goto 0



    '//----------------------------------------------------------------------------
    '//  Install Binaries
    '//----------------------------------------------------------------------------

    oLogging.CreateEntry "Begin Installation...", LogTypeInfo

    Set Installer = updateSession.CreateUpdateInstaller()
    Installer.Updates = UpdatesToDownload
    Set UpdateResult = nothing

    On Error Resume Next
    Set UpdateResult = Installer.BeginInstall(oProgress, oProgress, vbNull)
    If UpdateResult is nothing then

        ' Some unknown error returned from the installer, reboot and try again.

        oLogging.CreateEntry "Installer.Install() returned Unknown failure! " & err.number & " " &
Err.Description, LogTypeInfo
        oEnvironment.Item("SMSTSRebootRequested") = "true"
        oEnvironment.Item("SMSTSRetryRequested") = "true"
        Exit Function

    End if
```

```
□□On Error Goto 0

□□On Error Resume Next
□□While not UpdateResult.IsCompleted
□□□oLogging.ReportProgress "Installing " &
UpdatesToDownload(UpdateResult.GetProgress.CurrentUpdateIndex).Title,
UpdateResult.GetProgress.PercentComplete
□□□WScript.Sleep 500
□□WEnd
□□On Error Goto 0

□□bReboot = False
□□bFailure = False
□□For item = 0 to UpdatesToDownload.Count - 1
□□□If not UpdatesToDownload.Item(item).IsInstalled then
□□□□If UpdateResult.GetProgress.GetUpdateResult(item).ResultCode <> 2 then
□□□□□oLogging.CreateEntry "    " & UpdatesToDownload.Item(item).Identity.UpdateID & _
□□□□□□"  result(" & UpdateResult.GetProgress.GetUpdateResult(item).ResultCode & " / HR = " &
hex(UpdateResult.GetProgress.GetUpdateResult(item).HResult) & _
□□□□□□" ) : " & UpdatesToDownload.Item(item).Title , MSIT_LogType
□□□□□bFailure = True
□□□□End if
□□□□If UpdateResult.GetProgress.GetUpdateResult(item).RebootRequired then
□□□□□bReboot = True
□□□□End if
□□□End if
□□Next

□□On Error Resume Next
□□Installer.EndInstall UpdateResult
□□On Error Goto 0



□□'//---------------------------------------------------------------------------
□□'//  Cleanup
□□'//---------------------------------------------------------------------------

□□If bFailure then
```

```
□□□oLogging.CreateEntry "Failure, Please run again!" , LogTypeInfo
□□□oEnvironment.Item("SMSTSRetryRequested") = "true"
□□□oEnvironment.Item("SMSTSRebootRequested") = "true"

□□ElseIf bReboot then

□□□oLogging.CreateEntry "More to install, Please reboot and run again!" , LogTypeInfo
□□□oEnvironment.Item("SMSTSRetryRequested") = "true"
□□□oEnvironment.Item("SMSTSRebootRequested") = "true"

□□Else
□□
□□□' A recently installed MicrosoftUpdate/WindowsUpdate component *may* require more/new update
□□□' Rerun Main() to ensure that all updates are installed. Exit above when MU/WU returns NO
updates.

□□□oLogging.CreateEntry "Success! Please rerun WindowsUpdate to ensure machine is FULLY up to
date." , LogTypeInfo
□□□Main = Main()

□□□If LCase(oEnvironment.Item("SMSTSRetryRequested")) <> "true" then
□□□□CleanUpWhenDone
□□□End if

□□End if



□End function


□'//---------------------------------------------------------------------------
□'//  Functions
□'//---------------------------------------------------------------------------

□Function CleanUpWhenDone

□□Dim NoAutoUpdateState
□□NoAutoUpdateState = oEnvironment.Item("NoAutoUpdate_Previous")
```

```
  If NoAutoUpdateState = "<empty>" or NoAutoUpdateState= "" then
    oLogging.CreateEntry "Restore NoAutoUpdateKey to <empty>.", LogTypeInfo
    On Error Resume Next
      oShell.RegDelete
"HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\AU\NoAutoUpdate"
    On Error Goto 0
  ElseIf NoAutoUpdateState <> "" then
    oLogging.CreateEntry "Restore NoAutoUpdateKey to " & NoAutoUpdateState, LogTypeInfo
    oShell.RegWrite
"HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\AU\NoAutoUpdate",
NoAutoUpdateState, "REG_DWORD"
  Else
    oLogging.CreateEntry "Unknown previous NoAutoUpdateKey State, Do Nothing [" &
NoAutoUpdateState & "].", LogTypeInfo
  End if
  
End Function

Function VerifyWUA

  Dim iResult
  Dim strExePath, bUpdateNeeded, objAgentInfo
  Dim intMajorVersion
  Dim sArchitecture
  Dim iNoAutoUpdate


  '//---------------------------------------------------------------------------
  '//  Ensure the desired tracing registry entries are in place
  '//---------------------------------------------------------------------------

  On error resume next
  
  If UCase(oEnvironment.Item("Debug")) = "TRUE" then
  
    oShell.RegWrite "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\WindowsUpdate\Trace\Level",
"REG_DWORD"
```

```
□□□oShell.RegWrite
"HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\WindowsUpdate\Trace\Handler\Flags",
&h000000ff, "REG_DWORD"
□□□oShell.RegWrite
"HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\WindowsUpdate\Trace\Handler\Level", 3,
"REG_DWORD"

□□□oShell.RegWrite
"HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\WindowsUpdate\Trace\COMAPI\Flags", &h000000ff,
"REG_DWORD"
□□□oShell.RegWrite
"HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\WindowsUpdate\Trace\COMAPI\Level", 3,
"REG_DWORD"
□□□On error goto 0

□□End if

□□'//---------------------------------------------------------------------------
□□'//  Configure Windows Update settings
□□'//---------------------------------------------------------------------------

□□If oEnvironment.Item("WsusServer") <> "" then

□□□' Configure the WSUS server in the registry.  This needs to be a URL (e.g. http://myserver).

□□□oLogging.CreateEntry "Configuring client to use WSUS server " &
oEnvironment.Item("WsusServer"), LogTypeInfo

□□□oShell.RegWrite
"HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\WUServer",
oEnvironment.Item("WsusServer"), "REG_SZ"
□□□oShell.RegWrite
"HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\WUStatusServer",
oEnvironment.Item("WsusServer"), "REG_SZ"
' ajoute
□□□If oEnvironment.Item("TargetGroup") <> "" then
 □□□□oShell.RegWrite
"HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\TargetGroup",
oEnvironment.Item("TargetGroup"), "REG_SZ"
```

```
□□□□oShell.RegWrite
"HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\TargetGroupEnabled",
00000001, "REG_DWORD"
□□□End if
'
□□End if

□□oLogging.CreateEntry "Configuring Windows Update settings (manual update, use server)",
LogTypeInfo

□□If oEnvironment.Item("NoAutoUpdate_Previous") = "" then
□□□On Error Resume Next
□□□□iNoAutoUpdate =
oShell.RegRead("HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\AU\NoAuto
Update")
□□□□If iNoAutoUpdate = "" then
□□□□□iNoAutoUpdate = "<empty>"
□□□□End if
□□□□oLogging.CreateEntry "Archive NoAUtoUpdate State: Was [" & iNoAutoUpdate & "].", LogTypeInf
□□□□oEnvironment.Item("NoAutoUpdate_Previous") = iNoAutoUpdate
□□□On Error Goto 0
□□End if

□□oShell.RegWrite
"HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\AU\UseWUServer", 1,
"REG_DWORD"
□□oShell.RegWrite
"HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\AU\NoAutoUpdate", 1,
"REG_DWORD"



□□' Restart the service to get the latest settings

□□oShell.Run "net stop wuauserv", 0, true
□□oShell.Run "net start wuauserv", 0, true



□□'//-------------------------------------------------------------------------
□□'//  Ensure the needed Windows Update Agent version is installed
```

```
□□'//---------------------------------------------------------------------------

□□bUpdateNeeded = True   ' init value, do not touch



□□' See if the version is sufficient

□□On Error Resume Next
□□Set objAgentInfo = CreateObject("Microsoft.Update.AgentInfo")
□□If Err.Number = 0 then

□□□' Make sure ApiMajorVersion is 4 or higher (Version 4 is needed to opt-in to Microsoft Updat

□□□intMajorVersion = 0   ' init value
□□□intMajorVersion = objAgentInfo.GetInfo("ApiMajorVersion")
□□□If intMajorVersion >= 4 Then
□□□□bUpdateNeeded = False
□□□□oLogging.CreateEntry "Windows Update Agent verion " & intMajorVersion & " found, OK to
continue", LogTypeInfo
□□□Else
□□□□oLogging.CreateEntry "Windows Update Agent verion " & intMajorVersion & " found, upgrade
needed", LogTypeInfo
□□□End if

□□Else
□□□oLogging.CreateEntry "Unable to create Microsoft.Update.AgentInfo object, Windows Update Age
upgrade is needed", LogTypeInfo
□□End if



□□If not bUpdateNeeded then
□□□VerifyWUA = 0
□□□Exit Function
□□End if



□□'//---------------------------------------------------------------------------
□□'//  Try to find the standalone installer file and install from it
□□'//---------------------------------------------------------------------------
```

```
□□' From http://technet.microsoft.com/en-us/library/bb932139.aspx, you can obtain the
□□' Windows Update Agent stand-alone installer from:
□□'
□□' http://go.microsoft.com/fwlink/?LinkID=100334 (WindowsUpdateAgent30-x86.exe)
□□' http://go.microsoft.com/fwlink/?LinkID=100335 (windowsupdateagent30-x64.exe)
□□'
□□' Place these files in the Distribution\Tools\<platform> folder so this script can find them.


□□sArchitecture = lcase(oEnvironment.Item("Architecture"))
□□If sArchitecture = "" then
□□□sArchitecture = lcase(EES("%Processor_Architecture%"))
□□End if
□□If sArchitecture = "amd64" then
□□□sArchitecture = "x64"
□□End if

□□iResult = oUtility.FindFile("WindowsUpdateAgent30-" & sArchitecture & ".exe", strExePath)
□□If iResult = Success then
□□□oLogging.CreateEntry "About to install updated Windows Update Agent from " & strExePath,
LogTypeInfo
□□□iResult = oShell.Run(strExePath & " /quiet /norestart", 0, true)
□□□oLogging.CreateEntry "Windows Update Agent installation return code = " & iResult, LogTypeIn
□□□VerifyWUA = 3010
□□□Exit Function
□□End if

□□oLogging.CreateEntry "Unable to find WindowsUpdateAgent30-" & sArchitecture & ".exe, will
attempt to download", LogTypeInfo


□□'//---------------------------------------------------------------------------
□□'//  Download the Windows Update Agent
□□'//---------------------------------------------------------------------------

□□' See http://msdn2.microsoft.com/en-us/library/aa387285.aspx for the basic logic used here.
```

```
□□Dim sWURedistCab, oWUXML, sFileVer1, sFileVer2, sWUDownload

□□sWURedistCab = InternetFileDownload("http://update.microsoft.com/redist/wuredist.cab")
□□VerifyCriticalFile sWURedistCab, "wuredist.cab"

□□' Extract XML File

□□oUtility.RunWithHeartbeat ees("Expand.exe -r " & sWURedistCab & " -F:wuRedist.xml %temp%")
□□VerifyCriticalFile "%Temp%\WURedist.xml", "wuRedist.xml"

□□' Load XML File

□□Set oWUXML = oUtility.CreateXMLDOMObjectEx(EES("%Temp%\WURedist.xml"))
□□If oWUXML is nothing then
□□□oLogging.CreateEntry "Failed to load: %Temp%\WURedist.xml" , LogTypeError
□□□VerifyWUA = 1
□□□Exit function
□□End if

□□' Get Local File Version

□□sFileVer1 = oFSO.GetFileVersion ( ees("%SystemRoot%\System32\WUAUENG.DLL" ) )
□□oLogging.CreateEntry "Current Version %SystemRoot%\System32\WUAUENG.DLL : " & sFileVer1 ,
LogTypeInfo

□□' Get New File Version

□□sFileVer2 = oWUXML.selectSingleNode ("//WURedist/StandaloneRedist/architecture[@name='" &
sArchitecture & "']/@clientVersion").Text
□□oLogging.CreateEntry "Current Version wuredist.cab : " & sFileVer2 , LogTypeInfo

□□' Download and install if file Versions don't match
```

```
  If sFileVer1 <> sFileVer2 then
    sWUDownload = InternetFileDownload( oWUXML.DocumentElement.selectSingleNode
("//WURedist/StandaloneRedist/architecture[@name='" & sArchitecture & "']/@downloadUrl").Text
)
    VerifyCriticalFile sWUDownload, "WUDownload.exe"
    iResult = oUtility.RunWithHeartbeat(sWUDownload & " /wuforce /quiet /norestart")

    VerifyWUA = iResult
    Exit Function
  End if


  ' Cleanup

  On Error Resume Next
  For each item in array ( sWURedistCab, EES("%Temp%\WURedist.xml"), sWUDownload )
    If oFSO.FileExists(item) then
      oFSO.DeleteFile item
    End if
  Next
  On Error Goto 0

End Function


Function InternetFileDownload( InternetURL  )
  Dim InternetBuffer
  Dim ADODB

  Set ADODB = CreateObject("ADODB.Stream")
  Set InternetBuffer = CreateObject("Msxml2.XmlHttp")
  InternetBuffer.open "GET", InternetURL , false
  On Error Resume Next
  InternetBuffer.send ""
  On Error Goto 0

  If InternetBuffer.ReadyState = 4 then
    oLogging.CreateEntry "Status: " & InternetBuffer.Status & " " & InternetURL, LogTypeInfo
  Else
```

```
□□□oLogging.CreateEntry "Ready State : " & InternetBuffer.ReadyState & " " & InternetURL ,
LogTypeWarning
□□End if

□□If InternetBuffer.Status = 200 then
□□□If ADODB.State <> 0 then ADODB.Close
□□□ADODB.Type = 1 '(1=binary,2=Text)
□□□ADODB.Mode = 3 '(1=Read,2=Write,3=RW)
□□□ADODB.Open
□□□ADODB.Write InternetBuffer.ResponseBody
□□□ADODB.SaveToFile EES( "%temp%\" & oFSO.GetFileName(InternetURL) ) , 2
□□□ADODB.Close
□□End if

□□If InternetBuffer.Status = 200 then
□□□InternetFileDownload = EES( "%temp%\" & oFSO.GetFileName(InternetURL) )
□□End if

□End function

□Function FormatLargeSize( lSize )

□□Dim i
□□For i = 1 to len(" KMGTPEZY")
□□□If cdbl(lSize) < 1024 ^ i then
□□□□FormatLargeSize = int(cdbl(lSize)/(1024^(i-1))) & " " & mid(" KMGTPEZY",i,1) & "B"
□□□□Exit function
□□□End if
□□next

□End function

□Function EES ( EnvStr )
□□EES = oShell.ExpandEnvironmentStrings( EnvStr )
□End function



□Sub VerifyCriticalFile (FileName, Description)
```

```
□□If FileName = "" or not oFSO.FileExists(ees(FileName)) then
□□□oLogging.CreateEntry Description & " not found: " & FileName , LogTypeError
□□□oLogging.CreateEntry "   Most likely cause: No Internet Access or unconfigured Proxy
settings!", LogTypeError
□□□oLogging.ReportFailure "Critical file " & FileName & " was not found, aborting", 9906
□□End if

□End sub

End Class

Class Progress
□Public Default Function Process
□End Function
End Class
□</script>
</job>
```

Raccourcis.ps1

Ce script me permet d'ajouter des raccourcis dans le bureau public de l'ordinateur.

```
# Raccourci Office.com avec icône
Copy-Item "\\RN-SRV-WDS01.ad.khroners.fr\GPO$\Raccourcis\Office.ico" -Destination
"C:\Office.ico"
$TargetFile = "https://www.office.com/"
$shortcutFile = "C:\Users\Public\Desktop\Portail Office Web.lnk"
$WScriptShell = New-Object -ComObject WScript.Shell
$shortcut = $WScriptShell.CreateShortcut($ShortcutFile)
$shortcut.TargetPath = $TargetFile
$shortcut.IconLocation = "C:\Office.ico"
$shortcut.Save()

# Raccourcis suite Microsoft365
Copy-Item -Path "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Word.lnk" -Destination
"c:\users\Public\Desktop" -Force
Copy-Item -Path "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Excel.lnk" -Destination
"c:\users\Public\Desktop" -Force
Copy-Item -Path "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\PowerPoint.lnk" -
```

```
Destination "c:\users\Public\Desktop" -Force
Copy-Item -Path "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\OneNote.lnk" -
Destination "c:\users\Public\Desktop" -Force
Copy-Item -Path "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Outlook.lnk" -
Destination "c:\users\Public\Desktop" -Force
Copy-Item -Path "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Publisher.lnk" -
Destination "c:\users\Public\Desktop" -Force
```

Install_Drivers.ps1

Ce script me permet de télécharger les pilotes via Windows Update.

```
$Session = New-Object -ComObject Microsoft.Update.Session
$Searcher = $Session.CreateUpdateSearcher()

$Searcher.ServiceID = '7971f918-a847-4430-9279-4a52d1efe18d'
$Searcher.SearchScope =  1 # MachineOnly
$Searcher.ServerSelection = 3 # Third Party

$Criteria = "IsInstalled=0 and Type='Driver'"
Write-Host('Searching Driver-Updates...') -Fore Green
$SearchResult = $Searcher.Search($Criteria)
$Updates = $SearchResult.Updates
□
#Show available Drivers...
$Updates | select Title, DriverModel, DriverVerDate, Driverclass, DriverManufacturer | fl

$UpdatesToDownload = New-Object -Com Microsoft.Update.UpdateColl
$updates | % { $UpdatesToDownload.Add($_) | out-null }
Write-Host('Downloading Drivers...')  -Fore Green
$UpdateSession = New-Object -Com Microsoft.Update.Session
$Downloader = $UpdateSession.CreateUpdateDownloader()
$Downloader.Updates = $UpdatesToDownload
$Downloader.Download()

$UpdatesToInstall = New-Object -Com Microsoft.Update.UpdateColl
$updates | % { if($_.IsDownloaded) { $UpdatesToInstall.Add($_) | out-null } }

Write-Host('Installing Drivers...')  -Fore Green
```

```
$Installer = $UpdateSession.CreateUpdateInstaller()
$Installer.Updates = $UpdatesToInstall
$InstallationResult = $Installer.Install()
if($InstallationResult.RebootRequired) {
Write-Host('Reboot required! please reboot now..') -Fore Red
} else { Write-Host('Done..') -Fore Green }

$updateSvc.Services | ? { $_.IsDefaultAUService -eq $false -and $_.ServiceID -eq "7971f918-
a847-4430-9279-4a52d1efe18d" } | % { $UpdateSvc.RemoveService($_.ServiceID) }
```

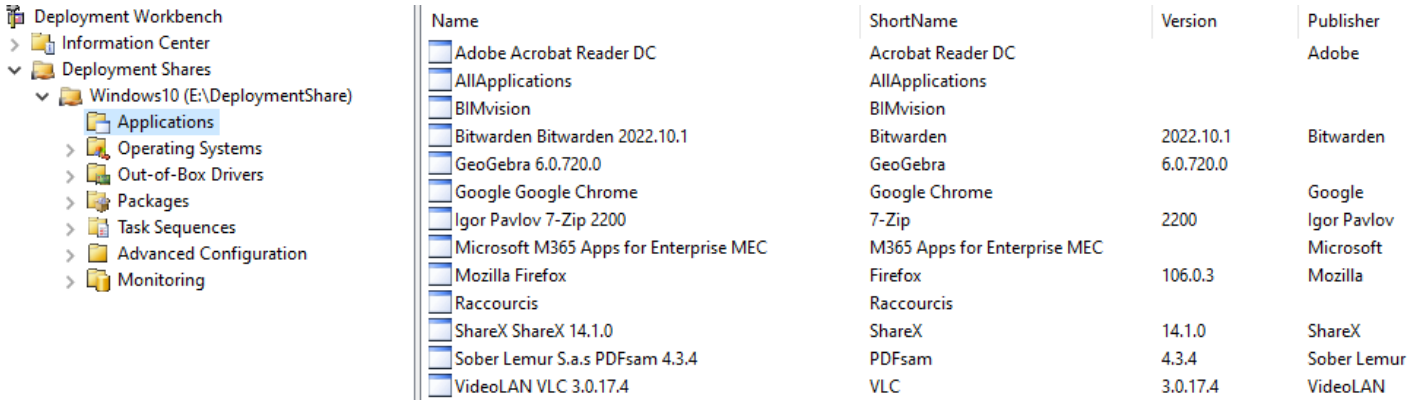
RestoreREG.ps1

Ce script permet de restaurer le registre suite à ma modification de ce dernier avant le déploiement : [Modifier le registre d... | Docs Khroners]

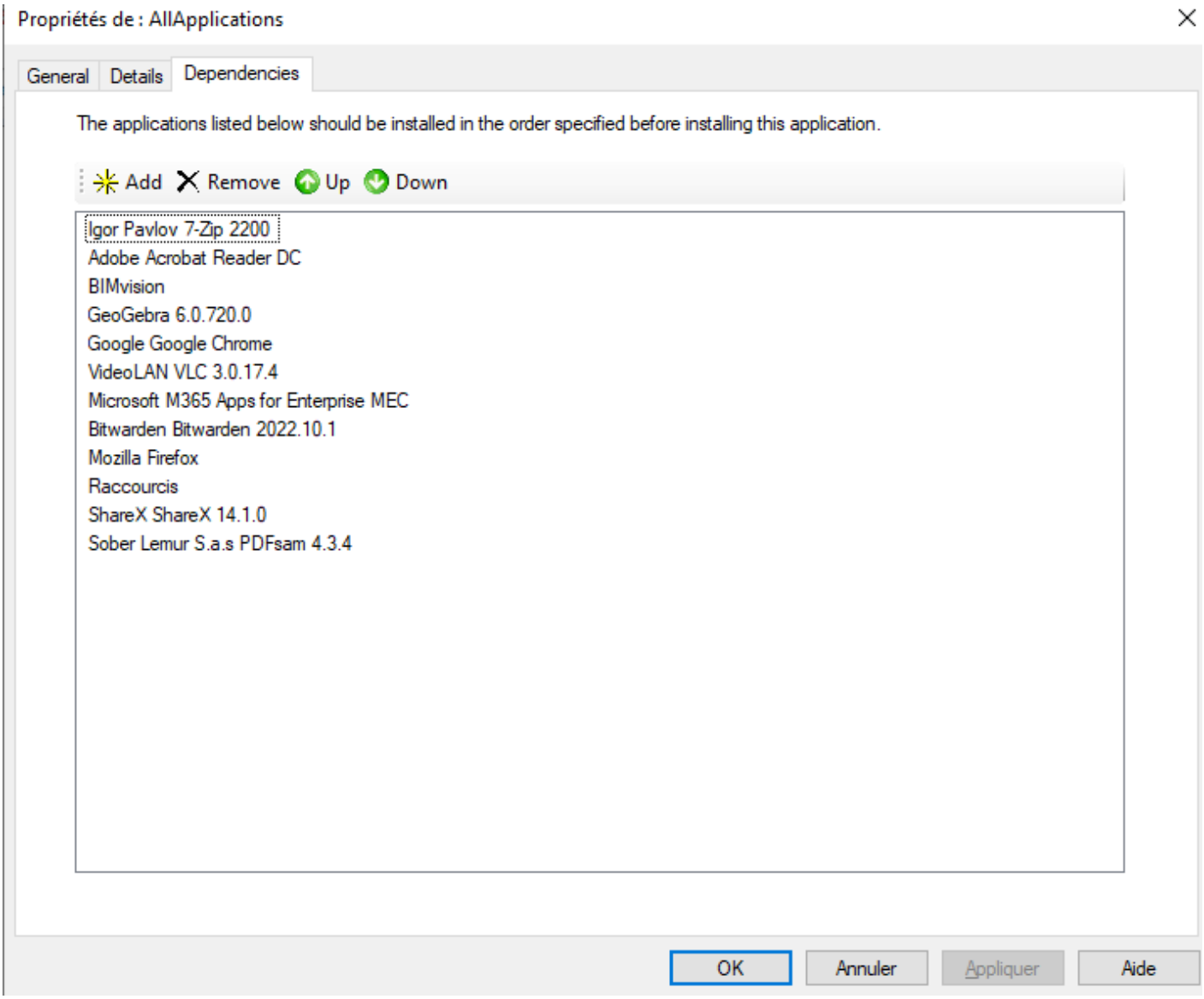
```
reg add HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\DriverSearching /v
SearchOrderConfig /t REG_DWORD /d 00000001 /f
reg delete HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate /f
```

Applications

Liste



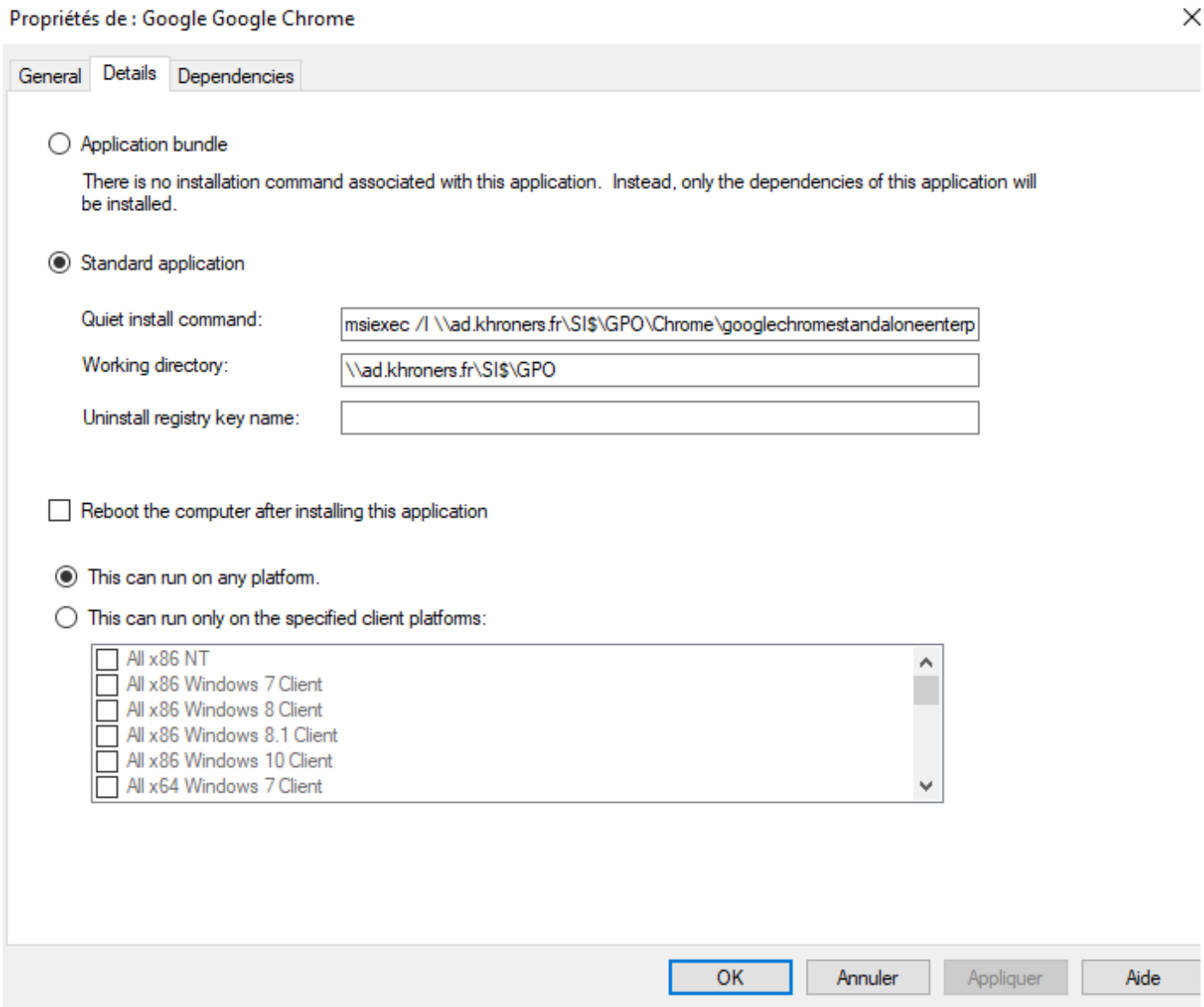
Bundle pour installer toutes les applications



Exemple de commande d'installation

MSI

```
msiexec /I \\ad.khroners.fr\SI$\GPO\Chrome\googlechromestandaloneenterprise64.msi /qn
```



> Si le répertoire où est placé le setup est sur un partage DFS (ou simplement en dehors du DeploymentShare (à vérifier)), il faut qu'une des applications du bundle comprenne le partage dans le champ "Working Directory".

EXE

```
\\ad.khroners.fr\SI$\GPO\Bitwarden\Bitwarden-Installer-2022.10.1.exe /S /ALLUSERS
```

Revision #6
Created 2022-11-06 16:51:22 UTC by Khroners
Updated 2022-11-07 17:37:00 UTC by Khroners