

## Minutes from National iMedConsent™ VANTS Call

Wednesday, January 9, 2008

Ray Frazier, National Center for Ethics in Health Care

### 1. Upcoming Release

There will be no content release in January.

### 2. Dell “Pilot” Set to Begin

The beta testing of these leased machines will take place in mid-January. We want all iMed POCs at facilities where these machines are being tested to get involved and make sure that iMedConsent is tested and functions properly. Let me know if and when you have any results from this testing and I will pass along the results to the group.

### 3. State Authorized Portable Orders, Handbook 1004.04

The group discussed the Handbook passage:

g. Ensuring that state-authorized portable order forms and identifiers, from the state in which the facility is located, are available for ready access on units, in clinics, and in community-based settings. NOTE: To facilitate access to state forms, the Facility Director may want to instruct the iMedConsent Administrator to enter relevant forms into the iMedConsent library for easy download.

After providing a brief description of what portable orders are (see the [Handbook](#) for details), some POCs shared their progress investigating whether iMedConsent can be used to complete their State’s order.

Facility POCs in El Paso TX and Miami FL indicated that they had difficulty replicating the State form using the iMedConsent forms creator. Careful replication is essential because the form needs to be readily recognized by first responders. Miami indicated that they attempted to get approval from the Florida Department of Health and were denied. They are continuing to work on the process and will connect with Dialog Medical to get assistance in refining the appearance of their iMed form. We may be able to work with Dialog Medical to make the form replication process more nuanced so that the portable orders forms will be able to be represented more faithfully in the iMed program. We will explore this in the coming months.

No facilities on the call had progressed beyond the creation and approval stage to the implementation phase. We expect that there will be numerous issues to overcome (printer availability, colored stock printing, etc.) if and when facilities are able to implement an approved form in iMedConsent. Ultimately, implementation of portable order documentation in iMedConsent may not give a sufficient “bang for the buck” at most facilities. This is up to you to decide at the local level.

#### 4. **Identifying Practitioners on the Consent Form**

This issue was discussed briefly. The patient needs to know who is going to be involved in the performance and supervision of the treatment/procedure. If the individuals identified in the consent form change before the treatment/procedure, the patient needs to be informed and the patient's assent must be documented in the patient record. A new consent form does not necessarily need to be completed.

#### 5. **Vergence Update**

There is no new news on this issue, however the Ethics Center has renewed our efforts to have this addressed by OI&T leadership.

#### 6. **Flash 9 Update**

A security flaw necessitated Flash to release an update (version 9). If sites update Flash, and have used an early version of the iMedConsent installer on to download iMedConsent on their workstations, they can experience problems with iMedConsent.

Bill Taylor at Dialog Medical drafted a summary of this issue which I am including below. After Bill drafted his comments, Jason Woodard in VISN 7 provided a very comprehensive, illustrated guide detailing step-by-step how to push this reinstaller via SMS (THANKS JASON!). In reading this document, please remember that each site is different, and that each site should conduct their own small-scale testing of the installation before attempting a wide-scale push.

I have included Jason's document beginning on Page 6.

Here is Bill's summary:

A couple of weeks ago, it was reported to use that several sites in VISN 07 were seeing Error 1406 multiple times when attempting to open iMedConsent. The error further referenced a CLSID that, when explored, proved to be a Flash registry key. The appearance of these errors coincided with a push done by VISN 07 to update the Flash version on all the workstations due to a security vulnerability. The latest Flash version is now Flash 9e, or 9.0.115.

This impacted many of the workstations within the VISN that had been installed with an older iMedConsent installer. That installer tried to ensure that workstations using iMedConsent had Flash installed by embedding into the iMedConsent installation an earlier version of Flash (7.x). Once the workstations had been updated to Flash 9.0.115, any action by the workstation which caused it to reference the original .MSI used to install iMedConsent would cause the application to try to 'self-repair' or reinstall itself back to the original version of Flash. The new version of Flash, however, locks the registry keys against the attempted downgrade. That is the 1406 error that is thrown on-screen to the user.



You will need to edit the `reinstaller.bat` file to a) specify the name of the iMedConsent server in the `set serverName=` line. This installer, by default, already has the uninstallation/reinstallation of Flash disabled. This should allow the workstation to continue using whatever version of Flash was already on the workstation without issue.

the opening lines of the `reinstaller.bat` file look like this (see below). It is important that you edit the file in the **set serverName=** location (see bolded green text below if viewing in color) and replace that with the NETBIOS name of your iMedConsent server. Changing the name once here will populate the server name throughout the rest of the batch file – you need not make the change in any other place.

By default, the Flash portion of the batch file is inactive. It is recommended that you leave this inactive, and do Flash management via some other method. However, if you do want to use the batch file to push Flash, please contact us for further instructions.

---

```
@echo off

rem -----
rem -----
rem -
rem - iMedConsent silent uninstall and install
rem - sample BATCH Script
rem -
rem - Last updated: 2007-01-11
rem -           2007-09-05- Turned off Flash reinstall
rem -----
SETLOCAL ENABLEDELAYEDEXPANSION
set serverName=YOUR_SERVER_NAME_HERE
set doFlash=
set doEpad=
set copyLocal=
set doRuntimes=

rem -----
rem Customization section
rem -----

rem uncomment the next line to reinstall flash
rem set doflash=1

rem uncomment the next line to install epad drivers
rem set doEpad=1

rem uncomment the next line to copy the install package from a local
folder
set copyLocal=1

rem uncomment the next line to install prerequisite runtimes
set doRuntimes=1
```

```
rem -----  
rem -- DO NOT EDIT BELOW THIS LINE without testing  
rem -----
```

---

Run this batch file on the workstations which you are having issues.

From a workstation (or in your case, the computer on which you're building the image), map back to the UNC path `\\%iMed_server%\iMedConsent$\iMed37\Install\Workstation` and run the `reinstaller.bat` file (note the dollar sign after 'iMedConsent' denoting the share). This will copy the other self-extracting .EXE file to the /TEMP directory of the computer, extract itself there, uninstall any existing instance of iMedConsent, install some prerequisite VB6 and Visual C++ runtime libraries, and then install iMedConsent.

# 2008

## Visn7 iMedConsent 3.81 SMS Deployment



Woodard, Jason

Department of Veterans Affairs

1/8/2008

**Contents**

Background ..... 4

SMS Queries ..... 5

SMS Collections ..... 5

    Building the SMS Collections ..... 5

SMS Package ..... 7

    Building the SMS Package ..... 8

SMS Advertisement ..... 13

Gradually Add Client to the "Empty Collection" ..... 16

Appendix ..... 20

    MakeColl.vbs ..... 20



## Background

Obtain the Custom Installer Script written by Visn7 EST team. This Installer Script will perform the same steps as the “Reinstaller.bat”, but is more SMS friendly. The script will accept command line option for specifying the iMedServer name. The Executable is ~78mb and contains all files needed for the installation.

The command line options are:

```
iMedConsent3.81_Full.EXE /?  
iMedConsent3.81_Full.EXE /S:{Site_iMedServer}
```

Command Line Options:

```
/S:{Your iMedServer} **in the /S the “S” is capital**
```

A site specific example:

```
iMedConsent3.81_Full.EXE /S:vhadubmul3
```

This EXE depends on the iMed Server folder structure being:

```
\\vhaXXXimed\imedconsent\$\iMed37
```

The Visn7 EXE can be obtained at this location:

```
http://vhadubfpc14.v07.med.va.gov/iMedconsent3.81\_full.zip
```

This install includes the Updated FlashPlayer. 9.0.115.0

You may still want download the files from Dialog Medical for the purpose of installing directly from the iMed Server.

Download the following file from support website for Dialog Medical  
3.805reinstaller.exe

Extract this file to the following location on the iMed Server  
\\{Your\_iMED\_Server}\imedconsent\$\iMed37\Install\WorkStation

Edit the “reinstaller.bat” file. Look for the following line and set your sites iMed Server name (no \\)  
(\*\*Not needed for this the EXE , but will be used for installed from the iMed Server)  
set serverName=YOUR\_SERVER\_NAME\_HERE

## SMS Queries

1. Make 2 SMS queries to find how many clients have the software installed. This is based off what is listed in “Add Remove Programs” on each client.

SMS Query: (Visn7 – iMedConsent (All Versions))

```
select SMS_G_System_SYSTEM.Name, SMS_R_System.IPAddresses,
SMS_R_System.LastLogonUserName, SMS_G_System_ADD_REMOVE_PROGRAMS.DisplayName,
SMS_G_System_ADD_REMOVE_PROGRAMS.Version from SMS_R_System inner join
SMS_G_System_SYSTEM on SMS_G_System_SYSTEM.ResourceID = SMS_R_System.ResourceID
inner join SMS_G_System_ADD_REMOVE_PROGRAMS on
SMS_G_System_ADD_REMOVE_PROGRAMS.ResourceID = SMS_R_System.ResourceID where
SMS_G_System_ADD_REMOVE_PROGRAMS.DisplayName like "%iMedConsent Workstation%"
```

SMS Query: (Visn7 – iMedConsent (<3.81))

```
select SMS_G_System_SYSTEM.Name, SMS_R_System.IPAddresses,
SMS_R_System.LastLogonUserName, SMS_G_System_ADD_REMOVE_PROGRAMS.DisplayName,
SMS_G_System_ADD_REMOVE_PROGRAMS.Version from SMS_R_System inner join
SMS_G_System_SYSTEM on SMS_G_System_SYSTEM.ResourceID = SMS_R_System.ResourceID
inner join SMS_G_System_ADD_REMOVE_PROGRAMS on
SMS_G_System_ADD_REMOVE_PROGRAMS.ResourceID = SMS_R_System.ResourceID where
SMS_G_System_ADD_REMOVE_PROGRAMS.DisplayName like "%iMedConsent Workstation%"
and SMS_G_System_ADD_REMOVE_PROGRAMS.Version != "3.81"
```

## SMS Collections

Being the install point is the iMED Server, we have found through trial and error that ~500 deployments at one time will almost over utilize the iMed server. In our deployments ~450 clients took 1hour to complete the install. This is due to the nature of the .bat file, the “reinstaller.bat” actually copies the compressed install files to a local %temp% location. The file is ~80mb.

## Building the SMS Collections

Visn7 EST made the decision to advertise the program to one collection the gradually increase the number of client over time. This was the first attempt at this concept, so this procedure is subject to change.

1. Use the queries above to export a list of all machines needing the update.
2. Open the exported list in Excel and remove all column except the computer name column
3. Remove header row

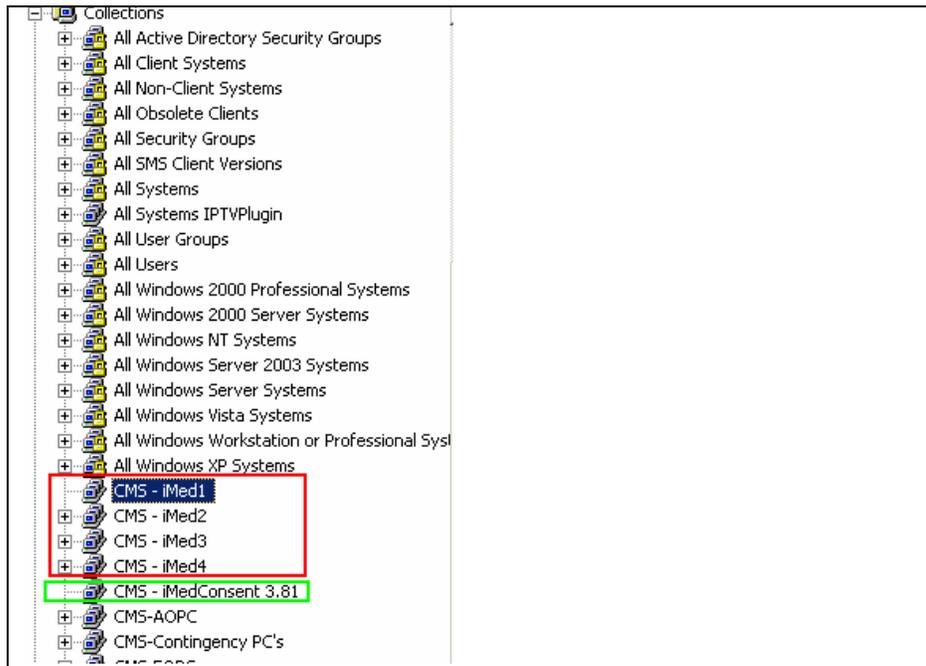
```
CMS-WSEE1014391
CMS-WSEE1010478
CMS-WSEE1014407
CMS-WSEE1014481
CMS-ROCKHILL50
CMS-24BBK21
CMS-WSEE1018008
CMS-WSEE1009227
CMS-WSEE1014444
CMS-WSEE1007903
CMS-WSEE1007955
CMS-WSEE1016422
```

4. In Excel, highlight ~400 machines and save them out to a txt file. (iMed\_XXX1.txt)
5. Repeat until you have several text files containing all the machines needing the iMed Update.
6. Use the "MakeColl.vbs" ([See appendix](#)) to make several collections based on the txt files created above.
  - a. This will need to be run once per text file created to make the collections in SMS.

The following lines in the MakeColl.vbs will need to be edited for your SMS Site Server and collection names... starting around line 20 in the code. Username and Password can be left blank if your logged on account has permissions to SMS.

```
' Constants - change as appropriate  
  
server = "VHAXXSMS1" ' Server name of the SMS site server  
user = "" ' Optional user ID  
password = "" ' Optional password  
gbIPAddress = False ' False = file contains machine names; true = file contains IP addresses  
fileName = "c:\iMed_XXX4.txt" ' File containing the list of machines.  
collectionName = "XXX - iMed4" ' Collection name  
parentCollectionID = "COLLROOT" ' Change this if you want to make a subcollection
```

7. Notice below that you will have a collection for each txt file created earlier. Each collection will contain the list of machines in that txt file. (Red square)
8. Create an "Empty" Collection for the advertisement. This collection will be increased gradually until it contains all other iMed collections. (green square) (No Membership Rules)



## SMS Package

The Visn7 EST Team has written a custom Installer Script to perform the same action that the “reinstaller.bat” (written by Dialog Medical) performs. The Installer Script was written with SMS Installer.

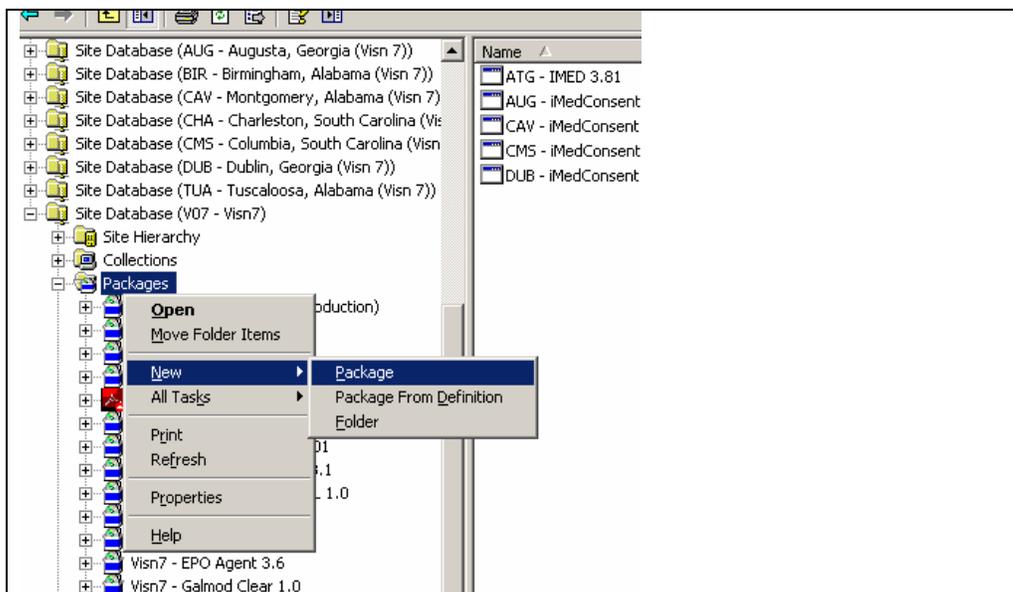
SMS Installer can be downloaded from here:

<http://www.microsoft.com/technet/sms/20/downloads/tools/installer.msp>

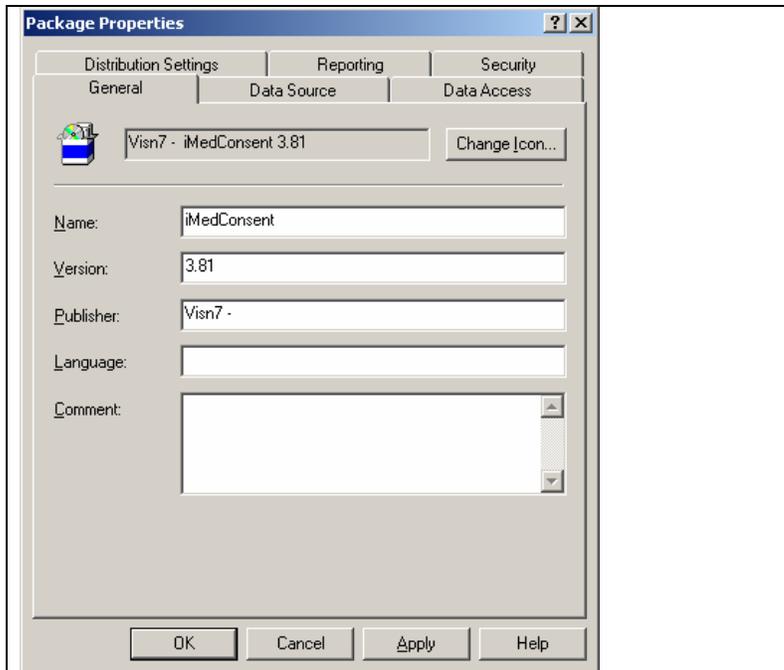
## Building the SMS Package

Inside the SMS Console, create the package to be deployed.

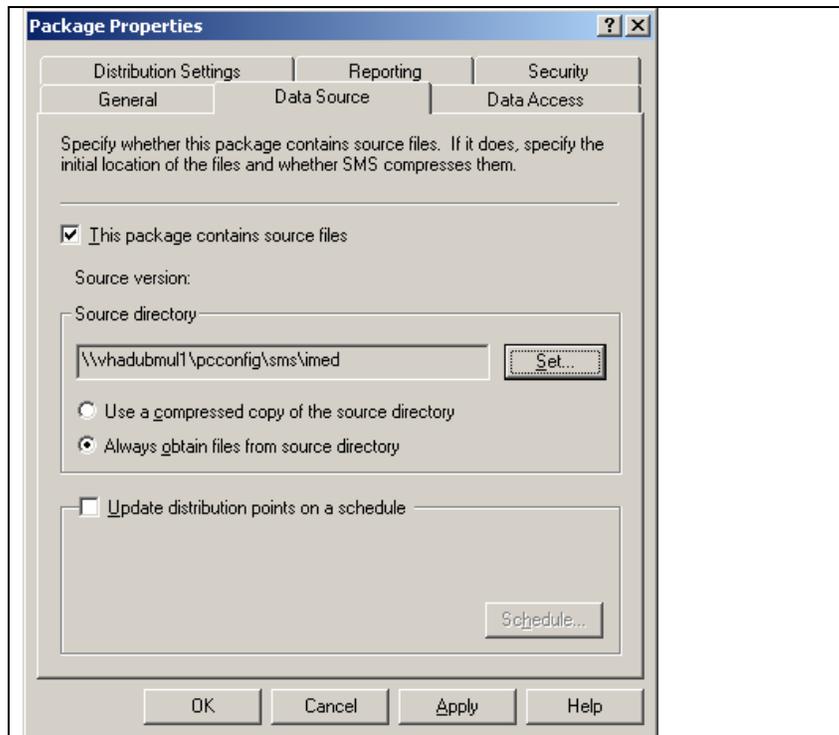
1. Right Click on Packages and select “New\Package”



2. On the “General Tab” fill in the Name, Version, and Publisher information

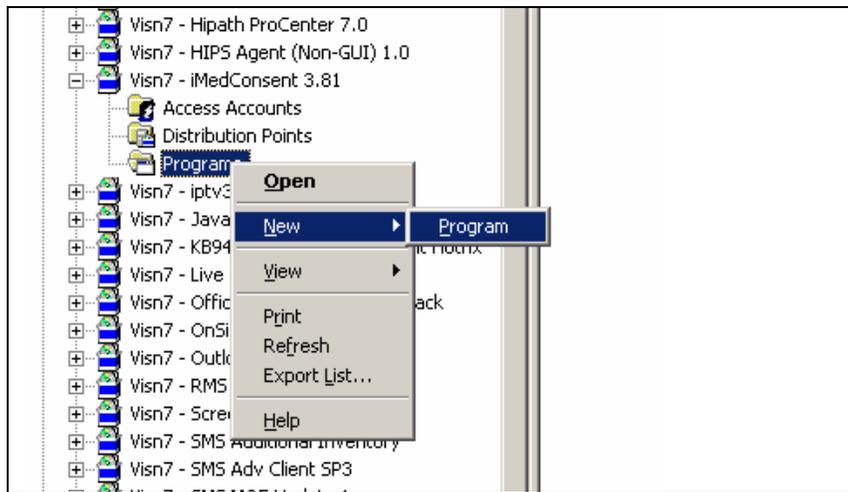


3. On the "Data Source Tab"
  - a. Select **"This package contains source files"**
  - b. Under **"Source Directory"** click set and choose the path to the Custom EXE.

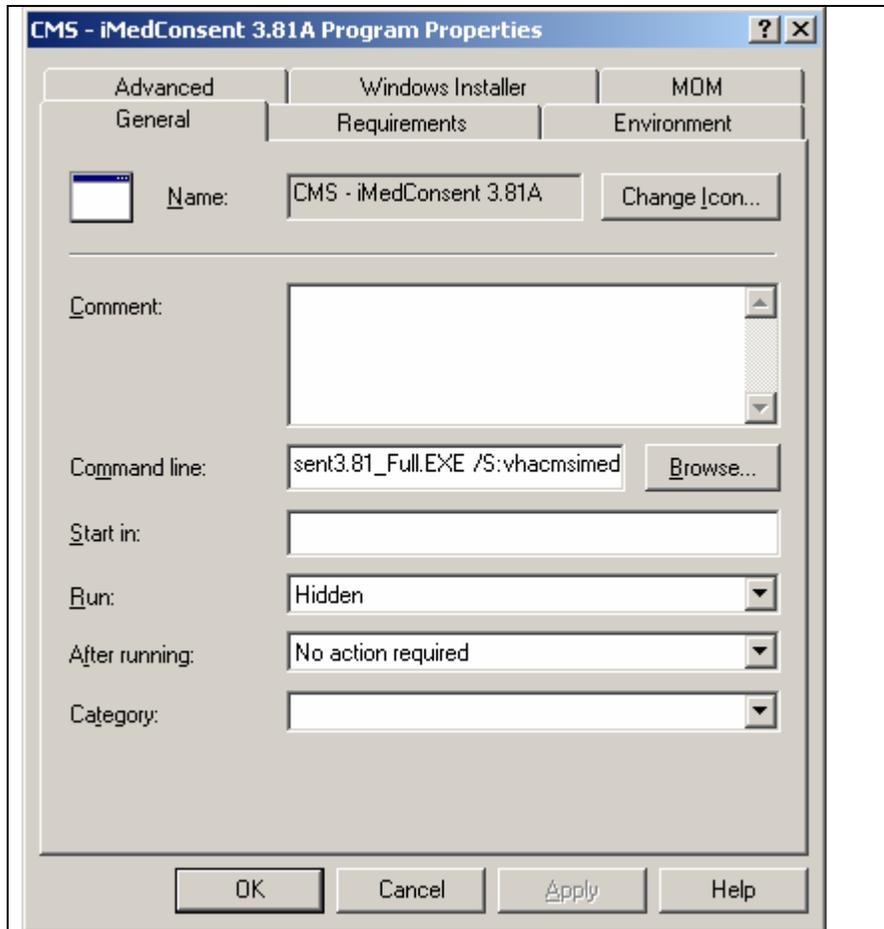


4. Click "Ok" to complete the 1<sup>st</sup> step of the package.

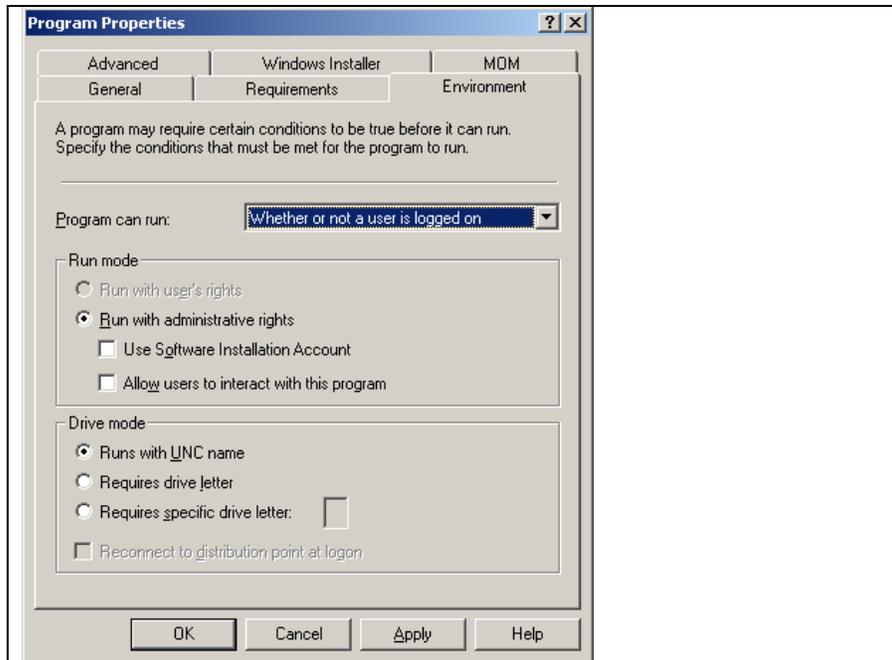
5. Navigate to the Package and click the “+” to expand.
6. Right Click on “**Programs**”
  - a. Select “**New\Program**”



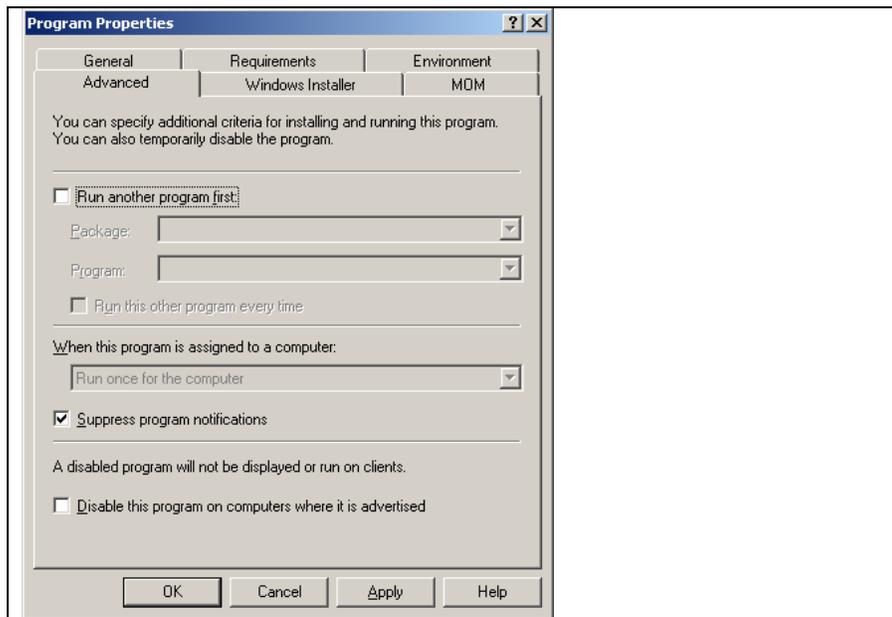
7. On the Program Properties page, fill out the appropriate information
  - a. Name: “**CMS – iMedConsent 3.81**”
  - b. Comment:
  - c. CommandLine: Click Browse and select the Installer Script that was made downloaded earlier.  
“**CMS – iMedConsent3.81\_Full.exe /S:{Your iMedServer}**”  
My Example command line: “**CMS\_iMedConsent\_3.81.exe /S:vhaxximed**”
  - d. Run: “**Hidden**”
  - e. After Running: “**No Action Required**”



8. On the "Environment Tab"
  - a. Program Can Run: **"Whether or not a user is logged on"**
  - b. Run Mode: **"Run with Administrative Rights"**



9. On the “Advanced Tab”
  - a. Select “**Suppress program notifications**”

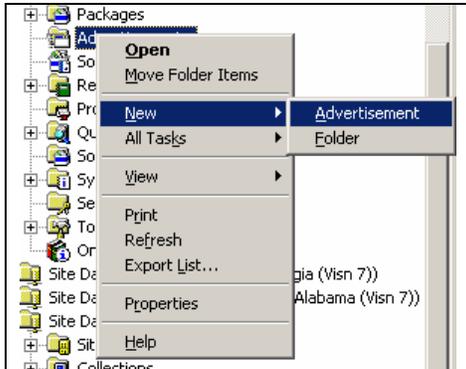


10. Click “**Ok**” to complete the program setup.
11. Send the Program to the Sites Distribution Points

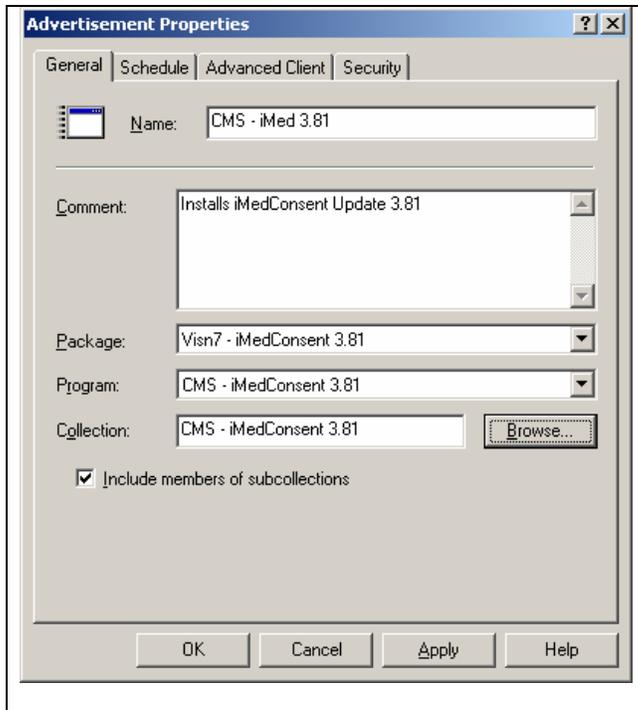
## SMS Advertisement

Create an advertisement to the “Empty” Collection. In our example “CMS – iMedConsent 3.81”

1. In SMS, right click on “Advertisements”
  - a. Select “New”
    - i. “Advertisement”

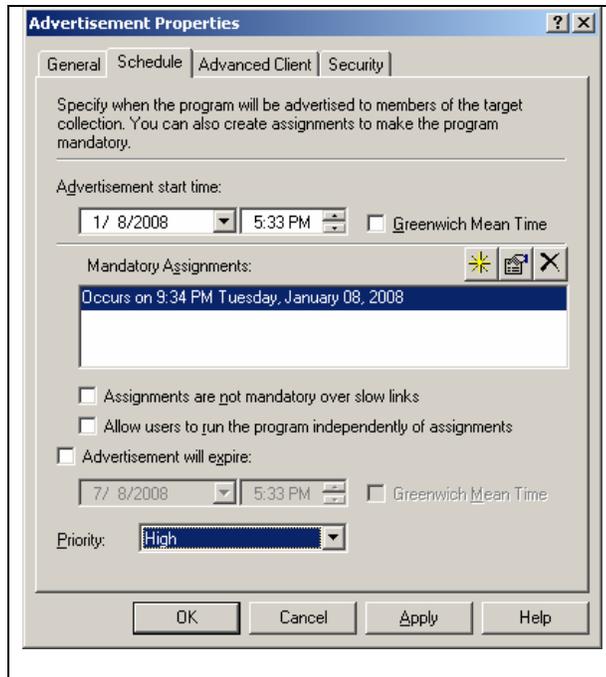


2. In “Advertisement Properties”
  - a. “General Tab”
    - i. Name: “**CMS – iMedConsent 3.81**”
    - ii. Comment:
    - iii. Package: “**Visn7 – iMedConsent 3.81**” (the name of the package you created)
    - iv. Program: “**CMS – iMedConsent 3.81**”
    - v. Collection: “**CMS – iMedConsent 3.81**”

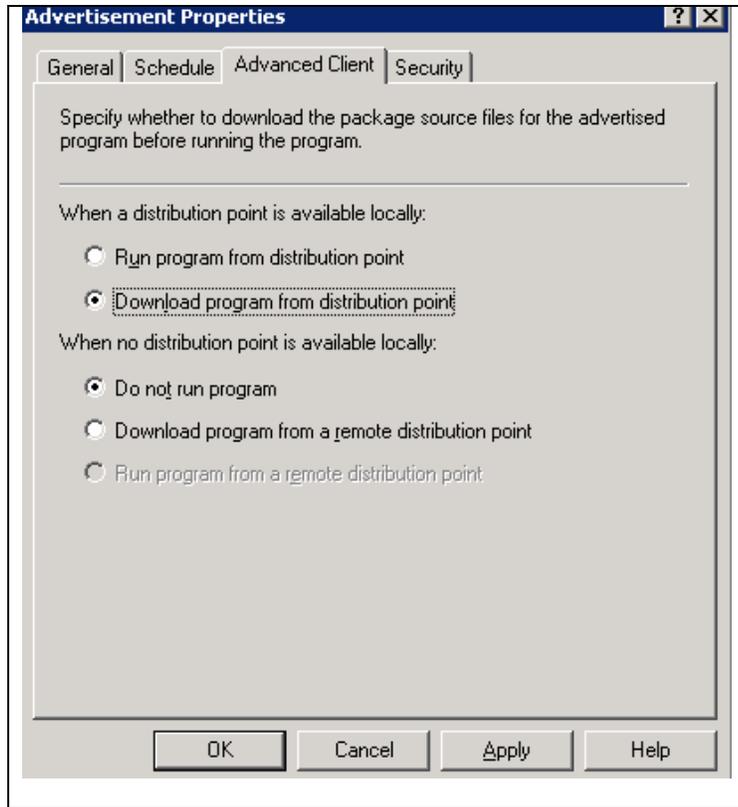


b. "Schedule Tab"

- i. Set "Advertisement Start Time"
- ii. Set "Mandatory Assignment"
- iii. Priority: "High"



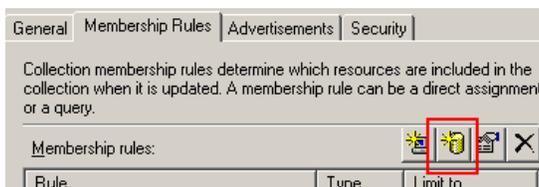
- c. On the "Advanced Client" Tab
  - i. When a distribution Point is available locally: Select "**Download the Package**"  
**\*\* If BITS is enabled, the package is downloaded using BITS**
  - ii. When no distribution point is not available locally: Select "**Do Not Run**"
- d. Click "**OK**" to complete the Advertisement



Everything has been configured and the advertisement is configured for an empty collection.

## Gradually Add Client to the “Empty Collection”

1. Right click the “Empty Collection” (in our example “CMS – iMedConsent 3.81”) and choose properties
2. On the “Membership Tab”
  - a. Select Add Query

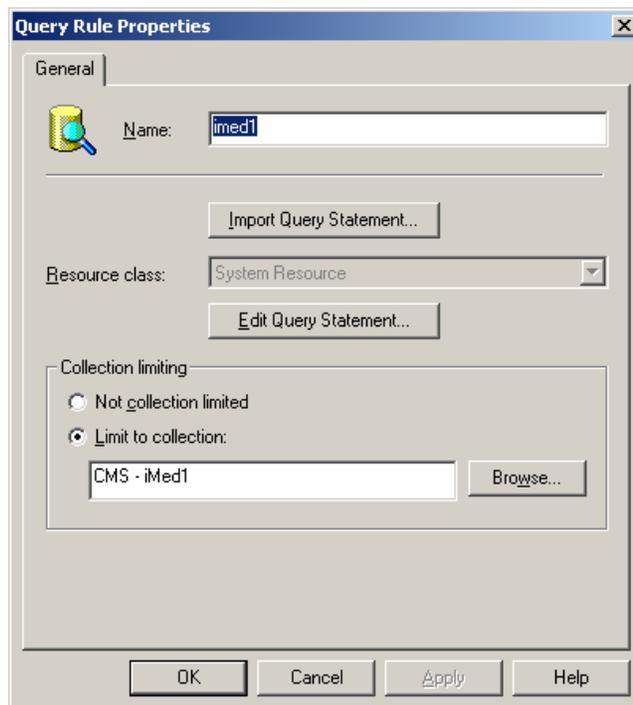


b. Query Rule Properties

i. Name: "imed1"

ii. Collection Limiting: "CMS – iMed1" (*This is the collection containing the first 400 clients*)

iii. Edit Query Statement

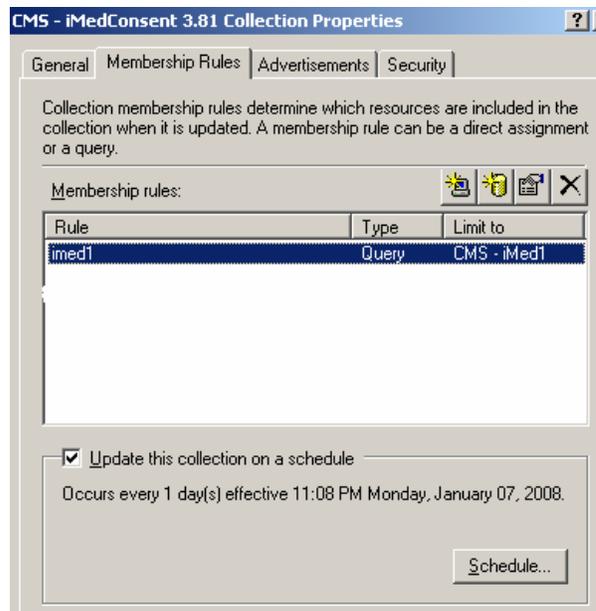
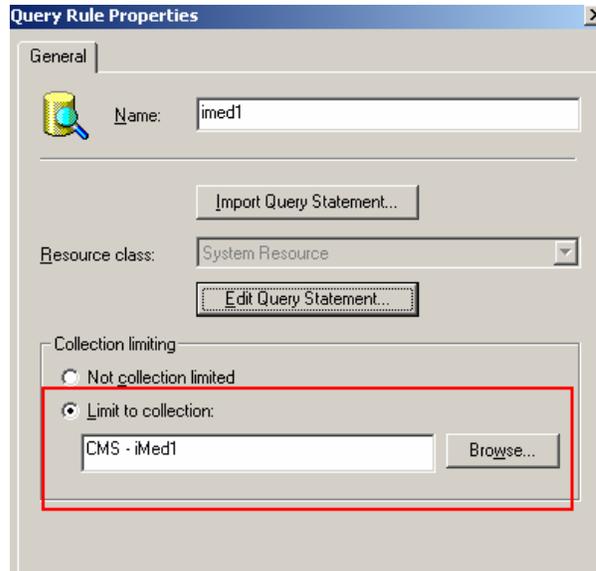


iv. Click "Show Query Language"

v. Query Statement (paste and overwrite with the following)

```
select
SMS_R_System.ResourceID,SMS_R_System.ResourceType,SMS_R_System.Name,SMS_R_System.SMSUniqueIdentifier,SMS_R_System.ResourceDomainORWorkgroup,SMS_R_System.Client
from SMS_R_System inner join SMS_G_System_ADD_REMOVE_PROGRAMS on
SMS_G_System_ADD_REMOVE_PROGRAMS.ResourceID = SMS_R_System.ResourceId
where SMS_G_System_ADD_REMOVE_PROGRAMS.DisplayName like "%iMedConsent
Workstation%" and SMS_G_System_ADD_REMOVE_PROGRAMS.Version != "3.81"
```

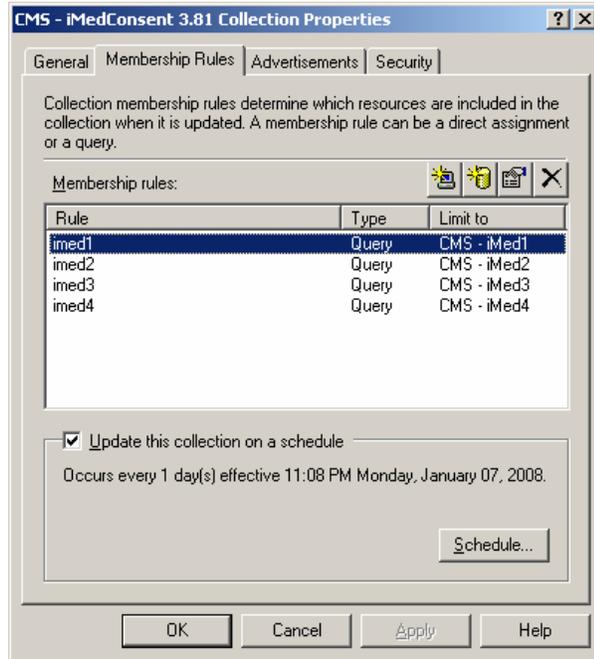
- vi. Click **“Ok”**
- vii. **\*\* Verify “Collection Limiting” \*\***
- viii. Click **“Ok”**



- ix. Click **“Ok”** to finish the collection update
3. Update Collection Membership to reflect new changes.
  4. Show count should reveal only ~400 client (listed in the 1<sup>st</sup> collection)

5. In SMS \System Status\Advertisement Status

- a. Monitor the progress of the deployment. When the deployment gets close to being complete, add another query to the collection limiting it to the 2<sup>nd</sup> collection. Follow the same steps described in this section.



## Appendix

### MakeColl.vbs

```
'=====
' *****
' File:    MAKECOLL.vbs
' Version: 1.1
' Author:  Michael Niehaus
' *****

' The following script is very similar to the MAKECOLL C++ example Microsoft
' provides in the Platform SDK. The main differences:
' 1. This is readable VBScript, not C++/COM gibberish.
' 2. This doesn't include logic to pass in parameters. Instead, edit the
'    constants below.
' 3. No error handling. It clutters up the code.
' Questions or problems? Contact me at "niehaus@mac.com". Thanks.

' *****

' Initialization
' *****

' Constants - change as appropriate

server = "VHAXXSMS1" ' Server name of the SMS site server
user = ""           ' Optional user ID
password = ""       ' Optional password
gbIPAddress = false ' False = file contains machine names; true = file contains IP addresses
fileName = "c:\iMed_XXX4.txt" ' File containing the list of machines.
collectionName = "XXX - iMed4" ' Collection name
parentCollectionID = "COLLROOT" ' Change this if you want to make a subcollection

' *****

' Main code
' *****

' Call the "Init" function to connect to SMS. This returns the "services" object.
Init services, server, user, password

' Create the collection given the collection name and the services object. This
' returns the collection path.
CreateCollection services, collectionName, collectionPath
```

```
' Associate the collection with its parent. Without this step, the collection
' would be orphaned, as you wouldn't be able to see it in the SMS Administrator
' (although it would still exist).
```

```
CreateCollectToSubCollect services, collectionPath
```

```
' Process the file containing the list of machine names or IP addresses. This
' function will then call all the other functions needed to create membership
' rules, add them to SMS, then refresh the collection.
```

```
ProcessListFile services, fileName, collectionPath
```

```
' Done.
```

```
*****
```

```
' Subroutines
```

```
*****
```

```
Sub Init(services, server, user, password)
```

```
    On Error Resume Next
```

```
    ' Connect to the "root\sms" namespace on the specified server to locate the SMS provider,
    ' using the provided server name, user ID and password.
```

```
    Set locator = CreateObject("WbemScripting.SWbemLocator")
```

```
    Set tempServices = locator.ConnectServer(server,"root\sms",user,password)
```

```
    If Err then
```

```
        CheckError
```

```
        Exit Sub
```

```
    End if
```

```
    ' Now figure out the site code for this server.
```

```
    Set result = tempServices.ExecQuery("SELECT * FROM SMS_ProviderLocation WHERE
ProviderForLocalSite=true")
```

```
    For each r in result
```

```
        providerServer = r.Machine
```

```
        namespace = Mid(r.NamespacePath,InStr(3,r.NamespacePath,"\")+1)
```

```
    Next
```

```
    Set tempServices = Nothing
```

```
    ' Finally, connect to the true provider location.
```

```
    Set services = locator.ConnectServer(providerServer,namespace,user,password)
```

```
    If Err then
```

```
        CheckError
```

```
        Exit Sub
```

```
    End if
```

```
    ' We could set context values at this point, but for what we're doing it isn't worth the effort.
```

```
' The Platform SDK sample sets the application name and machine name, which would be seen  
' in the SMS provider log (SMSPROV.LOG) and any status messages generated as a result of this  
' code.
```

```
End Sub
```

```
Sub CreateCollection(services, collectionName, collectionPath)
```

```
    On Error Resume Next
```

```
    ' Spawn an instance of SMS_Collection  
    Set collection = services.Get("SMS_Collection").SpawnInstance_  
    If Err then  
        CheckError  
        Exit Sub  
    End if
```

```
    ' Store the collection name in the collection instance  
    collection.Name = collectionName
```

```
    ' Store the OwnedByThisSite value in the collection instance  
    collection.OwnedByThisSite = true
```

```
    ' Store the collection instance itself in the database.  
    Set pathObj = collection.Put_  
    If Err then  
        CheckError  
        Exit Sub  
    End if
```

```
    ' Get the collection ID  
    collectionPath = pathObj.Path
```

```
End Sub
```

```
Sub CreateCollectToSubCollect(services, collectionPath)
```

```
    On Error Resume Next
```

```
    ' Easy way: retrieve the collection to get its collection ID, instead of  
' parsing the path  
    Set collection = services.Get(collectionPath)  
    If Err then  
        CheckError  
        Exit Sub  
    End if  
    collectionID = collection.CollectionID
```

```

' Spawn an instance of SMS_CollectToSubCollect
Set association = services.Get("SMS_CollectToSubCollect").SpawnInstance_()
If Err then
    CheckError
    Exit Sub
End if

' Store the collection ID
association.subCollectionID = collectionID

' Store the parent ID
association.parentCollectionID = parentCollectionID

' Store the association instance itself
association.Put_
If Err then
    CheckError
    Exit Sub
End if

End Sub

Sub GetResourceIDForNetbiosName(services, netbiosName, resourceID)

    On Error Resume Next

    ' Build Query string.
    query = "SELECT ResourceId FROM SMS_R_System WHERE NetbiosName = "" & netbiosName &
    """"

    ' Execute the query
    Set result = services.ExecQuery(query)
    If Err then
        CheckError
        Exit Sub
    End if

    ' Walk through the elements of the enumerator.
    ' Assume there will only be 1 computer with the requested name, if any.
    resourceID = 0 ' Assume the worst
    For each r in result
        resourceID = r.ResourceID
    Next

End Sub

Sub GetResourceIDForIPAddress(services, IPAddress, resourceID, netbiosName)

```

```
On Error Resume Next
```

```
' Build Query string.
```

```
query = "SELECT ResourceId FROM SMS_R_System WHERE IPAddresses = '' & IPAddress & ''"
```

```
' Execute the query
```

```
Set result = services.ExecQuery(query)
```

```
If Err then
```

```
    CheckError
```

```
    Exit Sub
```

```
End if
```

```
' Walk through the elements of the enumerator.
```

```
' Assume there will only be 1 computer with the requested name, if any.
```

```
resourceID = 0 ' Assume the worst
```

```
For each r in result
```

```
    resourceID = r.ResourceID
```

```
    netbiosName = r.NetbiosName
```

```
Next
```

```
End Sub
```

```
Sub AddMembershipRule(services, rules, resourceID, netbiosName)
```

```
On Error Resume Next
```

```
' Spawn an instance of a direct collection rule.
```

```
Set rule = services.Get("SMS_CollectionRuleDirect")
```

```
If Err then
```

```
    CheckError
```

```
    Exit Sub
```

```
End if
```

```
' Store the resource class name (sms_r_system)
```

```
rule.ResourceClassName = "SMS_R_System"
```

```
' Build a rule name
```

```
rulename = netbiosName
```

```
' Store the rule name
```

```
rule.RuleName = rulename
```

```
' Store ResourceID
```

```
rule.ResourceID = resourceID
```

```
' Add it to the collection
```

```
rules.Add rulename, rule
```

End Sub

Sub AddCollectionMembershipRules(services, collectionPath, rules)

    On Error Resume Next

    ' Get the collection

    Set collection = services.Get(collectionPath)

    If Err then

        CheckError

        Exit Sub

    End if

    ' Add the membership rules

    collection.AddMembershipRules rules.Items

    If Err then

        CheckError

        Exit Sub

    End if

End Sub

Sub UpdateCollectionMembership(services, collectionPath)

    On Error Resume Next

    ' Get the collection

    Set collection = services.Get(collectionPath)

    If Err then

        CheckError

        Exit Sub

    End if

    ' Request the refresh (without subcollections; change to true if you want those)

    collection.RequestRefresh false

End Sub

Sub ProcessListFile(services, filename, collectionPath)

    ' Build a dictionary object to hold the rules. We'll use this as a dynamically-  
    ' sized array.

    Set rules = CreateObject("Scripting.Dictionary")

    ' Open the file

    Set fs = CreateObject("Scripting.FileSystemObject")

```

Set listFile = fs.OpenTextFile(fileName)

' Read the file
While not listFile.AtEndOfStream

    ' Get a line from the file
    line = listFile.ReadLine

    count = count + 1

    ' Call the appropriate GetResourceID function
    If gbIPAddress then
        GetResourceIDForIPAddress services, line, resourceID, netbiosName
    Else
        netbiosName = line
        GetResourceIDForNetbiosName services, netbiosName, resourceID
    End if

    ' Add a rule
    If resourceID > 0 then
        AddMembershipRule services, rules, resourceID, netbiosName
    Else
        WScript.Echo line & " not found on SMS_R_System table"
    End if
WEnd

' Add all the rules to the collection
AddCollectionMembershipRules services, collectionPath, rules

' Update the collection (this should be automatic, but it shouldn't hurt to tell SMS again)
UpdateCollectionMembership services, collectionPath

End Sub

Sub CheckError
    WScript.Echo "An error occurred: " & Err.Description & " (" & Err.Number & ")"
    Set lastError = CreateObject("WbemScripting.SWbemLastError")
    WScript.Echo "WMI error details:"
    For each p in lastError.Properties_
        WScript.Echo "  " & p.Name & " = " & p.Value
    Next
End Sub

```