UBC op Microsoft Windows 64-bits

Inleiding

Op de 64-bits varianten van Windows werkt de UBC (en vele andere pakketten) op een andere manier dan op de “oudere” 32-bits varianten van deze Windows versies. In dit document wordt uitgelegd wat de verschillen zijn en waar op gelet moet worden bij het gebruiken van 32-bits onderdelen op een 64-bits besturingssysteem.

Windows 64-bits

Deze versies van Windows zijn speciaal gemaakt om beter gebruik te kunnen maken van de nieuwere generaties 64-bits processoren. Zo kan b.v. meer geheugen direct worden aangesproken en kunnen bepaalde programma’s die specifiek voor 64-bits processoren zijn gemaakt efficiënter gegevens verwerken.

Omdat het besturingssysteem nu een 64-bits omgeving is kunnen “oude” 32-bits programma’s standaard niet meer uitgevoerd worden. Om dit op te lossen heeft Microsoft in elke 64-bits Windows een 32-bits subsysteem opgenomen waarin deze 32-bits applicaties uitgevoerd kunnen worden. Dit systeem noemt Microsoft “Windows on Windows 64” of kortweg WOW64. Binnen dit gedeelte wordt een “standaard” 32-bits omgeving nagebootst, waarin de eventuele vertalingen voor de 64-bits processor worden uitgevoerd (of wordt de compatibiliteitsmode van de processor gebruikt als deze aanwezig is). Binnen dit systeem valt o.a. ook een eigen (32-bits) “Program Files” map, een eigen (32-bits) “System32” map en een eigen gedeelte van het register dat gebruikt wordt voor de instellingen voor de 32-bits onderdelen.

Vanwege technische beperkingen is het voor een 32-bits applicatie niet mogelijk om 64-bits onderdelen te gebruiken, omgekeerd kan een 64-bits applicatie geen gebruik maken van 32-bits onderdelen.

WOW64

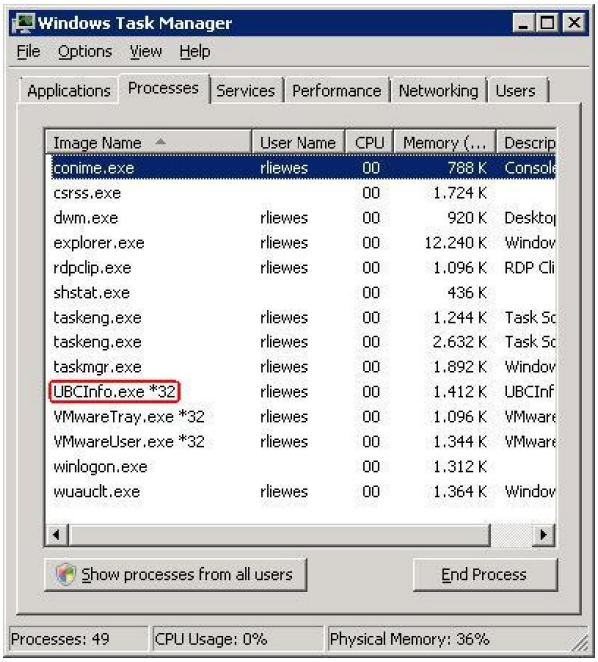
Om vervuiling tussen beide systemen (32-bits en 64-bits) zoveel mogelijk te voorkomen maakt het WOW64 systeem gebruik van een aantal redirections (omleidingen). Als een 32-bits programma wordt gestart dan worden een aantal mappen (onzichtbaar) omgeleid naar andere mappen op het systeem die speciaal voor het 32-bits subsysteem zijn gereserveerd.

De volgende redirections worden door WOW64 uitgevoerd:

Omschrijving	**Standaard (64-bits)**	**WOW64 (32-bits)**
Programma bestanden	Program Files	Program Files (x86)
Systeembestanden	%Systemroot%\System32	%Systemroot%\SysWOW64
Register system-wide	HKLM\Software	HKLM\Software\Wow6432
Register gebruiker	HKCU	HKCU (NIET omgeleid!)

Als in het 32-bits subsysteem b.v. een bestand in de “System32” map wordt weggeschreven of gelezen dan wordt deze door het systeem eigenlijk in de map “SysWOW64” weggeschreven of gelezen.

Programma’s die in het 32-bits subsysteem WOW64 worden uitgevoerd worden in het taakbeheer scherm gemarkeerd met een “*32” achter de naam.

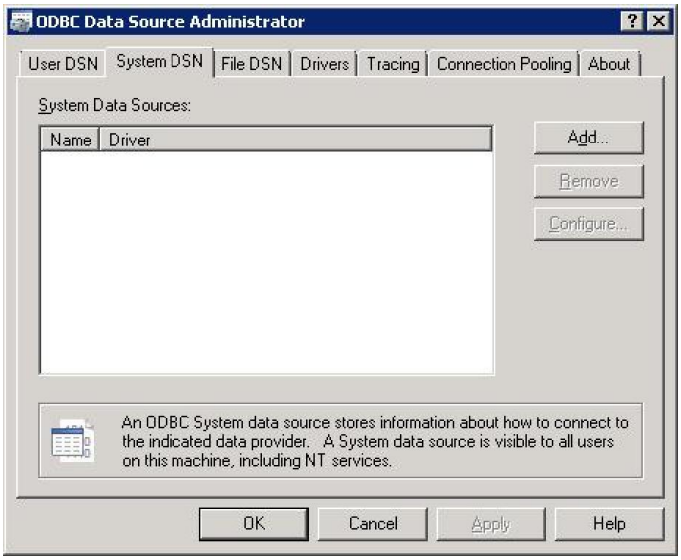


Figuur 1: In het taakbeheer scherm worden 32-bits applicaties gemarkeerd met “ 32”*

Multivers ODBC

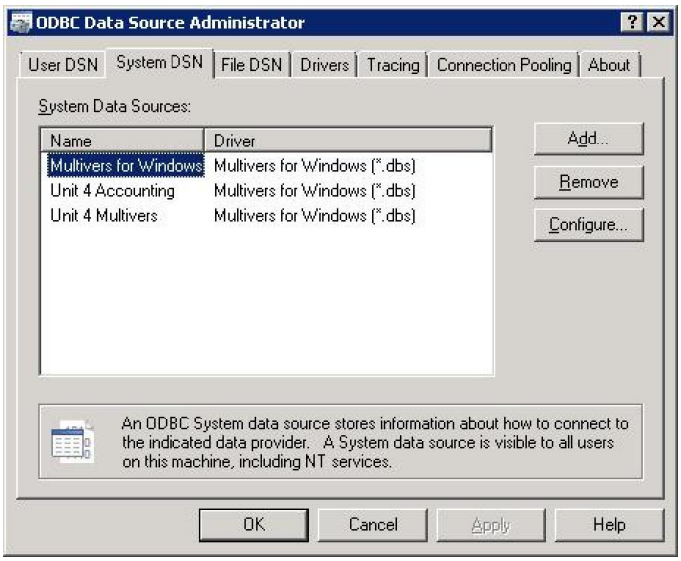
Bij het installeren van Multivers wordt automatisch een ODBC stuurprogramma en een DSN voor de SQLBase database server aangemaakt. Deze ODBC koppeling wordt o.a. ook gebruikt door de UBC. Het ODBC stuurprogramma is echter een 32-bits driver en valt daardoor automatisch onder het 32-bits subsysteem WOW64. Dit betekent dat je vanaf de “standaard” 64-bits omgeving van Windows dit stuurprogramma niet kunt zien. Alle programma’s die net als het stuurprogramma zelf ook onder het WOW64 subsysteem worden uitgevoerd zullen dit stuurprogramma wel gewoon kunnen zien.

Bij een 64-bits versie van Windows worden standaard beide versies van het ODBC administrator programma meegeleverd, alleen wordt via het configuratiescherm standaard de 64-bits versie opgestart ("%Systemroot%\System32\odbcad32.exe"). In deze versie kun je geen (32-bits) DSN's of stuurprogramma's van Multivers zien:



Figuur 2: Multivers DSN's zijn niet te zien in 64-bits versie van odbcad32.exe

Als je de 32-bits versie van het ODBC administrator programma opstart ("%Systemroot%\SysWOW64\odbcad32.exe") dan wordt deze in het WOW64 subsysteem uitgevoerd. Met deze versie zijn het ODBC stuurprogramma en de DSN's voor Multivers wel zien en kun je ook succesvol een testverbinding opzetten:



Figuur 3: Multivers DSN's zijn wel te zien in 32-bits versie van odbcad32.exe

Multivers UBC

Ook de UBC zelf is een 32-bits DLL en valt daarmee ook onder het WOW64 subsysteem. Omdat zowel de UBC als het ODBC stuurprogramma onder WOW64 uitgevoerd worden kan UBC gewoon gebruik maken van het ODBC stuurprogramma.

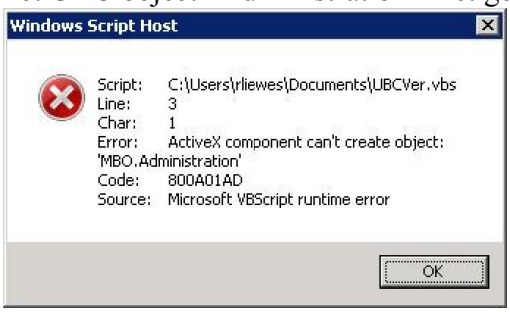
De UBC kan echter uit zichzelf niets doen en zal door een ander programma aangeroepen moeten worden. Ook hierbij is het van belang dat dit door een proces gedaan wordt dat net als de UBC zelf ook onder WOW64 wordt uitgevoerd (een 32-bits applicatie dus). Een applicatie dat in de “standaard” 64-bits omgeving wordt gestart zal de UBC niet kunnen zien omdat de UBC onder het subsysteem WOW64 draait.

En uitzondering hierbij is als de UBC wordt aangeroepen vanuit een script, in dat geval is niet het script, maar het programma dat dit script uitvoert bepalend of deze al dan niet in het WOW64 subsysteem wordt uitgevoerd en dus of de UBC te zien zal zijn vanuit het script.

UBC vanuit VBScript (.vbs bestanden)

Standaard worden bestanden met extensie “.vbs” uitgevoerd door het programma “wscript.exe”. Ook hiervan zijn er in een 64-bits versie van Windows standaard twee versies: een 64-bits versie (“%Systemroot%\System32\wscript.exe”) en een 32-bits versie (“%Systemroot%\SysWOW64\wscript.exe”), waarbij standaard de 64-bits versie wordt gebruikt bij het dubbelklikken op een “.vbs” bestand vanuit de Windows verkenner.

Als er nu een VB script wordt uitgevoerd (met de standaard 64-bits versie van wscript.exe) waarin gebruik wordt gemaakt van de UBC dan krijg je een foutmelding dat het UBC object “Administration” niet gemaakt kan worden:



Figuur 4: UBC kan niet gevonden worden bij het standaard uitvoeren van een VB script.

Om dit te verhelpen moet het script uitgevoerd worden met de 32-bits versie van wscript.exe. Dit kan op verschillende manieren:

1. De 32-bits versie van wscript.exe kan direct aangeroepen worden met als parameter het script dat uitgevoerd dient te worden. In dit geval zal het script de UBC wel kunnen gebruiken:

“%systemroot%\SysWOW64\wscript.exe” “<pad>\<bestandsnaam>.vbs”

2. Door gebruik te maken van een eigen extensie (b.v “vbs32”). Aan deze extensie kan specifiek de 32-bits versie van wscript.exe gekoppeld worden. Nu kan het script dat gebruik maakt van UBC gewoon weer uitgevoerd worden door hierop te dubbelklikken vanuit de Windows verkenner (ten minste als het de juiste, aangepaste, extentie-naam “vbs32” heeft).

Om deze extensie bekend te maken bij Windows moeten de volgende regels worden toegevoegd aan het register:

HKEY_CLASSES_ROOT\.vbs32 (let op de punt “.” voor “vbs32”)
(default) = “vbs32_auto_file”

HKCR\vbs32_auto_file\shell\open\command
(default) = ““%Systemroot%\SysWOW64\wscript.exe” /E:VBscript “%1””

Let op dat hierbij expliciet aan wscript.exe moet worden opgegeven dat het hier een “VB script” betreft omdat deze dat nu niet meer zelf uit de extensie kan opmaken.

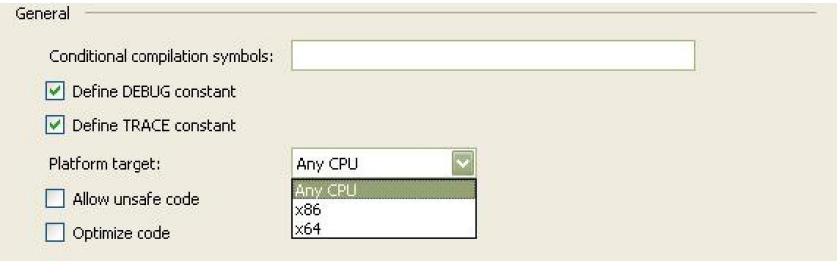
UBC vanuit applicaties

Net als bij scripts moet de applicatie ook onder het (32-bits) WOW64 subsysteem draaien om gebruik te kunnen maken van de UBC.

Hierbij moeten de ontwikkelaars er rekening mee houden dat hun applicatie ook daadwerkelijk in de 32-bits omgeving zal worden uitgevoerd. Voor programmeertalen die een uitvoerbaar bestand maken (een executable of een .exe bestand) is dit meestal een optie bij het compileren. Niet alle programmeertalen hebben al een optie om een 64-bits executables te maken, deze kunnen dan alleen maar 32-bits executables maken (die automatisch in het 32-bits subsysteem zullen worden uitgevoerd). Een voorbeeld van zo’n programeertaal die nog geen 64-bits executables kan genereren is Visual Basic 6.

Voor het .Net platform zijn zowel 64-bits als 32-bits runtime omgevingen beschikbaar. Op een 64-bits versie van Windows zullen doorgaans beide versies van deze run-time omgevingen geïnstalleerd zijn.

Bij het ontwikkelen kan bij een .Net project worden opgegeven welke .Net runtime omgeving (32-bits of 64-bits) gebruikt moet worden voor de betreffende applicatie, hierbij heb je de volgende keuzes:



Figuur 5: Opties in .Net voor het doelsysteem.

Bij de optie “AnyCPU” wordt bij het uitvoeren bepaald welke omgevingen er op het systeem staan. Bij een 32-bits versie van Windows wordt de 32-bits .Net runtime omgeving gebruikt, en op een 64-bits versie van Windows zal de 64-bits versie van de .Net runtime omgeving worden gebruikt. Dit betekent dat op een 64-bits versie van Windows geen gebruik gemaakt zal kunnen worden van de UBC of van de ODBC drivers voor Multivers.

Bij x86 (32-bits) wordt in alle gevallen gebruik gemaakt van de 32-bits versie van de .Net runtime omgeving. UBC en ODBC kunnen zowel op 32-bits versies van Windows als op 64-bits versies van Windows gebruikt worden.

Bij x64 (64-bits) wordt in alle gevallen gebruik gemaakt van de 64-bits .Net runtime omgeving. Als deze niet beschikbaar is (zoals op een 32-bits versie van Windows) dan zal het programma niet uitgevoerd kunnen worden. UBC en ODBC kunnen in alle gevallen niet gebruikt worden.

Hieronder volgt een overzicht van de verschillende opties en de werking van de gegenereerde executable op een 32-bits versie van Windows en op een 64-bits versie van Windows.

Target	32-bits versie van Windows	64-bits versie van Windows
Any CPU	Werkt, met UBC	Werkt, zonder UBC
x86	Werkt, met UBC	Werkt, met UBC
x64	Werkt niet	Werkt, zonder UBC

Huidige 64-bits applicaties

Op dit moment zijn de 64-bits applicaties nog dun gezaaid, het zijn voornamelijk specifieke toepassingen die echt baat kunnen hebben bij het verwerken en verplaatsen van grote hoeveelheden data. Daarnaast worden er voor de meeste applicaties voorlopig ook nog steeds 32-bits varianten geleverd.

Een aantal 64-bits applicaties die al bestaan zijn:

- Microsoft SQL Server 2005
- Microsoft Exchange 2007
- .Net 2.0 runtime (CLR) *
- Internet Explorer *

* Van de .Net 2.0 runtime en Internet Explorer worden standaard zowel de 64-bits als de 32-bits versies meegeleverd met een 64-bits versie van Windows.

Van de .Net 1.0 / 1.1 runtime omgeving is geen 64-bits variant.

Er is sprake van dat er ook een 64-bits versie van Microsoft Office in de maak zou zijn, maar alle versies die tot nu toe beschikbaar zijn (inclusief Office 2007) zijn op dit moment nog 32-bits. Deze zullen dus ook op een 64-bits versie van Windows op het 32-bits WOW64 subsysteem worden uitgevoerd.

64-bits versie van UBC

Naarmate 64-bits versies van Windows meer en meer gebruikt zullen gaan worden zou het mooi zijn als de UBC ook “direct” vanaf het 64-bits systeem aan te sturen zou zijn. Hiervoor moet er dan wel een 64-bits versie van de UBC komen, en een 64-bits ODBC stuurprogramma voor SQLBase. In alle gevallen zouden deze dan nog steeds naast een “oude” 32-bits UBC en een “oude” 32-bits ODBC stuurprogramma moeten draaien zodat deze zowel vanaf het 64-bits “hoofd” systeem als vanaf het 32-bits subsysteem te benaderen zouden zijn.

De UBC zal hiervoor echter wel omgezet moeten worden naar een andere programmeertaal omdat Visual Basic 6 geen 64-bits componenten kan maken.

Voor het ODBC stuurprogramma zijn we afhankelijk van Unify om een 64-bits versie van de ODBC driver te leveren. Alternatief voor de UBC is om via een andere techniek verbinding te maken met de SQLBase database (b.v. OLEDB).

Conclusie

De UBC werkt in een 64-bits versie van Windows alleen in het 32-bits subsysteem. Zolang er voor gezorgd wordt dat de applicatie of het script dat deze UBC aanstuurt ook binnen dit 32-bits subsysteem werkt dan kan er “normaal” gebruik van de UBC worden gemaakt.

Voor de toekomst kan onderzocht worden of een “universele” UBC kan worden ontwikkeld waarvan zowel een “echte” 64-bits DLL als een 32-bit DLL kan worden gemaakt.