Problem: How can I give a Windows look-and-feel to an ARC/INFO command or to an AML/MENU application that I have written?
Solution: Develop the user interface using Visual Basic (the user interface would collect the input parameters to the command or AML), then execute the ARC/INFO command or run the ARC/INFO AML from within Visual Basic. The procedure below enables you to do the latter.
Visual Basic 6.0 Code:

```
Public Sub ExecArcCmd (CmdLine As String)
*****************************************************************************
' This procedure executes an ARC/INFO Command (or runs an AML) and suspends
' VB execution until it finishes.  Note that this procedure does not return
' anything to indicate whether or not the command (or AML) executed
' correctly.

' An alternative approach to the method employed by this procedure would be
' to use the ARC/INFO Open Development Enironment (ODE).

' Input: CmdLine (a text string to execute at the arc prompt)
'              (e.g., "describe hydro", "&run killcovs.aml")

' Requirements:
'   - Visual Basic 6.0
'   - NT ARC/INFO 7.2.1
'   - The following API function declarations (place in the general
'     declarations section of a code module ... each should be on a single
'     line):
'       - Public Declare Function OpenProcess Lib "Kernel32"
'         (ByVal dwDesiredAccess As Long, ByVal bInheritHandle As Long,
'          ByVal dwProcessId As Long) As Long
'       - Public Declare Function WaitForSingleObject Lib "Kernel32"
'         (ByVal hHandle As Long, ByVal dwMilliseconds As Long) As Long
'   - The following constant declaration (place in the general declarations
'     section of a code module):
'       - Public Const ONEHOUR As Long = 3600000

Dim ArcStart As String
Dim pID As Long
Dim pHandle As Long
Dim retVal As Long

'Set the string variable ArcStart to the location of the ARC/INFO
'executable on your machine ...
ArcStart = Environ("ARCHOME") & "\bin\arc.exe "

'Run the ARC/INFO executable and supply the CmdLine string that was passed
'in to this procedure ... (returns a process ID)
pID = Shell(ArcStart & CmdLine, vbNormalFocus)

'Open the handle to the process just kicked off ...
pHandle = OpenProcess(&H100000, True, pID)

'Wait on that handle (up to one hour) ...
retVal = WaitForSingleObject(pHandle, ONEHOUR)

End Sub
*****************************************************************************
```