Controlling the Agilent ChemStation Software for UV-visible Spectroscopy through Dynamic Data Exchange

Technical Note

Dynamic data exchange (DDE) is a fast and easy way to link independent applications that run on Microsoft® Windows® and NT® operating systems. The two DDE examples in this note show the capabilities of the Agilent ChemStation for UV-visible spectroscopy.

In the first example a small Visual Basic program communicates with the Agilent ChemStation application. In this DDE server situation the Agilent ChemStation can be remotely controlled and monitored by the Visual Basic client.

In the second application the Agilent ChemStation uses Microsoft Excel as a server for tabular data. Here an example is given of how the content of a table window can be transferred to an Excel spreadsheet.

Both examples can be used as a start-up for user applications.

Agilent Technologies
Innovating the HP Way

Introduction

Special UV-visible spectroscopy applications often require special software. To avoid rewriting code and to simplify the generation of applications, this technical note gives examples of how to use the Agilent ChemStation software for UV-visible spectroscopy as a remotely controlled DDE server. This approach allows to use all features of Agilent ChemStation and allows, for example, use Microsoft Visual Basic to generate the user interface and task control. The advantage of this approach is the free choice of the programming language and tools to create an application.

Requirements

As well as Agilent ChemStation a second application is needed. The application that initiates communication is called a client and the application providing the service to the client is called a server. The programmability through macros and the built-in features of Agilent ChemStation mean it can be a client as well as a server in a DDE conversation. The examples in this note show a client and a server approach.

The server example uses a simple command line interface, programmed in Visual Basic 4.0, to control the Agilent ChemStation. Visual Basic is a tool that is often used to create user interfaces. Its graphical form designer offers many user interface elements which are able to communicate with the server application.

The second example shows the Agilent ChemStation in a client situation. The server used is Microsoft Excel. This example requires the advanced software for Agilent ChemStation to generate macro programs. The application sends tabular data to Excel for further evaluation.

Principles of DDE

The DDE of the Microsoft Windows and NT operating systems allows to transfer information between standalone applications. The information interchange is organized in links. Applications can have multiple links. These links can be established in parallel to a single application or even different applications. Due to the direction of the transfer of information two types of links are available: a source link and a destination link. A source link sends information to the linked application and a destination link requests information from the linked application. In a typical DDE application both links are established at the same time: a source link to send information to the application and a destination link to receive feedback from the application.

Two examples are shown below. The first example uses the Agilent ChemStation as a server for a simple command line interface. The DDE communicaton is initiated by the command line window application.

In the second example a DDE link to Excel is established by the Agilent ChemStation. Excel is used as a server for results from the Agilent ChemStation.

Server example—from command line interface to Agilent ChemStation

Modern applications like Microsoft Word, Excel or Agilent ChemStation offer programmability in non-standard macro languages. These macro languages usually allow access to the entire functionality of the application. In addition, user macros can be written for specific tasks.

The first example, written in Visual Basic, is a remote command line interface to Agilent ChemStation. This remote command line interface allows to send all Agilent ChemStation macro commands to a Agilent ChemStation application. In the example code the Agilent ChemStation is automatically started in its offline version, if it's not already running.

The command line interface demonstrates a flexible way to control Agilent ChemStation. In a real application, buttons or other user interface elements could send the commands. Even complex sequences can be handled through this link. The command line window is shown in figure 1.

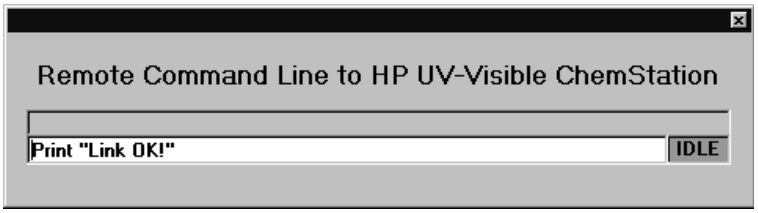


Figure 1
The remote command line window

Explanation

Three links are established by this Visual Basic application: a source link to a text box, and two destination links for the status information and possible error messages generated by the command entry.

The program listing is shown in table 1. The two destination links (Label2, Label3) are set to automatic links (vbLinkAutomatic). The link topics are properties of the label objects (Label2: HPUV-VIS|CPNOWAIT, Label3: HPUV-VIS|SYSTEM) as well as the link items (Label2:

_ErrMsg$, Label3: Status). These links are handled automatically between the two applications. Whenever the status or the content of the error message variable

_ErrMsg$ of the Agilent ChemStation server change, the caption of the linked labels are updated.

The source link to send commands to Agilent ChemStation is handled manually. Only when the Enter key is pressed, is entire line of text send to Agilent ChemStation server through DDE. The link item is the Agilent ChemStation's macro command interpreter. In addition, the current status information is used to check whether the server is busy or not. If the server is busy, no command is send to the server and the text box is not cleared.

Table 1
Listing of the Visual Basic source code of the command line window application

```
VERSION 4.00
Begin VB.Form Form1
   BackColor        =    &H00C0C0C0&
   BorderStyle      =    4  'Fixed ToolWindow
   ClientHeight     =    1620
   ClientLeft       =    1260
   ClientTop        =    1530
   ClientWidth      =    7110
   BeginProperty Font \
     {0BE35203-8F91-11CE-9DE3-00AA004BB851}
      Name             =    "MS Sans Serif"
      Size             =    8.25
      Charset          =    0
      Weight           =    700
      Underline        =    0    'False
      Italic           =    0    'False
      Strikethrough    =    0    'False
   EndProperty
   ForeColor        =    &H00000000&
   Height           =    2025
   Icon             =    "Form1.frx":0000
   Left             =    1200
   LinkTopic        =    "Form1"
   MaxButton        =    0    'False
   MinButton        =    0    'False
   ScaleHeight      =    1620
   ScaleWidth       =    7110
   ShowInTaskbar    =    0    'False
   Top              =    1185
   Width            =    7230
   Begin VB.TextBox Text1
      Height           =    285
      Left             =    180
      LinkTimeout      =    5000
      LinkTopic        =   \
                        """HPUV-VIS|CPNOWAIT"" "
      TabIndex         =    1
      Text             =    "Print ""Link OK!"""
      Top              =    960
      Width            =    6135
   End
   Begin VB.Label Label3
      Alignment        =    2  'Center
      BorderStyle      =    1  'Fixed Single
      Height           =    285
      Left             =    6300
      LinkItem         =    "Status"
      LinkTimeout      =    500
      LinkTopic        =    "HPUV-VIS|SYSTEM"
      TabIndex         =    3
      Top              =    960
      Width            =    615
   End
   Begin VB.Label Label2
      BorderStyle      =    1  'Fixed Single
      Height           =    255
      Left             =    180
      LinkItem         =    "_ErrMsg$"
      LinkMode         =    3  'Notify
      LinkTimeout      =    500
      LinkTopic        =    "HPUV-VIS|CPNOWAIT"
      TabIndex         =    2
      Top              =    720
      Width            =    6735
   End
   Begin VB.Label Label1
      Alignment        =    2  'Center
      Caption          =    \
            "Remote Command Line to " + \
            " HP UV-Visible ChemStation"
      BeginProperty Font \
        {0BE35203-8F91-11CE-9DE3-00AA004BB851}
         Name             =    "MS Sans Serif"
         Size             =    12
         Charset          =    0
         Weight           =    700
         Underline        =    0    'False
         Italic           =    0    'False
         Strikethrough    =    0    'False
      EndProperty
      Height           =    375
      Left             =    180
      TabIndex         =    0
      Top              =    240
      Width            =    6735
   End
End
Attribute VB_Name = "Form1"
Attribute VB_Creatable = False
Attribute VB_Exposed = False

Private Sub Form_Load()

   On Error GoTo LoadApplication
      Label2.LinkMode = vbLinkAutomatic
                        ' initiate link error
      Label2.Caption = ""
                        ' clear error
      Label3.LinkMode = vbLinkAutomatic
                        ' initiate link status

Exit Sub

LoadApplication:
                        ' trap application not
   On Error GoTo 0
                        ' started, reset trapping
   h = MsgBox \
         ("Starting UV-Visible Chemstation", 65)
                        ' allow user to exit
   If h = 1 Then
      h = \
      Shell("C:\HPCHEM\SRELEASE\HPUV-VIS 1", 4)
                        ' Start ChemStation.
      Call Sleep(90000)
                        ' wait 90 seconds
   Else
      End
   End If
   Resume               ' continue

End Sub

Private Sub Form_Unload(Cancel As Integer)

      Text1.LinkMode = vbNone
      Label2.LinkMode = vbNone
      Label3.LinkMode = vbNone
```

```
End Sub

Private Sub Label2_Change()

        h = Len(Label2.Caption)
        If h > 2 Then
            Label2.BackColor = &HFF&
        End If

End Sub

Private Sub Label2_LinkOpen(Cancel As Integer)

      Label2.Caption = ""    ' reset

End Sub


Private Sub Label3_Change()

     If InStr(1,Label3.Caption,"BUSY") > 0 Then
         Label3.BackColor = &HFF0000
         Label3.ForeColor = &HFFFFFF
     Else
         Label3.BackColor = &HFF00&
         Label3.ForeColor = &H0&
     End If

End Sub

Private Sub Label3_LinkOpen(Cancel As Integer)

      Label3.Caption = ""    ' reset

End Sub

Private Sub Text1_KeyPress(KeyAscii As Integer)

If KeyAscii = 13 Then        ' Enter ?
   Label2.Caption = ""       ' reset
   Label2.BackColor = &HC0C0C0
   If InStr(Label3.Caption, "IDLE") > 0 Then
        If Text1.LinkMode = vbNone Then
                          ' not yet established ?
            Text1.LinkTopic = \
                             "HPUV-VIS|CPNOWAIT"
            Text1.LinkMode = vbLinkManual
        End If
        Text1.LinkExecute Text1.Text
                          ' Set link item.
        Text1.Text = "" ' reset command line
   Else
        Beep              ' cannot execute
   End If
End If

End Sub
```

Discussion

The command line window example in Visual Basic shows the principle of how Agilent ChemStation can be used as a server to a user application. To help customizing these special applications, a few useful macro commands are listed in table 2.

When using these macros care must be taken of the correct order and appropriate status of the spectrophotometer and the application. For example, before any sample or standard measurement, a blank must be measured. Or, in quantitative analysis, the system must be calibrated before samples can be analyzed. The simplest way to achieve such conditions is to set up the method according to the applications requirements.

These macros use the current parameter of the method. In addition to the system macros, user macros can be created to further simplify the programming effort.

Table 2
Example macros for selected tasks
(S = standard tasks, A = advanced, D = dissolution)

Task	Commands	Application
Blank measurement	MeasureBlank()	S,A,D
Sample measurement	MeasureSample()	S,A,D
Standard measurement	MeasureStd()	S,A,D
Auxilliary measurement	MeasureAux()	S,A,D
Control measuement	MeasureControl()	
Change application	zzswGUISwitchMode "<Mode name>"	all
Disable userinterface	__busy=1	S,A,D
	zzswCheckButtons	
	MenuDelete	
	zzDisableFKeys	
Enable userinterface	zzBaseMenu	S,A,D
	zzStartMenu	
	zzGenericMenuPart "UserContrib"	
	__busy=0	
	zzswCheckButtons	
Load samples	LoadSamples	S,A,D
Load standards	LoadStandards	S,A,D
Load auxiliary	LoadAuxiliary	S,A,D
Save samples	SaveAsSamples	S,A,D
Save standards	SaveAsStandards	S,A,D
Print results report	zzrpPrint("Results")	A,D
View results report	zzrpPrint("Results","SCREEN")	A,D
Print method	zzrpPrint("Method")	A,D
View method	zzrpPrint("Method" ,"SCREEN")	A,D
Load method	LoadMethod()	S,A,D

Client example—transfer of tabular Agilent ChemStation data to Excel

This Agilent ChemStation macro uses Excel as a server. Agilent ChemStation initiates the DDE conversation and transfers data to spreadsheets. In the example macro code shown in table 3, a simple user interface was added to check and demonstrate the macros. The data of a table displayed in Agilent ChemStation windows can be transferred to Excel using the **Export** menu's **Export selected table to Excel** command. The macros used with this example can be integrated into automated sequences.

Explanation

To transfer the data of an existing table only two macro functions are required: DDE_Excel_Init() and SendTable().

The macro function DDE_Excel_Init() establishes a DDE link to the specified spreadsheet. In addition this macro function starts an Excel session, if Excel was not already running. The path required to start Excel is stored in Agilent ChemStation's Features.ini file. Either a text editor or the comand SetPrivateProfileString can be used to set a valid Excel path.

Table 3
Listing of macro file DDE_Excel.MAC

```
!*****************************************************************************
!
!    FILE
!         DDE_Excel.MAC
!
!    DESCRIPTION
!         Macros for sending tables to MS-Excel an spreadsheet.
!
!
!    Version Rev 1.0   23.11.98   Thomas Klink
!
!
!*****************************************************************************


!*****************************************************************************
!                          E X P O R T S
!*****************************************************************************
!    RunExcelMacro    SysChannel,           Send run macro instruction to MS-Excel
!                      MacroName$,           Spreadsheet via SysChannel
!                      [SpreadSheet$]
!    MenuExport                             Add entry to Export menu
!    ExportSelectedTable                    Get and check for valid selection
!                                             of table and export to MS-Excel
!    DDE_Excel_Init   [SpreadSheet$]        Initialize transfer to MS-Excel
!    SendColumn       RegObj$,TabName$,     Send table column to  MS-Excel
!                      ColumnName$,          spreadsheet
!                      DDE_Chan,StartRow,
!                      Col
!    SendTable        RegObj$,TabName$,     Send table to MS-Excel spreadsheet
!                      DDE_Chan,[StartRow],
!                      [StartCol]
!    SendVal          Val,DDE_Chan,         Send value to spreadsheet
!                      Row,Col
!    SendText         Text$,DDE_Chan,       Send string to spreadsheet
!                      Row,Col
!
!


!*****************************************************************************
!                          M A C R O S
!*****************************************************************************


!*****************************************************************************
NAME RunExcelMacro

   Parameter SysChannel                              ! active DDE channel
   Parameter MacroName$                              ! run macro
   Parameter SpreadSheet$      Default ""            ! active spreadsheet


!    DESCRIPTION
!         Send run macro instruction to MS-Excel Spreadsheet via SysChannel
!
!
!  RETURN
!      True (1), no error, False(0), error
!*****************************************************************************

    On Error Return 0

    If SpreadSheet$ = "" then
```

```
        DDEExecute SysChannel,"[run("""+MacroName$+""")]"
    else
        DDEExecute SysChannel,"[run("""+SpreadSheet$+"!"+MacroName$+""")]"
    Endif

    Return 1

EndMacro


!************************************************************************
Name MenuExport


!   DESCRIPTION
!       Add entry to Export menu
!
!   RETURN
!       None
!
!************************************************************************

        MenuAdd "E&xport",,"SEPARATOR"
        MenuAdd "E&xport", "Export selected Table to Excel",ExportSelectedTable

EndMacro

!*****************************************************************************
NAME ExportSelectedTable

!   DESCRIPTION
!       Get and check for valid selection of table and export to MS-Excel
!
!
!  RETURN
!     None
!*****************************************************************************

    LOCAL Chan,LastRow,Win,x
    LOCAL RegObj$,TabName$,C$

    Win = ActiveWindow()                              ! get current selection
    IF Win = 0 THEN                                    ! check
        PRINT #2, "Select/activate a table window!" ! nothing selected
        RETURN
    ENDIF

    L$ = TabText$(_CONFIG[1],"WINDOW",Win,"Command")! get command to restore
    if Len(L$) > 0 then                                ! not empty ?
        C$ = zzrpGetNextParam$("L$")                  ! get command keyword
        if C$="EdTab" OR C$="edtab" OR C$="EDTAB"  then
            RegObj$   = zzrpGetNextParam$("L$")       ! ignore first (WinNr)
            RegObj$   = ConvertText$("TRIM",zzrpGetNextParam$("L$"))  ! get RegObj
            TabName$ = ConvertText$("TRIM",zzrpGetNextParam$("L$"))  ! get TabName

            Chan = DDE_Excel_Init()                   ! open DDE communication
            Print "Sending table "+RegObj$+","+TabName$+" to Excel"  ! message
            LastRow = SendTable(RegObj$,TabName$,Chan)                ! send table
            DDEterminate Chan                          ! close DDE channel
            Return                                      ! ready!, exit
        EndIf
     EndIf
     x = Alert("Could not export:"+Chr$(13)+Chr$(10)+\
                L$+Chr$(13)+Chr$(10)+"to Excel!",3,"Export to Excel")


ENDMACRO
```

```
!************************************************************************
Name DDE_Excel_Init

   Parameter SpreadSheet$       Default ""            ! active spreadsheet

!   DESCRIPTION
!        Initialize transfer
!
!
!   RETURN
!        Channel, -1 = Error
!************************************************************************

    Local Chan,h,i,x
    Local Topics$,Topic$

    On Error Goto StartExcel
    x = 0

Again:
    Chan=DDEinitiate("Excel","System")
    DDErequest Chan,"topics",Topics$
    DDETerminate Chan

    If  SpreadSheet$ = "" then
        Topics$ = zzrpReplace$(Topics$,Chr$(9),"|")
        i = Instr(Topics$,"|System")
        Topics$ = Topics$[1:i-1]
        Topic$ = SubString$(Topics$,1)

        RemoveDialog ExportToExcel

        BeginDialog ExportToExcel, 50, 40,120, 80,"Send selected table to Excel"
            StaticText       10, 12, 80, 20, "Destination sheet:"
            ComboBox         10, 25,100, 80, Topics$,Topic$
            OkButton         10, 55, 40, 16
        CancelButton     70, 55, 40, 16
        EndDialog

        h = ShowDialog(ExportToExcel)
        RemoveDialog ExportToExcel

        if h = 0 then                                       ! cancel ?
            Return -1                                       ! exit
        EndIf
    Else
        Topic$ = SpreadSheet$
    EndIf

    Chan=DDEinitiate("Excel",Topic$)
    DDEexecute Chan,"[activate("""+Topic$+""")]"

    Print "Connecting to " + Topic$

Return Chan

StartExcel:

    If _Error = 41330 and x = 0 then                 ! Excel not running
        x = 1                                          ! first attempt
        Print "Trying to start Excel..."              ! display message
        ExcelPath$ = PrivateProfileString$("ExportToExcel","ExcelPath",\
                     _ExePath$+"features.ini","C:\MSOffice\Office")
        ExecNoWait ExcelPath$ + "\Excel.exe"
        Sleep 10
        Goto Again
```

```
    Else
        On Error                                        ! reset error trapping
        If _ErrCmd$ = "ExecNoWait" Then
            x = Alert("Could not start Excel!"+Chr$(13)+Chr$(10)+\
                      "Path : "+ExcelPath$+" correct?"+Chr$(13)+Chr$(10)+\
                      "Use 'Configure Excel path...' command to set."\
                      ,3,"Export to Excel")
        Else
            x = Alert("Error in macro DDE_Excel_Init: "+_ErrMsg$,3,"Export to Excel")
        EndIf
        Return -1                                       ! could not open channel
    EndIf

EndMacro

!*****************************************************************************
NAME SendColumn

    Parameter RegObj$                                  ! table register object
    Parameter TabName$                                 ! table name
    Parameter ColumnName$                              ! column name
    Parameter DDE_Chan                                 ! active DDE channel
    Parameter StartRow                                 ! spreadsheed start row
    Parameter Col                                      ! spreadsheet column


!   DESCRIPTION
!       Send table column to Spreadsheet via DDE_Chan
!
!
!  RETURN
!       None
!*****************************************************************************

    Local C$, H$,X$
    Local h,i,t,x


    C$ = "c"+Val$(Col)                                 ! initialize column id
    DDEpoke DDE_Chan,"r"+val$(StartRow)+C$,ColumnName$

    Evaluate "h =  TabHdrVal("+RegObj$+",TabName$,""NumberOfRows"");"+\
             "t =  TabColType("+RegObj$+",TabName$,ColumnName$)"

    If t = 0 then                                      ! string ?
        Evaluate "For i = 1 to h;"+\
                 "X$ = TabText$("+RegObj$+",TabName$,i,ColumnName$);"+\
                 "DDEpoke DDE_Chan,""r""+Val$(StartRow+i)+C$"+",X$;"+\
               "Next i"
    EndIf

    If t > 0 then                                      ! number ?
        Evaluate "For i = 1 to h;"+\
                 "x = TabVal("+RegObj$+",TabName$,i,ColumnName$);"+\
                 "if x > -1.7977e+308 then;"+\
                     "ddepoke DDE_Chan,""r""+Val$(StartRow+i)+C$"+\
                                       ",SPrintF$(x,""%2.21g"");"+\
                 "else;"+\
                     "ddepoke DDE_Chan,""r""+Val$(StartRow+i)+C$"+\
                                       ","""***""";"+\
                 "endif;"+\
                "Next i"
    EndIf

EndMacro
```

```
!*****************************************************************************
NAME SendTable

    Parameter RegObj$                                  ! table register object
    Parameter TabName$                                 ! table name
    Parameter DDE_Chan                                 ! active DDE channel
    Parameter StartRow  Default 1                     ! spreadsheed start row
    Parameter StartCol  Default 1                     ! spreadsheed start column


!   DESCRIPTION
!       Send table to  Spreadsheet via DDE_Chan
!
!
!  RETURN
!     True (1), no error, False(0), error
!*****************************************************************************


    Local ColumnName$
    Local h,Row,Col

    Row = StartRow                                     ! start row in spreadsheet
    DDEpoke DDE_Chan,"r"+Val$(Row)+"c"+Val$(StartCol),"Table: "+RegObj$+","+TabName$
    Row = Row + 2                                      ! increment row
    Col = 1                                            ! start with first column

    Evaluate "Repeat;"+\
                  "ColumnName$=TabColName$("+RegObj$+",TabName$,Col);"+\
                  "If ColumnName$ <> """ then;"+\
                       "SendColumn RegObj$,TabName$,ColumnName$,DDE_Chan,Row,StartCol+Col-1;"+\
                       "Col=Col+1;"+\
                  "EndIf;"+\
             "Until ColumnName$ = """;"+\
             "Row = Row + TabHdrVal("+RegObj$+",TabName$,""NumberOfRows"") + 2"

    Return Row

EndMacro


!*****************************************************************************
NAME SendVal

    Parameter Val                                      ! value to transfer
    Parameter DDE_Chan                                ! open DDE channel
    Parameter Row                                      ! spreadsheed  row
    Parameter Col                                      ! spreadsheed  column


!   DESCRIPTION
!       Send value to spreadsheet via DDE_Chan
!
!
!  RETURN
!     None
!*****************************************************************************


    DDEpoke DDE_Chan,"r"+Val$(Row)+"c"+Val$(Col),Val$(Val)

EndMacro
```

```
!*******************************************************************************
NAME SendText

    Parameter Text$                                    ! string to transfer
    Parameter DDE_Chan                                 ! open DDE channel
    Parameter Row                                      ! spreadsheed  row
    Parameter Col                                      ! spreadsheed  column


!   DESCRIPTION
!       Send string to spreadsheet via DDE_Chan
!
!
!  RETURN
!     None
!*******************************************************************************


    DDEpoke DDE_Chan,"r"+Val$(Row)+"c"+Val$(Col),Text$

EndMacro
```

For example, the path C:\MSOffice\Office can be set using:

```
SetPrivateProfileString
"ExportToExcel",\
"ExcelPath",
"C:\MSOffice\Office",\
_ExePath$+"Features.ini"
```

After successful completion of the DDE_Excel_Init macro function a DDE channel number is available for the data transfer. Even multiple channels in parallel can be used to send tables to different spreadsheets.

The information of an entire table is transferred by the 'SendTable' macro function. Only the register where the table resides, the table name and the DDE transfer channel have to be specified to send all table data. Optionally the starting row and column can be specified. If multiple tables shall be transferred, the return value of the SendTable function provides the starting row for the next table.

If only certain columns of a table shall be transferred, the macro command SendColumn can be applied. This macro transfers a single column only. In addition to the parameters of the SendTable function, the column name, the start row index and the column index have to be specified.

For other information the two macro commands SendVal and SendText can be used. They allow to set the content of a single spreadsheet cell with a numerical value or a text string.

To use the functionality of Excel, the RunExcelMacro function is provided. The macro function allows to execute macros created with a spreadsheet. Assuming you recorded your first Macro1 with the Excel macro recorder, you can execute this macro using:

```
Print RunExcelMacro
(DDE_Chan,"Macro1")
```

This command requires the Excel Macro1 to be available with the currently-used DDE channel (DDE_Chan).

To release a DDE channel, the command DDEterminate is available.

Conclusion

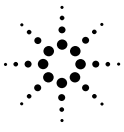
The three applications Microsoft Visual Basic, Microsoft Excel and the Agilent ChemStation for UV-Visible spectroscopy have been used to demonstrate how independent applications can be linked for remote control and data interchange. The examples given are meant as a starting point for creation of complex applications by linking specialized applications together. DDE is not limited to these applications only. Even two instrument sessions in the Agilent ChemStation world can be linked—spectral data can be sent from a chromatographic run to a spectroscopy session where they can be further analyzed.

References

A *Macro Programming Guide* is provided in Adobe® Acrobat® Portable Data Format (PDF) with the Advanced Agilent ChemStation software for UV-Visible spectroscopy. It is a good starting point to write macros for Agilent ChemStation. The command syntax is available in Agilent ChemStation help files. Macro examples are provided with Agilent ChemStation family software products disk. In the directory Ucl\, examples for the various Agilent ChemStation products can be found.

For the latest information and services visit our world wide web site:
http://www.agilent.com/chem

Publication Number 5968-3616E



Agilent Technologies
Innovating the HP Way