Dynamic Data Exchange

By Nick Decker

Overview

DDE stands for Dynamic Data Exchange. As the name suggests, it is a method for Windows applications to communicate to one another and share information in a dynamic fashion. It can be thought of as a conversation between two programs. The first program will initiate the conversation by asking a question and the second will oblige by sending the answer. The conversation ensues until one of the applications closes the connection.

Uses of DDE

DDE is usually used to share data between two applications, but can also be utilized to send commands to another application. For example, you could use DDE to allow Visual PRO/5 to get the year-to-date sales from an Excel spreadsheet. Another use would be the manipulation of the Windows Program Manager groups. DDE will allow Visual PRO/5 to create and delete groups, as well adding, deleting, and modifying programs within the group. Another possible use would involve Visual PRO/5 starting up Microsoft Word, running a macro, then retrieving the pertinent information for use in the program. A Visual PRO/5 program can be written to act as a DDE client to any DDE server application running on any machine on the network. A few of the applications that support DDE under Windows are Microsoft's Word, Excel, Visual Basic, and the Program Manager, as well as other programs such as JSB's MultiView.

Windows, Client, and Server

When an application initiates the DDE process, this application is said to be the *client* because it requests information from another application. The other application is called the *server* because it provides its services to a DDE client. Despite what the name implies, a client and server communicating through DDE never talk directly to each other. Rather, they send messages to Windows which passes the information on to the correct destination. Because of this, the world of DDE encompasses entire networks, and is not limited to a single workstation. With DDE it is possible for a Windows application on one machine to send data to an application that is running on another workstation, providing they are on the same network. Additionally, it is possible for a single application to participate in multiple DDE conversations simultaneously. This flexibility allows for a Visual PRO/5 program to communicate with many other programs - both as a DDE Server and as a DDE Client.

Protocol

As with most communications, a set of rules apply to determine how and when data is sent. These rules determine the protocol that is used to govern the exchange of data. DDE requires its own protocol to oversee the communication between applications. This protocol outlines the procedures used to start and end a DDE session; send data to another application; receive data from another application; and instruct another application to execute a macro or command. There are relatively few parameters that constitute this protocol, making DDE communications fairly elementary.

Conversation Elements

Before you attempt to perform a DDE conversation between Visual PRO/5 and another application, there are a few essential communication elements that have to be determined first. These components specify the application you want to converse with as well as the subject, or data, that you are interested in sharing.

Application: The program that Visual PRO/5 wants to converse with. In order to initiate a conversation with another application, you have to know the name of the other program. This is the name of the other executable, minus the **.exe** extension. As an example, if you were opening a DDE link with the Windows Program Manager, the application name would be PROGMAN. If the other application is not running, you will not be able to connect to it and most likely will get an !ERROR=70. It is the programmer's responsibility to ensure that the other application is running. If it is not, you should invoke it with the SCALL() function.

Topic: This is the data topic that is designated for DDE access from the DDE Server application. This field is predetermined by the server application as an object that it would like to share. For example, if you were going to communicate with an Excel spreadsheet via DDE, the topic of the conversation would be the name of the spreadsheet file with which you wish to exchange data.

Item: This is the specific piece of information that you are trying to read. Using the Excel example, the item would be the specific cell in the worksheet. If you wanted to find out what programs were defined in the Accessories group in the Windows Program Manager, you would use the group name of accessories for the item.

In order to use DDE successfully, you must have all the information pertaining to the remote application ahead of time. In other words, you cannot carry out a DDE conversation with another application unless you know the topics that are shared and the nature of the items. This can limit the usefulness and capabilities that DDE provides. For example, you may want to get information from your Microsoft Excel spreadsheet, but without knowing what Excel expects for DDE topics and items you can not communicate with it at all.

Usage

DDE conversations with Visual PRO/5 are conducted in a similar manner to file operations. Visual PRO/5 attempts to contact the other application by opening a channel with a special mode. After that, the transfer of information between applications is conducted via regular reads and writes to the channel. For example:

```
0010 OPEN (1,MODE="DDECLIENT")"Progman:Progman"
0020 READ RECORD(1,KEY="Progman")TEXT$
0030 PRINT "Current Program Manager Groups:",$0D0A$,TEXT$
```

This example is very simple as the application, item, and topic components for Program Manager are all the same. Line 10 initiates the DDE conversation by opening a channel with the DDE mode. Line 20 reads the information desired - a listing of groups defined in Program Manager. Note that in order to query the Program Manager's groups, you had to know Program Manager's executable name, shared topic, and shared items. A second example:

```
0010 OPEN (1,MODE="DDECLIENT")"Progman:Progman"
0020 LET CMD$="[CreateGroup(DDE Test)]"
0030 WRITE RECORD(1,KEY="DDE_EXECUTE")CMD$
```

This example makes use of the fact that Program Manager responds to the command "[CreateGroup(groupname)]" in order to create a group. It also demonstrates the ease in which Visual PRO/5 can be used to send commands to and manipulate other programs.

Network DDE

Network DDE is somewhat more complex and requires additional information. Network DDE involves accessing a pre-established network DDE Share that is available through the DDE handler on the remote machine. The example in the manual demonstrates Network DDE by means of a conversation between a local machine and a shared resource from the remote machine's Clipbook application.

Servername: This is only used if the DDE conversation is supposed to take place with another machine on the network. In this case, the SERVERNAME corresponds to the machine on the network that you wish to

talk to. You may use an asterisk as a wildcard, which will then establish a connection with the first machine on the network that supports the specified SHARENAME, outlined below. The SERVERNAME is optional, and your local machine will be selected if it is left out (negating the need for network DDE).

Sharename: This is the object that is designated as sharable on the remote machine. Before the Network DDE connection is attempted, the remote machine has to have an application running that has defined a shared object. The example in the manual refers to a shared Clipbook item as an illustration. In order for the example to work, you must verify that the remote machine is running the Clipbook application, the clipbook application has a page to share, and that you have instructed Clipbook to share that page via Network DDE with a given SHARENAME.

Once the Network DDE connection has been initiated, the remote application creates a table of topics that are available. The Visual PRO/5 client can then read the topic list, and carry on a normal DDE conversation by specifying one of the topics gleaned from the table.

The following two programs show how to use DDE to send and receive information from the Windows Program Manager.

```
0010 REM This program reads the groups defined in Program Manager
0020 PRINT 'CS'
0030 OPEN (1,MODE="DDECLIENT")"Progman:Progman"
0040 READ RECORD(1,KEY="Progman")TEXT$
0050 PRINT "Current Program Manager Groups:",$0D0A$
0060 PRINT TEXT$,
```

```
0010 REM This program creates and deletes a Program Manager Group
0020 OPEN (1,MODE="DDECLIENT")"Progman:Progman"
0030 LET CMD$="[CreateGroup(DDE Test)]"
0040 WRITE RECORD(1,KEY="DDE_EXECUTE")CMD$
0050 INPUT "Hit <CR> to delete the DDE Test group",*
0060 LET CMD$="[DeleteGroup(DDE Test)]"
0070 WRITE RECORD(1,KEY="DDE_EXECUTE")CMD$
```