Application Note 166

Using DDE in LabVIEW (Windows Only)

Introduction

Dynamic Data Exchange (DDE) is a communications protocol with which you can communicate with multiple applications on the same computer.

In TCP/IP communications, applications open a line of communication and then transfer raw data. With DDE, applications send messages to each other to exchange information. For example, one simple message is to send a command to another application. Other messages deal with transferring data and referencing the data by name.

All applications communicating through DDE must be running, and all must send Windows their callback function address before DDE communication can begin. Windows calls the function when you send a DDE message to the application.

A DDE client initiates a conversation with another application, a DDE server, by sending a connect message. After establishing a connection, the client can send commands to the server and change or request the value of data the server manages.

Use the DDE VIs to communicate between LabVIEW and other applications. Use these VIs to configure LabVIEW as a client or a server. You can communicate with other applications, such as Microsoft Excel, that accept DDE connections.

You must use a network dynamic data exchange (NetDDE) server to establish a conversation with an application across the network. NetDDE is an internal Windows server available on all Windows platforms.

Using LabVIEW as a DDE Client

You can create VIs that act as clients to request or send data to other applications, and you can create VIs that act as servers to provide data referenced by name that other applications can access.

If you set up LabVIEW as a DDE client, you can perform the following tasks:

- Request items from the application by using the request data command.
- Request the application to send LabVIEW a value when it changes by using the advise data command.
- Send data to the application by using the poke item command.
- Send a string command to the application telling it to execute by using the execute command. The execute command is not available when you create a DDE server on the block diagram. Refer to the Using LabVIEW VIs as DDE Servers section in this application note for more information about using DDE servers.

A client can request data from a server by using a request data command or an advise data command. The client uses a request data command to ask for the current value of the data. If a client wants to monitor a value over a period of time, the client must use an advise data command. By using this command, the client establishes a link between the client and server through which the server notifies the client when the data changes. The client can stop monitoring the value of the data by telling the server to stop the advise link.

When the DDE communication for a conversation completes, the client sends a message to the server to close the conversation.

Service, Topics, and Data Items for DDE Communication

With TCP/IP, you identify the process you want to talk to by its computer address and a port number. With DDE, you identify the application you want to talk to by referencing the name of a service and a topic. Each server picks its own service and topic names. A server generally uses its application name for the service, but not necessarily. The server offers several topics it allows other applications to access. With Microsoft Excel, for example, the topic may be the name of a spreadsheet.

To communicate with a server, first find the names of the service and topic you want to access. Consult the documentation for the application for this information. Open a conversation using these two names to link to the server.

Unless you send a command to the server, you usually work with data items that the server shares. You can treat these data items as a list of variables that you can manipulate. You can change variables by name, supplying a new value for the variable, or you can request the values of variables by name.

Examples of Client Communication with Microsoft Excel

Each application that uses DDE has a different set of services, topics, and data items that it can share. For example, two different spreadsheet programs can take very different approaches to how they specify spreadsheet cells. To find out if an application is DDE compatible, consult the documentation for that application.

With Microsoft Excel, the service name is `Excel`. For the topic, you use the name of an open document or the word `System`.

If you use the word `System`, you can request information about the status of Microsoft Excel or send general commands to Microsoft Excel that are not directed to a specific spreadsheet. For instance, for the topic `System`, Microsoft Excel communicates about items such as `Status`, which has a value of `Busy` if Microsoft Excel is busy or `Ready` if Microsoft Excel is ready to execute commands. `Topics` is another useful data item you can use when the topic is `System`. The topic returns a list of topics Microsoft Excel shares, including all open spreadsheet documents and the `System` topic.

Figure 1 shows how you can request a list of Microsoft Excel topics from LabVIEW. The DDE Request VI returns a string that contains the names of the open spreadsheets and the word `System`.

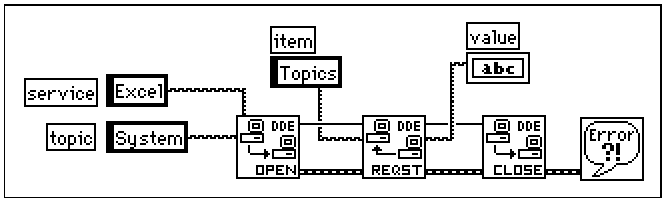


Figure 1. Requesting a List of Microsoft Excel Topics

Another way you can use the `System` topic with Microsoft Excel is to instruct Microsoft Excel to open a specific document. Use the DDE Execute VI to send a Microsoft Excel macro that instructs Microsoft Excel to open the document, as shown in Figure 2.

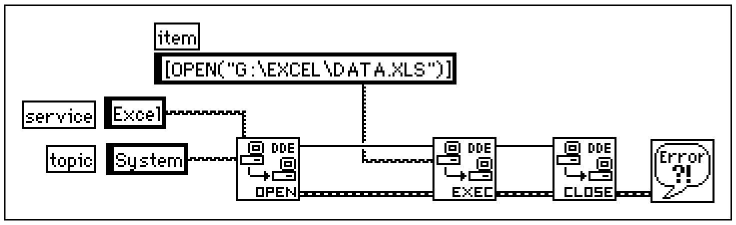


Figure 2. Using a Microsoft Excel Macro to Open a Microsoft Excel Document

Using LabVIEW VIs as DDE Servers

You can create VIs that act as servers for data items. The VI indicates that it is willing to provide information regarding a specific service and topic pair. LabVIEW can use any name for the service and topic name. You can specify the service name as the name of the application (`LabVIEW`) and the topic name as the name of the server VI or a general classification for the data it provides, such as `Lab Data`.

The Server VI registers data items for a service that it will talk about. LabVIEW remembers the data names and their values and handles communication with other applications regarding the data. When the server VI changes the value of data registered for DDE communication, LabVIEW notifies any client applications that have requested notification about that data. In the same way, if another application sends a Poke message to change the value of a data item, LabVIEW changes this value.

You cannot use the DDE Execute VI with a VI acting as a server. LabVIEW does not support this functionality. If you want to send a command to a VI, you must send the command using data items.

LabVIEW is not inherently a DDE server to which you can send commands or request status information. However, you can use VIs to create a DDE server.

Figure 3 shows how to create a DDE Server VI that provides data to other client applications. In this case, the data is a random number. You can easily replace the random number with real-world data from data acquisition devices connected to the computer.

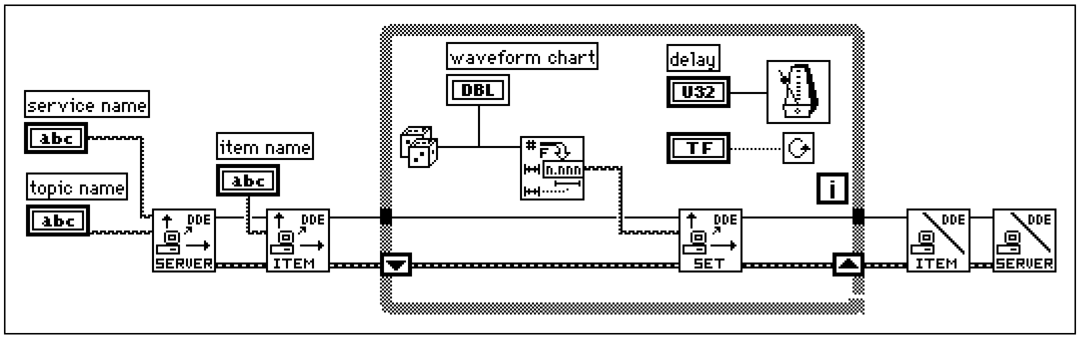
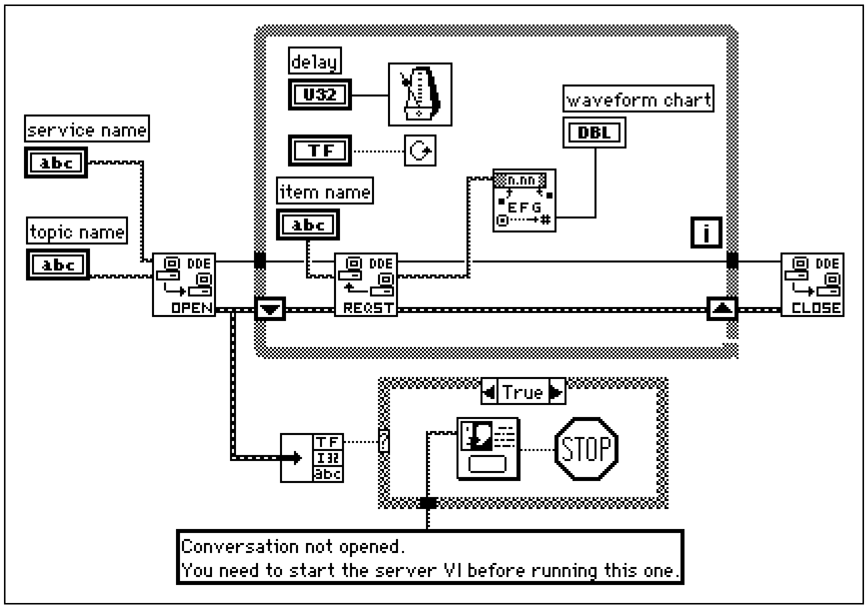


Figure 3. DDE Server VI Example

The VI in Figure 3 registers a server with LabVIEW and registers a data item that it can provide to clients. In the loop, the VI periodically sets the value of the data item. As mentioned earlier, LabVIEW notifies other applications that data is available. When the loop is complete, the VI unregisters the data item and the server. The clients for this VI can be any application that understands DDE, including other VIs.

Figure 4 illustrates a client to the VI shown in Figure 3. It is important that the service, topic, and data item names are the same as the ones the server uses.



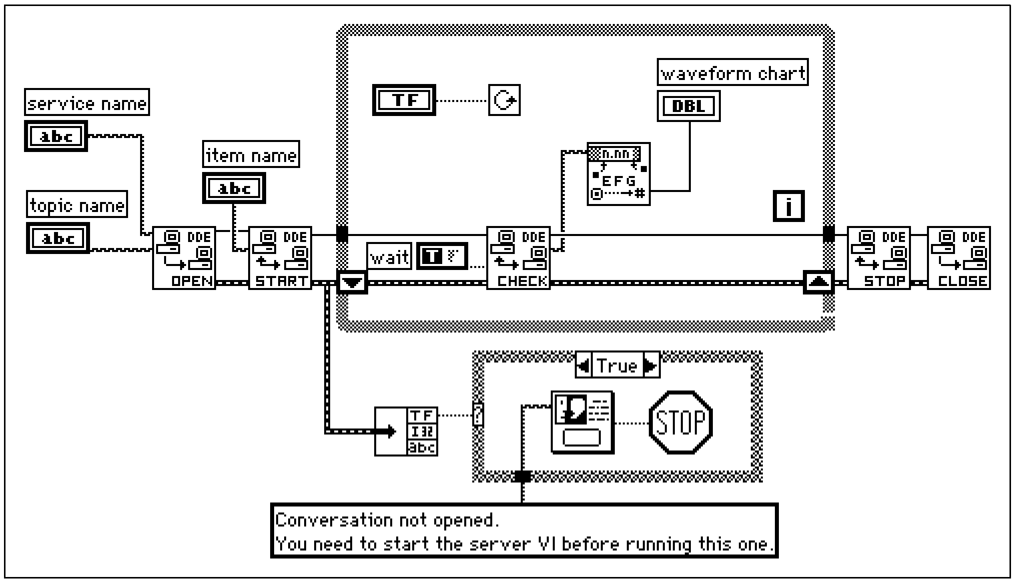
Conversation not opened.
You need to start the server VI before running this one.

Figure 4. Client VI to DDE Server VI

Requesting Data Versus Advising Data

Figure 4 uses the DDE Request VI in a loop to retrieve data. With DDE Request VI, LabVIEW retrieves the data immediately, regardless of whether you have seen the data before. If the server and the client do not loop at exactly the same rate, you can duplicate or miss data.

One way to avoid duplicating data is to use the DDE Advise VIs to request notification of changes in the value of a data item. Figure 5 shows how you can implement this technique.



Conversation not opened.
You need to start the server VI before running this one.

Figure 5. Using DDE Advise VIs

In Figure 5, LabVIEW opens a conversation and uses the DDE Advise Start VI to request notification of changes in the value of a data item. Every time the loop executes, LabVIEW calls the DDE Advise Check VI, which waits for a data item to change its value. When the loop is finished, LabVIEW ends the advise loop by calling the DDE Advise Stop VI and closing the conversation.

Synchronization of Data

The client-server examples in the preceding section work well for monitoring data. However, in these examples there is no assurance that the client receives all the data that the server sends. Even with the DDE advise loop, if the client does not check for a data change frequently enough, the client can miss a data value that the server provides.

In some applications, missed data is not a problem. For example, if you are monitoring a data acquisition system, missed data may not cause problems when you are observing general trends. In other applications, you may want to ensure that no data is missed.

One major difference between TCP and DDE is that TCP queues data so that you do not miss it and you get it in the correct order. DDE does not provide this service.

In DDE, you can set up a separate data item that the client uses to acknowledge that it has received the latest data. You then update the acquired data item to contain a new point only when the client acknowledges receipt of the previous data.

For example, you can modify the server example shown in Figure 5 to set a state item (`data available` in the following figure) to a specific value after it has updated a call to the server VI. The server monitors the state item until the client acknowledges receipt of data. Figure 6 shows this modification.

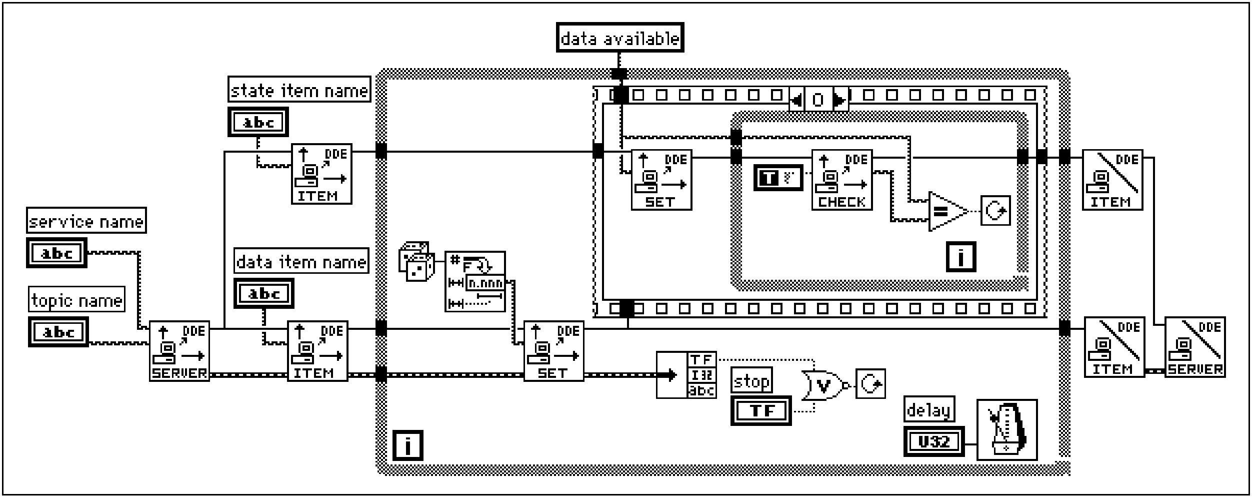


Figure 6. Monitoring State Item

A client for this server, shown in Figure 7, monitors the state item until it changes to `data available`. When the state item value is `data available`, the client reads the data from the acquired data item the server provides and updates the state item to `data read` value.

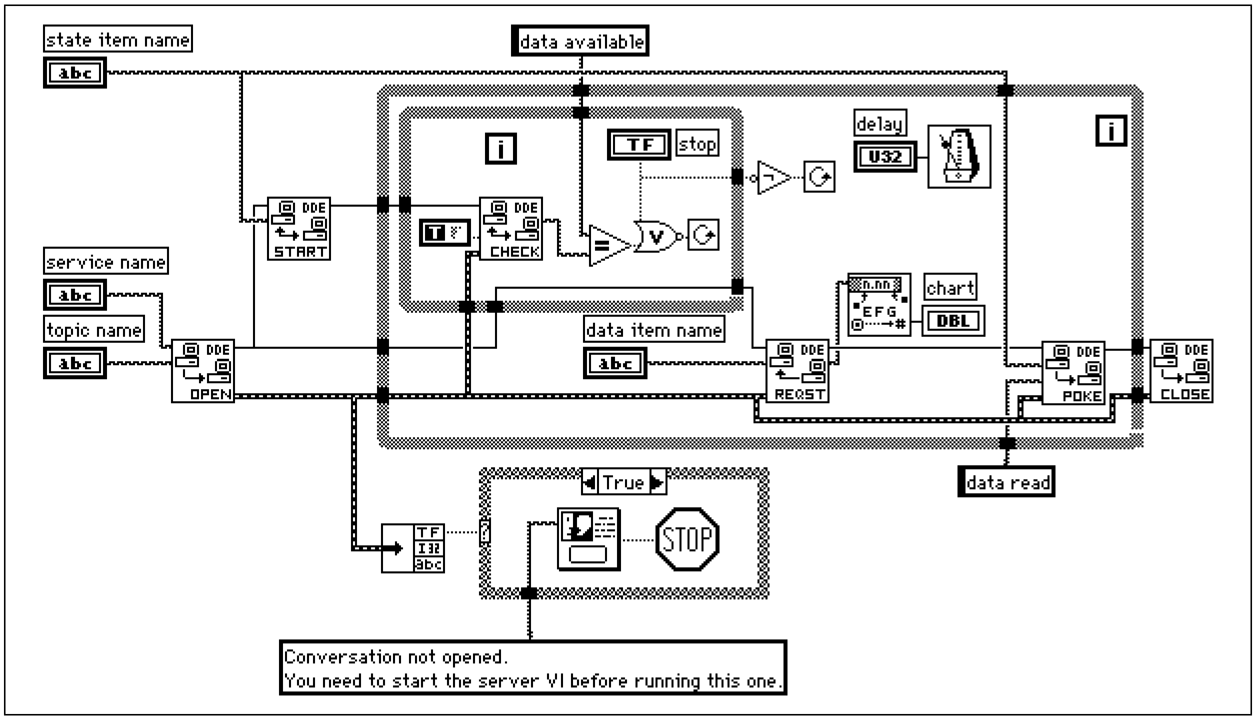


Figure 7. Client for State Item VI in Figure 6

Although this technique makes it possible to synchronize data transfer between a server and a single client, it has some shortcomings. You can have only one client. Two or more clients can conflict with one another. For example, one client might receive the data and acknowledge it before the other client notices that new data is available.You can build more complicated DDE block diagrams to deal with this problem, but they quickly become awkward.

If your application needs reliable synchronization of data transfer, you may want to use TCP/IP instead because it provides queueing, acknowledgment of data transfer, and support for multiple connections at the driver level.

Using NetDDE

In addition to using DDE to communicate with applications on the same computer, you can use it to communicate over the network with applications on different computers. A network dynamic data exchange (NetDDE) server is required to establish a DDE conversation with an application across the network. NetDDE is an internal Windows server available on all Windows platforms. You must set up a Windows 9x or Windows 2000/NT NetDDE server before you set up NetDDE clients.

Setting up a NetDDE Server on Windows 9x

If this is the first time you are running NetDDE for Windows 9x, you have to first run `\Windows\NetDDE.exe` to configure NetDDE on your computer. You can add this to the StartUp directory so that it always starts. If you do not add `NetDDE.exe` to your StartUp directory, then every time you want to use NetDDE, you need to select **Start»Run** and type `\\NetDDE`.

Complete the following steps to set up a NetDDE server on Windows 9x:

1. Select **Start»Run**.
2. Type `Regedit.exe`.
3. In the tree display, open `My Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\NetDDE\DDE Shares`.
4. Select **Edit»New»Key** to create a new DDE Share.
5. Name the new DDE share `lvdemo`.
6. With the **lvdemo** share open, select **Edit»New** to add all the values in Table 1 to the new DDE share.
7. Edit each value by right-clicking the value and selecting **Modify** from the shortcut menu.
8. Close REGEDIT.
9. Restart the computer for the changes to take effect.
10. Select **Start»Run**.
11. Type `NetDDE`. Your computer is now configured for DDE communication.

Table 1. NetDDE Value Types

Value Type	Name	Value
Binary	Additional item count	`00 00 00 00`
String	Application	`service_name`
String	Item	`service_name`
String	Password1	`service_name`
String	Password2	`service_name`
Binary	Permissions1	`1f 00 00 00`
Binary	Permissions2	`00 00 00 00`
String	Topic	`topic_name`

Note These keys are the same as in the **CHAT$** share, but REGEDIT does not allow you to cut, copy, or paste keys or values. When you create the share, a default value named (`Default`) and a value of (`value not set`) appears. Do not modify these values.

Setting up a NetDDE Server on Windows 2000/NT

Complete the following steps to set up a NetDDE server on Windows 2000/NT:

1. Select **Start»Run**.
2. Type `\winnt\system32\DDEShare.exe`.
3. Select **Shares»DDE Shares»Add a Share** to register the service and topic names on the server you are sharing.

Connecting from a DDE Client Computer

On the client computer that is running the application that initiates a DDE conversation, no configuration changes are necessary.

Complete the following steps to set up LabVIEW as a DDE client:

1. Place the DDE Open Conversation VI on the block diagram.
2. Right-click the **service** input and select **Create»Constant** from the shortcut menu.
3. Enter `\\machine_name\ndde$` into the constant where `machine_name` specifies the name of the DDE server computer.
4. Right-click the **topic** input and select **Create»Constant** from the shortcut menu.
5. Enter `lvdemo` into the constant where `lvdemo` matches the service and topic names in Table 1.
6. Run the VI.

342074A-01 Jul00