Using Dynamic Data Exchange (DDE) as an Integration Tool

Troy Kostek
Purdue University

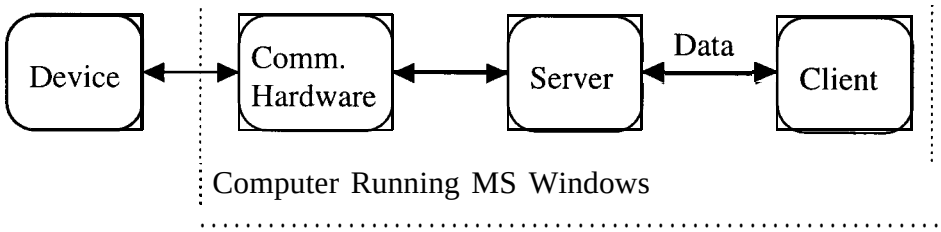
Dynamic Data Exchange (DDE) is a protocol describing how Microsoft Windows applications exchange data on a real-time basis. In today's Windows-based multitasking environment, two Windows programs simultaneously running on the same computer (or running on two different computers connected via a LAN) can communicate with one another using DDE. With the large variety of Windows-based data acquisition and control software available, DDE plays a vital role in the integration of today's automated manufacturing systems. As educators of students that will be entering the complex world of automated manufacturing, it is important to introduce the concepts of DDE and to teach how DDE can be used as an integration tool. This paper describes the fundamentals of DDE and provides two case studies of how DDE is used as an integration tool in laboratory-based manufacturing courses at Purdue University.

Clients and Servers

In any one particular DDE conversation, there is one server (also called the source) and one client (also referred to as the destination). Data can be transferred from the server to the client or from the client to the server. For instance, a Microsoft Visual Basic program can function as a server or a client, depending on the application. Some programs, such as Rockwell's WINtelligent Linx (discussed in more detail below) only function as a DDE server. A given application program can be involved in multiple DDE conversations at the same time and the application can function as the client in some conversations and as a server in other conversations [1]. In general, the client initiates the transfer of data by opening the DDE channel. After the communications channel is open, the client can then issue DDE commands which enable the client to read data from or write data to the server. The server, on the other hand, generally provides a special service for the client. For example, the server may graph data which is sent from the client or the server may support a communications link to hardware located on the factory floor.

Figure 1 depicts the client/server relationship commonly found in today's integrated manufacturing environments. As shown in Figure 1, the server program "knows" how to communicate to the device. The device could be a single device connected to the computer using an RS-232 point-to-point link or the device could be one of many devices connected to the computer via an RS-485 multidrop or LAN. Typical devices range from programmable logic controllers (PLCs) to intelligent sensors such as machine vision systems. The server communicates through the communications (comm.) hardware which could consist of a special purpose adapter card or could be as simple as one of the computer's built-in RS-232 serial ports. The server understands the communications protocol necessary to talk to the device. This protocol can be relatively simple or could require detailed low level communications programming involving a complex data packet with CRC (cyclic redundancy check) error checking. CIM integrators sometimes have to write their own

Figure 1- The Role of The DDE Server



servers; however, man y integrators prefer to work at a higher level and buy existing servers in order to save time and make the integration process simpler. The server program (a commercially available product typically written by a program development team) takes care of the technical details of how to communicate to the device, while the client program (typically written by a systems integrator or a manufacturing engineer) can simply ask for data from the device or send data to the device in a standard way using DDE. The server makes the technical details of how to communicate to the device transparent to the integrator.

An example of a commercially available DDE server is Rockwell software's (formally ICOM Inc.) WINtelligent Linx. The Linx package is a DDE server which facilitates computer-to-PLC communications to Allen-Bradley, GE/Fanuc, Modicon, Square D, and Reliance PLCs[4]. In an Allen-Bradley environment, the communications hardware shown in Figure 1 could consist of a 1784-KT adapter card which allows the computer to function as a node on Allen-Bradley's Data Highway Plus LAN. The client program can then communicate to any PLC on the LAN.

Application, Topic, and Item Names

Each DDE conversation can be described by 3 pieces of information: application name, topic name, and item name. The application name is the name of the server. The application name is the name of an executable (.EXE) program. For example, if Microsoft Excel is the DDE server then the application name is EXCEL. If WINtelligent Linx is the server, then the application name is ICOMWDRV (short for ICOM's Window DRiVer). The topic name is often a file name. For example, suppose a Visual Basic program (client) sends data to Microsoft Excel (server). The application name would be EXCEL and the topic name would be the name of the Excel file (filename. XLS). When using WINtelligent Linx as a server and Visual Basic as a client, the topic is not a filename. When Linx is used as the server, the topic contains all the information necessary to communicate with the intended PLC. The topic is basically a description of the path to the PLC (often over the Data Highway Plus LAN). One generally defines a topic for each PLC on the network. DDE topics are defined in Linx and then the topic name is either manually entered into the client program or copied into the client via the Windows clipboard facility. The item is the name of the data object that is transferred during the DDE conversation. Table 1 lists several examples of item names.

Poke and Request Transactions

After the DDE channel is open, the client can then issue transactions to the server. With respect to computer-to-PLC communications, the most common type of transactions include:

a) Poke - Data is transferred to the server from the client.

b) Request - Data is transferred to the client from the server.

Table 1- Examples of DDE Item Names

If the DDE server is:	The item is the name of a:	Example:	Comment:
Excel	cell or range of cells	R2Cl	name of cell A2
Visual Basic	label, picture box, or text box	txtData	name of text box
WINtelligent Linx	data table address or range of addresses	N7:0	name of PLC -5 integer file

Both the poke and the request transactions are one-time transfers in the sense that the DDE channel is opened, the data transfer occurs once, and the channel is then closed. Because of the one-time nature of the transfer, the poke and request transactions are often referred to as "cold" DDE links.

It is possible to setup a "hot" DDE link between the server and the client. In a "hot" link, the channel is opened and remains opened. Data is transferred to the client whenever new data becomes available (i.e., the data changes). When a hot link is established, the server (Linx for example) will poll the device at a user specified interval. For example, Linx can be configured to poll a specific PLC every 2 seconds. If Linx determines that the PLC data has changed since the last polling cycle, it will transfer the new data to the client over the DDE hot link. The term solicited is also used to describe the polling activity. In the solicited mode, Linx generates a great deal of network traffic which could drastically reduce the LAN's throughput if several hot links are established by multiple computers.

'The unsolicited mode is a more efficient way to communicate with field devices. In the unsolicited mode, the PLC sends data to the computer (Linx) when it needs to. When Linx receives the PLC data it simply passes the data onto the client. The only minor disadvantage of this mode is that it requires additional PLC logic. For example, in an Allen-Bradley PLC-5 application, one message command must be added for each unsolicited message.

DDE Formats

DDE communications can take place using different protocols or formats. From the end users perspective, the actual format employed does not change how the user performs the DDE transfers. The chosen format does affect how well (fast) the transfer takes place. When a DDE channel is first opened, the client and the server automatically determine among themselves which format to use via a hierarchical procedure. Table 2 illustrates a few commonly used formats.

Table 2- DDE Formats

Format:	Description:
CF_Text	Developed by Microsoft. Is the most fundamental format which all Windows compatible programs should recognize.
XLTable	Developed by Microsoft specifically for MS Excel
FastDDE	Developed by Wonderware Inc. for use with their software products
AdvanceDDE	Developed by Rockwell Software for use with their software products

Visual Basic version 3.0 supports only CF_TEXT. Rockwell software, however, offers a VBX (a Visual Basic extension) which supports AdvanceDDE which is much faster and more efficient than CF_Text.

Teaching DDE To Manufacturing Students

The remainder of this paper will consist of two case studies showing how DDE is used in an educational environment. The first example shows how DDE plays a vital role in PC-to-PLC shop floor communications. A commercially available server is employed (WINtelligent Linx) and the client application is a Visual Basic program written by the students. The second case study involves a data acquisition problem in which the students write their own DDE server and use a commercially available supervisory control and data acquisition (SCADA) package as the client.

Case Study 1- The CIM Application

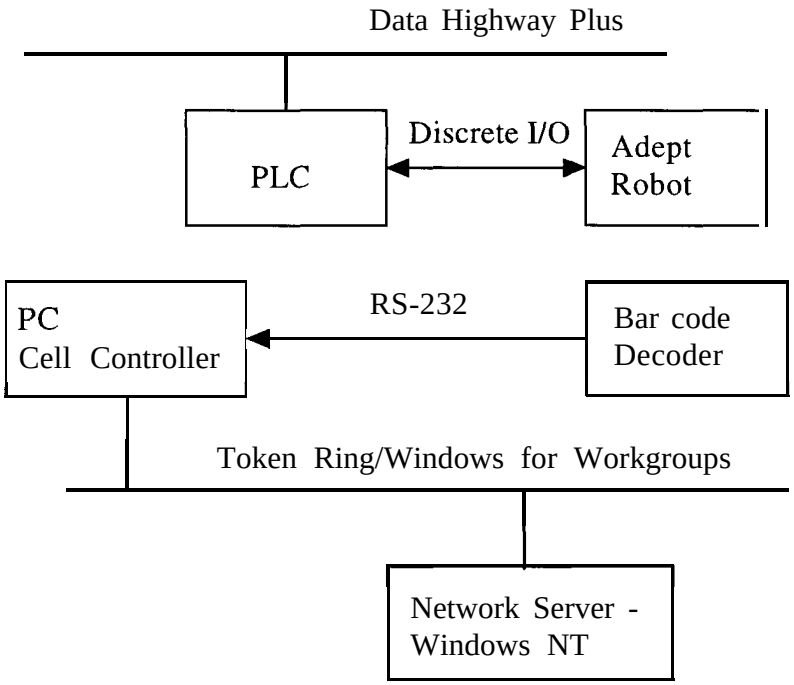
During the Fall 1994 semester, the instructor introduced a special laboratory assignment into a senior-level Computer-Integrated Manufacturing Technology (CIMT) course entitled "Manufacturing Networks." The fundamental goal of this course is to teach how data communications and LANs are used to integrate manufacturing operations at the shop floor level. The laboratory assignment, called the CIM application, was assigned at the end of the semester and was designed to pull together the material presented from day one of the semester into one relatively complex project. The CIM application forced students to link together multiple shop floor devices and acquire data from a Microsoft Access database in order to solve a practical manufacturing problem.

The CIM application consists of a hard disk rework cell. It is assumed that a hard disk (consisting of 4 disks separated by 3 spacers) was previously manufactured and tested. The test results are stored in a rework database. The students task is to implement the rework cell which basically rebuilds the disk pack. Each student work station consists of the following equipment:

- A section of hi-directional conveyor (from Shuttleworth Inc.) with various photoelectric sensors and product stops.
- An Allen-Bradley bar code decoder/scanner.
- An Allen-Bradley PLC-5/30 with a wide variety of I/O and operator interfaces. The PLCS are networked using Allen-Bradley's Data Highway Plus LAN.
- An Adept One SCARA electric robot with associated Adept CC controller. A vacuum type end effecter was attached to the arm.
- A 486-based IBM value point personal computer. Multiple computers are networked using IBM Token-Ring and Microsoft Windows for Workgroups. The computer was loaded with Microsoft Access, Microsoft Visual Basic, and Rockwell Software's WINtelligent Linx.
- A wiring patch panel. This panel allows safe and convenient wiring for application dependent PLC and robot wiring requirements.

The basic setup of the rework cell is illustrated in Figure 2.

Figure **2-** The Hard Disk Rework Cell (One of Three)



The basic manufacturing scenario flows as follows:

1) A failed hard disk, attached to a pallet, enters the rework cell via the conveyor system.

2) Based on a photoelectric sensor, the PLC recognizes the presence of the pallet. Using DDE, the PLC informs the PC cell controller that a new pallet is present.

3) The Visual Basic (VB) program running on the cell controller commands the bar code system to scan the bar code label. The VB program then captures the bar code data.

4) The VB program attaches to a Microsoft Access database which is located on the network file server. Using VB's data control, the VB program queries the database in order to find out which disk on the disk pack has failed.

5) Using another DDE link, the VB program then transfers the failed disk number "down" to the PLC.

6) The PLC then informs the Adept robot which disk is bad via discrete input/discrete output (Di/Do) lines.

7) Using a vacuum gripper, the robot rebuilds the disk pack.

8) - When finished, the robot informs the PLC that the disk pack was successfully rebuilt. The pallet is allowed to move "upstream."

9) Using another DDE link, the PLC informs the VB program that the disk pack was successfully rebuilt.

10) VB updates the Access database indicating that the disk pack was indeed rebuilt successfully and enters the date in which the disk pack was rebuilt.

Table 3 summarizes the DDE transactions required to complete the CIM application.

Table 3- Summary of DDE Transactions for The CIM Application

DDE Conversation	Direction	Type of Condensation	Description
1	PLC-to-PC	Unsolicited	New pallet present
2	PC-to-PLC	Cold link via Poke	Download failed disk number
3	PLC-to-PC	Unsolicited	Disk pack successfully rebuilt

Case Study 2- Data Acquisition

During the 1995 Spring semester, the instructor introduced a laboratory assignment into a senior-level CIMT-course entitled "Manufacturing Applications of Sensor Technology." The laboratory assignment used a Honeywell edge/width sensor to measure the width of a blown film bubble which was extruded from a blown film plastics machine. The edge/width sensor is essentially a low cost vision system consisting of a linear array of 256 pixels. Data is acquired from the sensor using RS-232 serial communications. The blown film plastics machine (donated by Mobil Chemical Corporation) is controlled by an Allen-Bradley PLC-5/30 PLC.

In this application the students VB program functioned as a DDE server. The VB program configured the sensor to measure the width of the plastic bubble as well as the left and right edge locations of the bubble. The sensor information was used to determine if the bubble's width was within established limits (often at startup the bubble would burst) and if the bubble was "creeping" left or right. After the sensor was configured, data was periodically (every 1000 ms) acquired from the sensor over the RS -232 serial communications channel.

The students VB program established a hot DDE link with a commercially available SCADA package. WINtelligent View (from Rockwell Software inc.) is a Windows-based SCADA package which supports both DDE and PLC communications (View communicates to PLCS using WINtelligent Linx). The DDE server was necessary because View did not support direct RS-232 serial communications. With the hot link established and View functioning as a DDE client, data was transferred from the VB program to the View application whenever the data changed.

The View application displayed the current width, left edge, and right edge values associated with the extruded bubble. Using View's built in "action/reaction" capabilities, alarm conditions were detected and specific bits in the PLC's data table were set if any of the bubble data was out of range. At this point in the

assignment's development cycle, the focus has been on data acquisition and alarming. In the near future it is desirable to close the loop and try to have the PLC compensate and automatically correct the alarm Cowd.iti.ems,-

Summary

This paper has explained the fundamental concepts and terminology associated with DDE and has described two case studies showing how DDE can be used as an integration tool. In today's integrated manufacturing environment, it is important to be able to share and exchange data from one program application to another, where each application plays a special role in the integration process. Dynamic Data Exchange is the "glue" which enables this level of integration. As a minimum, technical graduates entering the work force should have a basic understanding of DDE concepts and capabilities.

References

1. Microsoft Corporation. (199 1). Dynamic data exchange (DDE). Unpublished manuscript.
2. Feldman, P. (1993). Using visual basic 3. Indianapolis, IN: Que publishing.
3. Rockwell Software Incorporated. (1993). Using visual basic and winlinx. Unpublished manuscript.
4. Rockwell Software Incorporated. (1995). WINtelligent Iinx. User's guide.
5. Gurewich, N. (1993). Teach yourself visual basic 3 in 21 days. Indianapolis, IN: Sams Publishing.

TROY E. KOSTEK

Troy E. Kostek is an Associate Professor of Mechanical Engineering Technology at Purdue University. Since 1986, Troy has been teaching several senior-level courses in the Computer-Integrated Manufacturing Technology program at Purdue. His main area of interest includes programmable controllers, sensors, and machine vision.