Art_7.03

PERFORMANCE ASSESSMENT OF DDE VERSUS ACTIVEX IN MANUFACTURING ENVIRONMENTS

André Quintã*[1], José Santos[1], Carlos Cardeira[2]
[1]University of Aveiro, Department of Mechanical Engineering - Aveiro, Portugal
[2]Lisbon Technical University, Instituto Superior Técnico, GCAR-DEM - Lisboa, Portugal
**Email:* aquinta@mec.ua.pt

ABSTRACT

This paper assesses data communications at shop floor level. DDE protocol and ActiveX controls are common practice solutions for manufacturing system integration. In this paper we test several transmission times recorded using both DDE protocol and ActiveX controls to access shop floor resources from a personal computer. We used experimental scenarios to test both solutions. We present some advantages and constraints of both solutions, as well as some recommendations.

1. INTRODUCTION

In industrial environment system integrators have to find out solutions to exchange information between human and material resources (information flow integration). The industrial equipment and its external communications interfaces diversity (I/O, RS232, Ethernet, TCP/IP, Profibus, …) as well as the factory software diversity raises problems to system integrators and have to be considered in the choice of the best integration solution [Pimentel 1990] and [Halsall 1996].

This paper in particular, focus the integration of manufacturing resources with control and acquisition systems based on computers. These systems are widely spreaded in industry and they are a main key for factories competitiveness [Gershwin 1993], [Vernadat 1996], [Santos 2001] and [Quintã 2004].

The industrial equipment manufacturers, namely manufacturers of Programmable Logic Controllers (PLC's), make available some communication interfaces and programs, drivers, libraries that can be used by systems integrators to develop software programs to access and to control manufacturing resources and, by this form, allow a first level of industrial process integration.

In this paper, we deal with two common practices (DDE and ActiveX), usually available by manufacturers to access and control its PLC's from a PC running the Windows XP operating system.

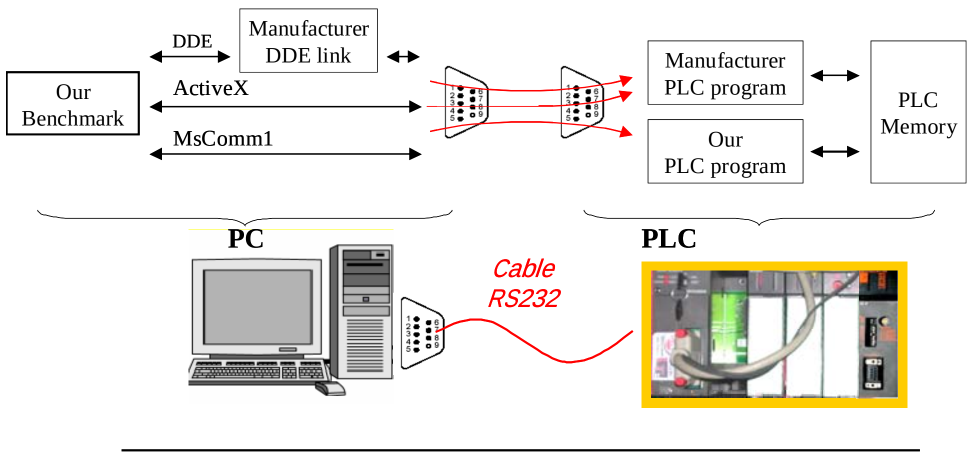
Having in mind the manufacturing system integration, several transmission times have been recorded using both DDE protocol and ActiveX controls to access shop floor resources from a personal Computer. Both DDE and ActiveX are briefly presented in section 2. The experimental scenario used to assess these two solutions and a proprietary approach, are described in section 3. The transmission times are also recorded, presented and assessed. Finally in the section 4 will be presented some advantages and constraints of both solutions, some recommendation are also presented.

1 - One of the interfaces supplied by PLC's manufacturers consists on a Windows program, that we will call *ManufacturerDDE_Link*. This Windows program works as a DDE (Dynamic Data Exchange) server allowing that other Windows programs, namely integration programs, became able to communicate with it through a DDE link. The *ManufacturerDDE_Link* is not only a DDE server, but it uses also the PC serial port RS232 or the PC Ethernet board to communicate with the PLC. In fact, the *ManufacturerDDE_Link* codifies and resends to the PLC the information generated by the others Windows programs. The data sent for the PLC must have a specific format in order to be well received and processed by the PLC program (*ManufacturerPLC_Program*), figure 1.

2- Another interface supplied by manufacturers consists of a library of ActiveX controls and it could be used during the development of new integration programs (compile time).

This library also codifies user's data in the format expected by the *ManufacturerPLC_Program.*

3 – There is also a third choice to access and to control the PLC that consists in the development of two proprietary programs: a proprietary PLC program that can read and write in the RS232 communications board of the PLC and a user program that can read and write in PC Rs232 port. In this case both programs will be, of course, compatible.



Rs232 Settings: 9600,8,1,n

Figure 1: The assessment scenarios (DDE, Active X, Mscomm1)

In all three alternatives it is necessary that the PC/Windows platform be physically connected to the PLC through a RS232 or TCP/IP/Ethernet connection.

So, to allow communication between two equipments, it is necessary that both "use the same language", by this reason the PLC came by default with programs that are able to receive and answer to messages created by the *ManufacturerDDE_Link* or by ActiveX controls.

Objective

This paper goal is to reply to the following question "to access and to control the PLC's from a computer what hypothesis should I choose":

1-to use the DDE/Windows protocol and DDE server supplied by PLC's manufacturers,

2-to use the ActiveX library also supplied by PLC's manufacturers, 3-to develop a proprietary solution, developing a program to the PC and another to the PLC that can directly communicate one with other through a serial or TCP/IP/Ethernet connection?

Evaluation Criteria

This choice must have in account the data exchange speed, its reliability, the time and cost of software development, the easiness of software reuse, and its portability.

2. COMMUNICATIONS PROTOCOLS IN WINDOWS ENVIRONMENT

DDE stands for Dynamic Data Exchange, DDE is a technology for data exchanging between Windows programs, and it was implemented by Microsoft with the first Windows versions (Windows 2.0).

DDE is based on a server/client model. To enable a DDE connection it is necessary a server program and, at least, one client program. Several DDE clients may exist simultaneously. The DDE server program makes available to others programs a group of items through witch is possible to exchange data.

The client takes the initiative to establish the DDE connection, and the server accepts or not the request, after establishing the connection both server and client are able to take the initiative of write and read data (strings) on DDE items.

The first Human Machine Interface (HMI) software used to supervise and control PLC's based process's, was developed with the DDE technology. The PLC manufacturers developed their own DDE servers which can also access to the PLC through a traditional serial or TCP/IP connection, making available an easy link to the PLC from PC/Windows programs.

The DDE communications, between the client and the server, have to pass through a message routing system (DDEML.dll) witch demands a lot of system resources and slows the process especially with large amounts of data.

Since 1990, most DDE programs have been replaced with the emergence of OLE and COM technologies, however several automation programs still uses DDE communication, especially those that only require the exchange of simple data type (strings).

The need for a better integration among Windows programs induced the development of OLE (Object Linking and Embedding) technology in 1990 with Windows 3.1. The next version OLE 2.0 was release in 1993, OLE 2.0 is based on the new software architecture Component Object Model (COM). COM provides an interaction between different pieces of software in a consisted, objected-oriented way. COM objects can be written in all sorts of languages and re-use code without requiring re-compilation with the implementation in DLL's [Chappel 1997].

The industrial automation uses the name OPC (OLE for Process Control) for OLE standards specifications. OPC resulted from the collaboration of a number of leading worldwide automation suppliers [OPC].

In 1996 Microsoft renamed OLE to ActiveX, that refers to COM based technologies and leave the OLE designation to compound documents.

ActiveX controls have been extremely used in the last years especially on the internet technologies. An ActiveX control is a control that uses ActiveX technologies. The PLC manufacturers also adopted the ActiveX technology to the development of HMI software. An

ActiveX Control, can be placed inside another program that understands how to host ActiveX controls, like Visual Basic or Visual C++ and use its functionality integrated with the final user program.

With the release of Visual Studio.Net, Microsoft introduces the .Net Framework, this technology is rapidly replacing the software based on the COM technology, having as main advantages the liberty for choosing programming languages, easiness to make databases available on internet with Web Services and .NET is backwards compatible with COM, DDE, OLE, ActiveX, etc… .

In the last couple of years the .NET technology has been implemented in nearly all computing domains and also on manufacturing industries the .NET based programs are emerging, mainly in business level such as ERP, MRP and e-business. In resources level some manufacturers of industrial resources, namely PLC's are developing new tools for remote control and monitoring based on .NET objects [Htservices].

3. IMPLEMENTATION AND EVALUATION SCENARIO

To assess the performance of these three hypotheses, one program was developed, in Visual Basic. This program is able to communicate with the PLC in all the three indicated ways and it is used to assess the communication performance between the PC and the PLC (figure 2). From this point ahead we will call to this program *T_Benchmark*.

The *T_Benchmark* resides in the PC and of course, consumes itself some processing time, but this time is similar in the three hypotheses that we will go to analyze, therefore this processing time will not influence the assessment. The *T_Benchmark* uses one of the three VBasic objects to access to the PLC. One of the objects establishes a DDE linking with the *ManufacturerDDE_Link* and this in turn uses the PC RS232 port to access to the PLC. The ActiveX and the "MsComm" VBasic objects directly access the control chip of the PC RS232 port.

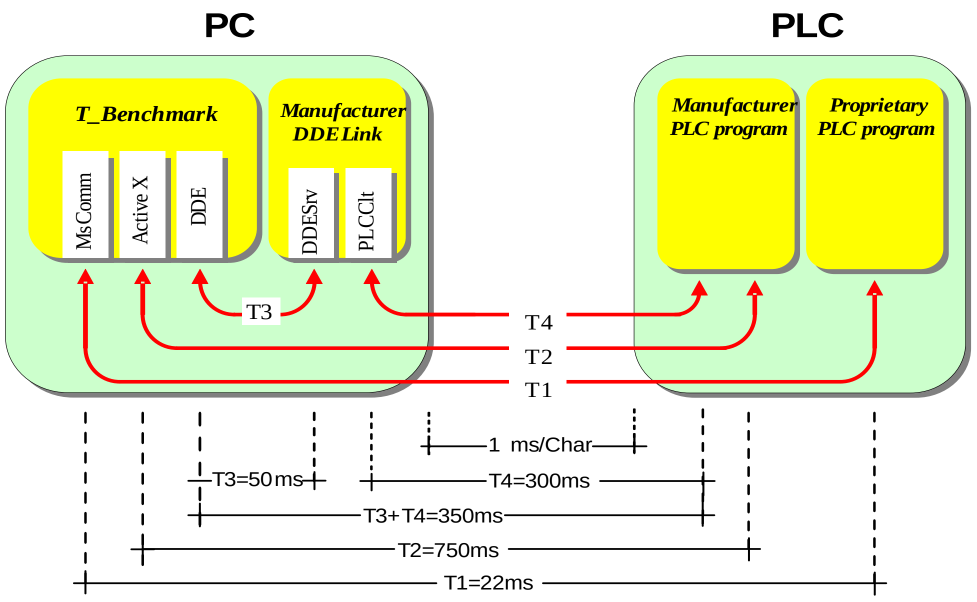


Figure 2: Communications hypotheses and transmissions times between PC and PLC

Independently of the chosen hypothesis, a physical connection must exist between the PC and the PLC to support the information flow between the two, namely a RS232 or a TCP/IP/Ethernet connection. In the local Ethernet network the transmission time needed to exchange data between two equipments is not always the same (not deterministic), because when two equipments try to access in simultaneous to the same communication environment (Multiple Access) they could origin a collision in the transmission medium "Collision Detect". When a collision is detected (Collision Detect), several retransmission attempts are done, with random waiting times between each attempt.

So, the physical connection chosen to connect the PC to the PLC, was the RS232 serial communication (9600, 8, 1, n) because the transmission time is well known. However, a special care is to be considered in RS232 communications, processing time in the PLC, as well as in the PC, must always be inferior to transmission times so that they do not interpose with the evaluation performance.

To measure the communication delay, between the PLC and the PC, the *T_Benchmark* sends a character to the PLC and the PLC returns the same character to it. When this character arrives and only then, the *T_Benchmark* sends a new character to the PLC. The *T_Benchmark* repeats this sequence some thousands times. The transmission times are stored in a database for posterior analyses. In fact, there were repeated and recorded several thousands sequences.

3.1. Communication between the *T_Benchmark* and the PLC, using MsComm object (T1)

The time T1 (figure 2) depends of the: *T_Benchmark* processing time, Rs232 baud rate and communication protocol complexity. Section 3.1.1 will present some transmission and processing times. Section 3.1.2 will present the proprietary protocol (packet format and message sequences) used to communicate with the Proprietary PLC program, developed by us.

3.1.1. Assessment of *T_Benchmark* processing time, using the MsComm object

The *T_Benchmark* can directly access the RS232 computer port using the MsComm object. In this case the *T_Benchmark* uses two RS232 ports of the PC and a RS232 cable connect them externally. By this way the PLC processing time is not considered because the serial connection is made between the serial port one (COM1) and the serial port two (COM2) of the same computer where the *T_Benchmark* program is executed. Because the serial communication transmission time is already known, it’s possible to deduce the processing time of the *T_Benchmark*.

The transmission time of one character, through a RS232 connection with 9600 bits/s, codified in 8 bits, with one stop bit for character, without parity bit, is approximately 1ms. Therefore, the time for sending and receiving one character must be at least 2 ms, which is, in fact, the measured time by the benchmark. This means that the *T_Benchmark* processing time is very small and it is not affect the evaluation of the communication time.

3.1.2. Measurement of Transmission Times (T1): between the *T_Benchmark* and the Proprietary PLC program, using the MsComm object

When the PLC receives one character on its RS232 port, stores the value in its internal memory and resend the same character to the PC. Because the PLC processing time is inferior to 0.1 ms, it does not affect the data transmission performance.

To read or write into the PLC is not enough to send a single character with a value, it is necessary to define a packet with several fields/characters to transport some extra information, as like as the PLC location where the value will be stored or read.

The developed and implemented proprietary protocol foresees two different operations: **write** (W) a value in the PLC and **read** (R) a value from the PLC.

To the **write** operation, the PC sends a W-packet defining the value to be written and the memory addresses where it should be stored.

For a **read** operation two packets must be exchanged between the PC and the PLC. A read request RR-packet should be sent by the PC and then the PLC respond with an R-packet, figure 3.

The RR-packet indicates the desired PLC memory address, and the response packet (R-packet) transports the PLC memory address and the respective read value (figure 3). The type of each packet is identified by its first character.

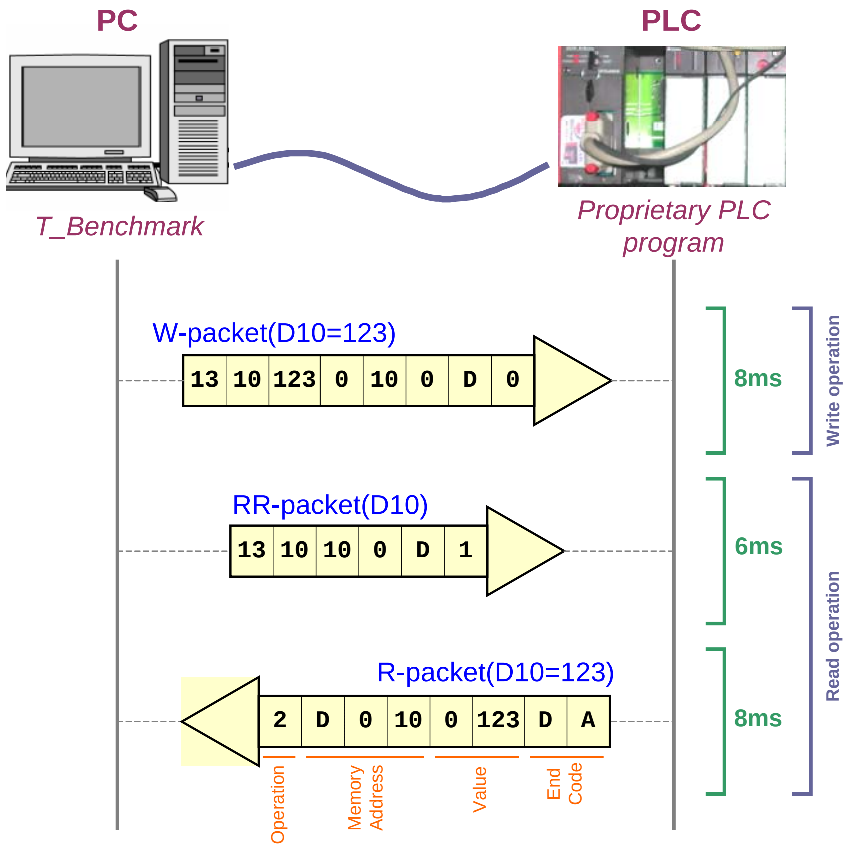


Figure 3: Proprietary protocol, communication between the PC and the PLC

The PLC memory address are identified by a letter (X, Y, M or D depending of memory type), followed by a decimal value (from 0 to 2047). Therefore, the proprietary protocol uses three bytes to identify the memory address, one to the memory type and two to the memory number. The PLC memory address type can be boolean (0..1), integer (0..32767) or word (0..65535), so two bytes were used in R-packet and RR-packet to transport the memory number. Also a terminal code (Hex 0D0A) was used to identify the packet end. It takes two bytes.

So the task of the *T_Benchmark* to write one character to the PLC and read it again, with this protocol involves three steps, write with a packet of 8 characters, read with a packet of 6 characters and the acknowledge with a 8 characters packet. As referred on section 3.1.1 the average transmission time of one character through serial communication is 1ms, so the total transmission time for sending and receiving one character with the proprietary protocol must be 22ms, corresponding to 22 bytes. In fact the measure time varies from 22 to 23ms, *T1* in figure 2.

3.2. Communication between the *T_Benchmark* and the PLC, using ActiveX control (T2)

The ActiveX control also implements a specific communication protocol; by each character sent by the *T_Benchmark* (using the ActiveX) several messages (characters) are exchanged with the PLC through the RS232 connection. For this reason the effective transference rate, on the user perspective is low, it varies from 749ms to 751 ms for each user character sent and received, *T2* in figure 2. However the use of ActiveX control is more reliable than the *ManufacturerDDE_Link* communications used in the next case.

3.3. Communication between the *T_Benchmark* and the PLC using the *ManufacturerDDE_Link* (T3+T4)

As presented, the *T_Benchmark* access to the PLC through another Windows program provided by the PLC manufacturer, so-called "*ManufacturerDDE_Link*". Therefore the total expended time by the *T_Benchmark* to exchange values with the PLC is equal to time T3 + time T4, figure 2.

3.3.1. Measurement of Transmission Times (T3): between *T_Benchmark* and *ManufacturerDDE_Link*, using a *DDE_Link*

These measurements assess the time expended for DDE communications, inside the Windows environment, without considering serial communications or PLC processing time influence. Using the same *T_Benchmark* and the same characters sequence of the previous example, it was possible to conclude that the time range for sending and receive one character through a DDE link varies from 50.1ms to 51.6 ms, *T3* in figure 2.

3.3.2. Measurement of Transmission Times (T4): between the *ManufacturerDDE_Link* and the PLC, using a RS232 connection

The *ManufacturerDDE_Link* exchange data with PLC through an RS232 connection and uses a specific protocol, with specific data packets (headers and bodies), to encapsulate each character sent by the *T_Benchmark*.

The expended time to exchange data with the PLC depends a lot of the communications protocol used by the *ManufacturerDDE_Link* program and the *ManufacturerPLC_Program*. Due to this protocol, by each character sent for the *T_Benchmark*, several characters are sent to the PLC by the *ManufacturerDDE_Link* and therefore the effective transfer rate, in the user perspective, is low.

In average, the time for sending and receiving one character, in the user perspective, varies between 351ms to 353 ms, *T3+T4* in figure 2.

From this time only 50ms are spent in DDE communications, the remaining *T4*=300ms are spent in the information exchange between the Windows program *ManufacturerDDE_Link* and the *ManufacturerPLC_Program*. To send and receive one character through the serial cable takes only 2 ms (9600 baud, 8 bits, 1 stop bit) but the measured time is approximately 350 ms. Therefore, we can conclude that by each character sent and received by the user many characters are exchanged between the PC and the PLC.

4. CONCLUSIONS

As referred before, three hypotheses were considered to assess the performance of PLC access and control: proprietary programs, DDE Link and ActiveX controls.

This choice of the best solution must have in account the speed of data exchange, its reliability, the time and cost of development of informatics applications based in these hypotheses, easiness of software reuse, and its portability.

1- The development of integration applications that directly access to the PLC through a RS232 serial communication, interacting with a proprietary PLC program, allows the faster data transference but require more development time, costs and aren't so easily reusable as the other two hypotheses. The proposed protocol presented on section 3.1.2 is not based on confirmed messages, so it is not as reliable and robust as manufacturers protocols used with DDE and ActiveX solutions. By this reason, only when the communication time will be a fundamental factor, this situation should be chosen.

2- The *ManufacturerDDE_Link* application, supplied by the PLC manufacturer, presents neither the faster times neither the slower transmission times. However it is easy to use and reuse in new applications, to communicate with the same or other PLC's from the same manufacturer, through a RS232 or TCP/IP/Ethernet connection. But the data exchange through the *ManufacturerDDE_Link* has some times great random delays, which can come go up to several seconds or even to total communication failure, especially when it is tried to send information with high cadence. Therefore, DDE can be a fast implementation solution to develop simple integrations tools, that don't involve large amount of data exchange.

3- The ActiveX control presents the slowest data communication rate of the three considered hypotheses, but on the other hand it allows a reliable and secure communication. The developments of new integration programs are also easy and fast. The ActiveX controls can be easily reused to access and to control several PLC's from the same manufacturer, also being able to use either RS232 or TCP/IP/Ethernet connection to communicate with the PLC. The implementation of ActiveX reveals to be the best solution especially to more complex applications.

When developing new integration tools to automation and manufacturing systems, unless when the transmission time is a fundamental factor, ActiveX controls are the most complete solution.

Taking into account all factors (speed, reliability and reusability) it will be up to the user to choose the best solution for each application. By presenting a performance evaluation of the several solutions, we hope this work will be helpful for determining the right choice for each application.

Despite this work does not tests OPC servers, this type of solution is very common on the automation industry. It's foreseen as future work to compare the access times presented in this work with an OPC based solution.

REFERENCES

[Chappel 1997] Chappel, D., Linthicum S., "ActiveX Demystified. It's invasive. It's ubiquitous. But what, exactly, is ActiveX?", Byte.com, 1997, http://www.byte.com/art/9709/sec5/art1.htm.

[Gershwin 1993] Gershwin, S. B., "Manufacturing Systems Engineering", Prentice-Hall, Englewood Cliffs, New Jersey, 1993).

[Halsall 1996] Halsall, F., "Data Communications, Computer Networks and Open Systems", 4th ed., Addison-Wesley, 1996.

[Htservices] High Tech Services Production & Laboratory Automation Systems Integration Consulting & Training, http://www.htservices.com/.

[OPC] OPC Foundation, http://www.opcfoundation.org/.

[Pimentel 1990] Pimentel, J. R., "Communication Networks for Manufacturing", Prentice Hall, Englewood Cliffs, New Jersey, 1990.

[Quintã 2004] Quintã, A. F., Santos, J. P. O., "WEB based Integration Infrastructure for remote rent a factory", 6º Conference on Automatic Control, Faro, Portugal, 2004.

[Santos 2001] Santos, J.P.O., "Uma arquitectura para a integração da produção baseada em modelos executáveis", PhD Thesis, University of Aveiro, 2001

[Vernadat 1996] Vernadat, F.B., "Enterprise Modeling and Integration: Principles and Applications", Chapman & Hall, London, 1996.