1. DISTRIBUTED CONTROL ARCHITECTURE

1.1. Structure of the industrial control system

Currently, there are three major trends in contemporary industrial control systems:

- distributed and decentralized structures of automation,
- increased integration of communication through all levels of the control systems supported by the application of wired and wireless networks,
- growing demand for the application of IT standards.

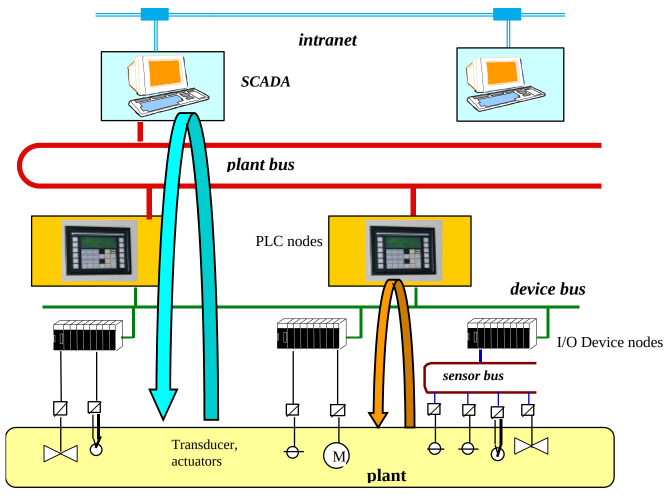

Fig.1.1 Multilevel structure of an industrial control system

Most industrial control systems adopt a multilevel, vertical control hierarchy. Logically, such a system (see Fig. 1.1) is structured in three levels: the direct (device) control level, the supervisory level and the management level.

The basic task of the direct (device) control level is to maintain the process states at the prescribed set values. The device controller level provides an interface to the hardware, either separate modules or microprocessors incorporated in the equipment to be controlled. Here, mainly PID digital control algorithms are implemented – in some cases these are more advances control methods such as multivariable control or adaptive functions. A number of embedded control nodes and Programmable Logical Controllers (PLC) are used as the front-ends to take the control tasks. High speed networks and fieldbuses are implemented at the direct control level to exchange in real time the information between front-ends and the device controllers and, vertically, with the supervisory control level. This architecture has the advantage of locating the hard real-time activities as near as possible to the equipment.

The supervisory level comprises workstations and industrial PCs providing high-level control support, database support, graphic man-machine interface, network management and general computing resources.

Classically, the supervisory level calculates set points for controllers according to the defined criteria. For this purpose more complex mathematical models of the process are employed at this level to find the optimal steady-state, by solving optimisation and identification tasks. Due to the rapid development of computer technology, there is growing scope for more advanced close-loop algorithms (predictive control, repetitive control) located at this level. However, increasing computational efficiency of PLCs at the device level supported by high performance networks transferring data and control signals vertically gives more flexibility to the designer. The control loops can be handled by local, device–level controllers, and also by the supervisory controllers (Fig.1.1). For example, a predictive control algorithm can be handled by a supervisory workstation as well as by a local PLC. In some cases similar control algorithms must be located in both levels if redundancy of the control system is required. It should be noted that upper level loops usually offer shorter computational time due to the higher efficiency of the workstations.

The role of the communication networks at each level is to ensure data transmission and coordinate manipulation among spatially distributed control nodes. The evolution of industrial communication has moved to Industrial Ethernet networks [1]. Since Ethernet is a shared network, the packets containing the digitized measurements need to share the network bandwidth with external traffic; thus, the available channel capacity is limited and dynamically changing. Therefore, the control over the upper level loop usually offers longer data transfer time due to high vertical network traffic and longer delays.

1.2. Integration problem

The ability to easily integrate information from control systems and „plant floor" measurements with supervisory and optimisation levels is a critical issue. It is very difficult to share data between various devices and software manufactured by different vendors without a common industrial communication protocol and integration standard to facilitate interoperability.

The key is an open and effective communication and integration architecture concentrating on data access, not on the type of data. Ethernet protocol comes as a solution to the data transmission problems mentioned above, while the OPC (OLE for Process Control) data exchange standard was developed as a solution which fulfills the requirements of open data integration architecture.

OPC is a widely accepted open industrial standard that enables the exchange of data without any proprietary restrictions between multi-vendor devices and control software (horizontal integration) as well as between different software applications (vertical integration, Fig.1.2). Currently, most plant-level control systems can be configured to be the OPC servers for supervisory control levels of the factory.

OPC is based on the Microsoft DCOM specification. Although OPC actually consists of many different data exchange specifications, its most commonly used form is Data Access (OPC DA), which supports both client-server and publisher-subscriber data exchange models. OPC DA deals only with current process data - not historical data or alarms.

An OPC client can be connected to several OPC servers provided by different vendors. An important feature of OPC is that the client is able to connect to the OPC server which is located in another computer in the network. This is possible because of DCOM (Distributed COM) the COM standard extension which enables a client running on one computer to create instances and invoke methods of servers on another computer within the network.

The OPC standard defines numerous ways to communicate between the server and its clients to satisfy different OPC applications. The client can specify that some operations should be performed on "cache" or "device". If "device" is chosen, operation directly on the physical device will be requested. If "cache" is selected data will be read from the server's internal memory where the server keeps a copy of the device data received during the last OPC refresh cycle.

Both reading and writing can be done synchronously or asynchronously:

- OPC synchronous functions run to completion before returning. This means that the OPC client program execution is stopped till operation is completed.
- OPC asynchronous functions use a callback mechanism to inform the requested OPC operation is over. The client program execution is not stopped while waiting for OPC operation to be completed.

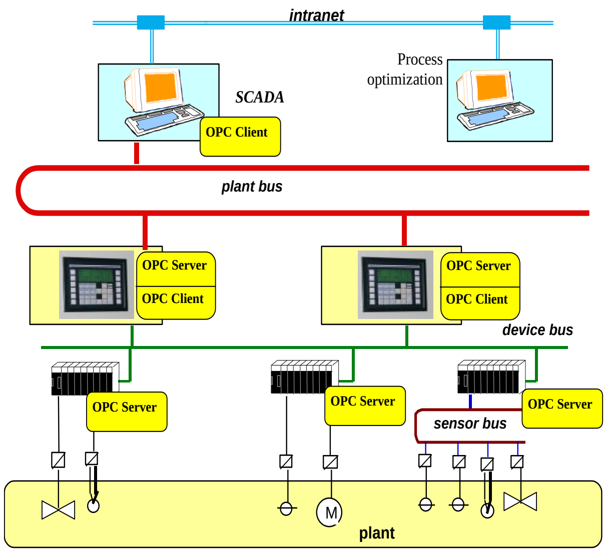

Fig. 1.2 Vertical integration implementing the OPC data exchange model

The OPC standard was not primarily intended for feedback control or communication with high-bandwidth, hard real-time requirements. However, many I/O devices require real-time capabilities that are specific to the hardware and essential to proper operation of the high-level SCADA application. Therefore, one of the most important issues for communication between devices is proper configuration and synchronization.

Author: W. Grega