A Real Time, Object Oriented Fieldbus Management System

Mr. Ole Cramer Nielsen
Managing Director – PROCES-DATA
Supervisor – International P-NET User Organisation
Navervej 8
8600 Silkeborg
Denmark
pd@post4.tele.dk

Abstract

Communication between networked PC's and fieldbuses requires the availability of a real time operating system to access and control external object variables. Since the majority of commercial and factory PC's in use today are based on one of the flavours of Windows (95,98,NT), it is essential that such real time features provide a seamless interface between a fieldbus and other communicating media, and the Windows operating system. Furthermore, Windows offers powerful object oriented features, which can be used for internal communication between PC applications, be these spreadsheets, databases or visualisation programs such as SCADA, and is known as OLE2 Automation (Object Linking & Embedding). If the real time enhancements can also utilise OLE technology, a completely transparent path between fieldbus objects and PC applications can be achieved.

1. Introduction

The paper describes the structure of VIGO [1], which is a real time fieldbus management system that operates within the Windows environment. One the one hand, VIGO provides all the drivers to inter-communicate with fieldbuses such as P-NET and other media such as Ethernet, Modbus, GSM, Modem and the Internet. On the other hand, VIGO is an OLE server, and provides all the data conversion required to enable two way communication between PC applications and local or remotely located fieldbuses. VIGO is in fact a suite of programs [3], which provides all the tools and utilities required to configure, monitor, simulate and maintain a fieldbus system, together with facilities to download programs and back up and restore system data.

Visual VIGO [2] is also described, as an example of an object oriented PC application that utilises the facilities offered by VIGO. Visual VIGO is a visualisation package (SCADA), enabling fieldbus objects to be graphically displayed and manipulated from a PC screen. An example will be demonstrated of communication and control between such a display and a remotely located P-NET fieldbus via GSM and the Internet.

2. General Principles

VIGO is a Fieldbus Management System, installed on PC's, servers and workstations, which use the Microsoft Windows 95, 98 or NT operating systems. VIGO is used in conjunction with process automation systems, where individual control units are distributed within a plant, and where one or more Fieldbuses are used for the data inter-communication. Whereas Windows is an operating system, which executes programs, controls the keyboard and screen, manages the hard disc and contains tools for configuration and program execution, VIGO is also an "operating system", used to handle the various real-time tasks specific to a Fieldbus system. This involves activities such as appropriately loading communication and hardware drivers, converting between various data formats and providing a dynamic interface with Windows. Within the context of this paper, real time means "virtually immediate" from the point of view of a human operator, constrained only by the inherent message rate of the fieldbus or network protocol and the responsiveness of the PC.

2.1 Windows Application Interface

Windows enables data to be transferred internally between applications, e.g. between a spreadsheet and a word processor or a database. If PC's or workstations are linked together within a Local Area Network (LAN), transfer of data can also take place between applications via such a network. However, real-time data, such as that found within a fieldbus network is not directly

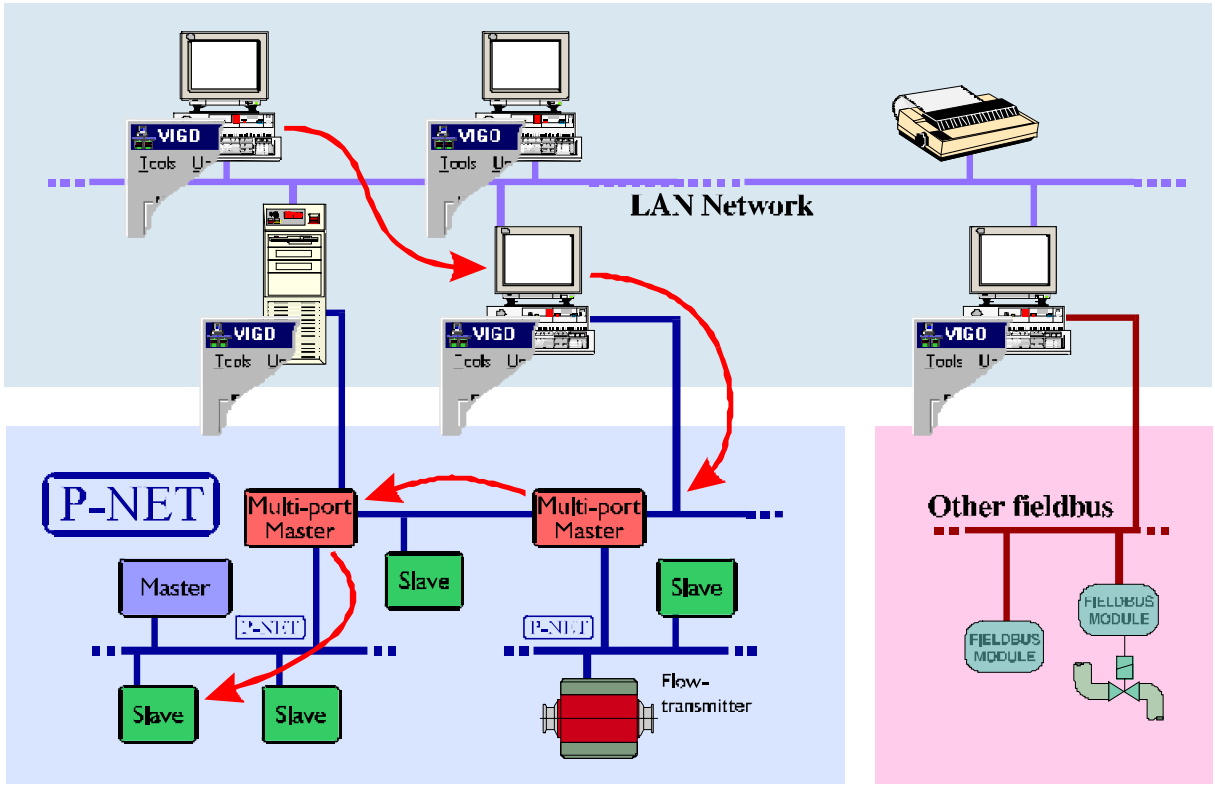

Figure 1. Internetworking and fieldbus interfacing principles of VIGO

supported. VIGO enhances Windows, to enable real-time data from a distributed environment such as that found in an industrial processing plant, to be utilised within standard or purpose designed applications on a workstation. Thus an EXCEL spreadsheet can include real-time data from the plant, for logging or trending purposes. Alternatively, ACCESS can be used to log and produce reports on live events occurring within the external system. Although the use of standard applications can be extremely useful, to quickly display or capture fieldbus data (variables), a more specific application may often be required. In this case, standard object oriented software development languages can be used, such as Visual Basic, Delphi, C++ etc., to produce the required result using real-time data from the field.

2.2 OLE2 Server

All this is possible because one part of VIGO acts as an OLE2 Automation Server. This means that any standard or proprietary application which supports OLE (Object Linking & Embedding), such as SCADA systems, Asset Management and Maintenance Scheduling packages, can transparently send data to, or request data from a fieldbus, such as P-NET. Applications don't need to know how the real-time data is obtained. All that is necessary is for the application to refer to a chosen symbolic identifier such as "Furnace_1_Temperature", and VIGO will do the rest. Furthermore, identifiers need not just be single entities such as a temperature reading already processed into engineering units, but can be complete objects containing the properties of a measurement value, set point, scaling value, location, type, maintenance information, error information etc.

2.3 Manager Information Base

To do this, VIGO needs to hold information about all the variables that are to be used within a project and used in a workstation application. All this information is held within an internal database and is controlled by VIGO's Manager Information Base (MIB). The database associates an application identifier with an address on the fieldbus from where the variable can be obtained (or written to). This may also involve obtaining information on the route to take to obtain the data, it's data type and even the type of network being used. Some of this information has to be provided by the user when a new project is being set up, or a current project is being expanded. This is performed using the MIB Editor.

2.4 MIB Editor

The MIB Editor is the user interface for the database and provides the facilities to define the elements within a project and to illustrate these in a meaningful way. In the same way that Windows provides a Device Manager and a File Manager to assess and modify how a workstation is set up, the Manager Information Base provides details

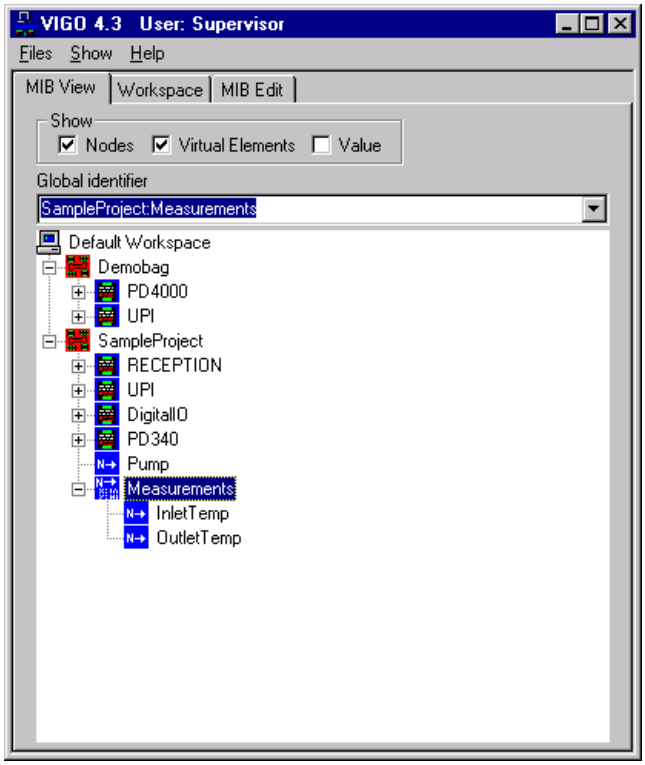

Figure 2. Illustrative structure of a project

of Devices (nodes) within the project. By selecting a device, as one would a folder to see the file contents, the MIB provides information on the available contents (channels and variables) within each device. With P-NET, process objects, such as digital or analogue I/O, are described as a set of related variables contained within a specific Channel type. Since VIGO is an object-oriented system, each device possesses a number of properties, such as device type or node address, which can be amended by the user using the MIB editor. The similarity with Windows continues, in so far as the MIB holds a complete library of standardised devices and channels, which can be included in a project. Therefore, in a similar manner to choosing and installing a new CDROM or sound card device within a PC, the MIB editor is used to select the required devices, which can then be included and configured for use within a particular project.

2.5. VIGO Tools

The MIB Editor, although probably the most important, can be regarded as one of a number of tools

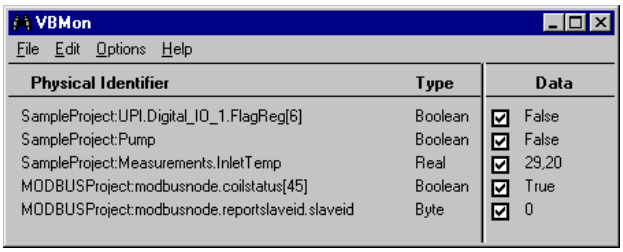

Figure 3. Monitoring selected variables

available, which uses the contents of the Database. Take for example Monitor, whereby using the illustrative structure of the project previously described, can select a variable, the value of which will immediately be displayed on the screen in real-time. These values can also be modified from the screen, assuming they are read/write variables. Any number of variables can be selected for display on a single or multiple screens, which can then be saved for later use. Another associated tool provides an easy way to set or change the node address of any P-NET device. All that is required is to input the unique Serial No. of the device in question. Reference is then made to the MIB database for the required node address, which is then downloaded to that physical node.

Other tools include utilities to easily configure standardised channels such as digital and analogue I/O. Facilities are also available for down loading programs and taking a complete backup of a system. Although a library of standardised devices and channels is included within the Database, there are often times when a specially written program has to be designed for one, or a number of distributed controllers within a project. During the design phase the contents of the MIB data for

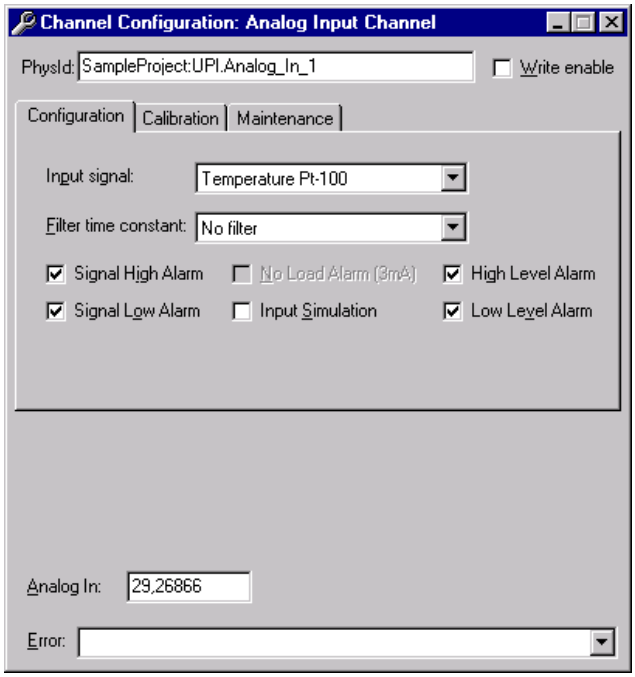

Figure 4. Configuration of specific channel

the project can be directly used during the global variable declaration stage. The SMB file generated during compilation of this program, which contains all the information about defined local controller variables and constants to be used, can also be utilised by the MIB in order that these can be included within the project illustration. This means that these can also be used for

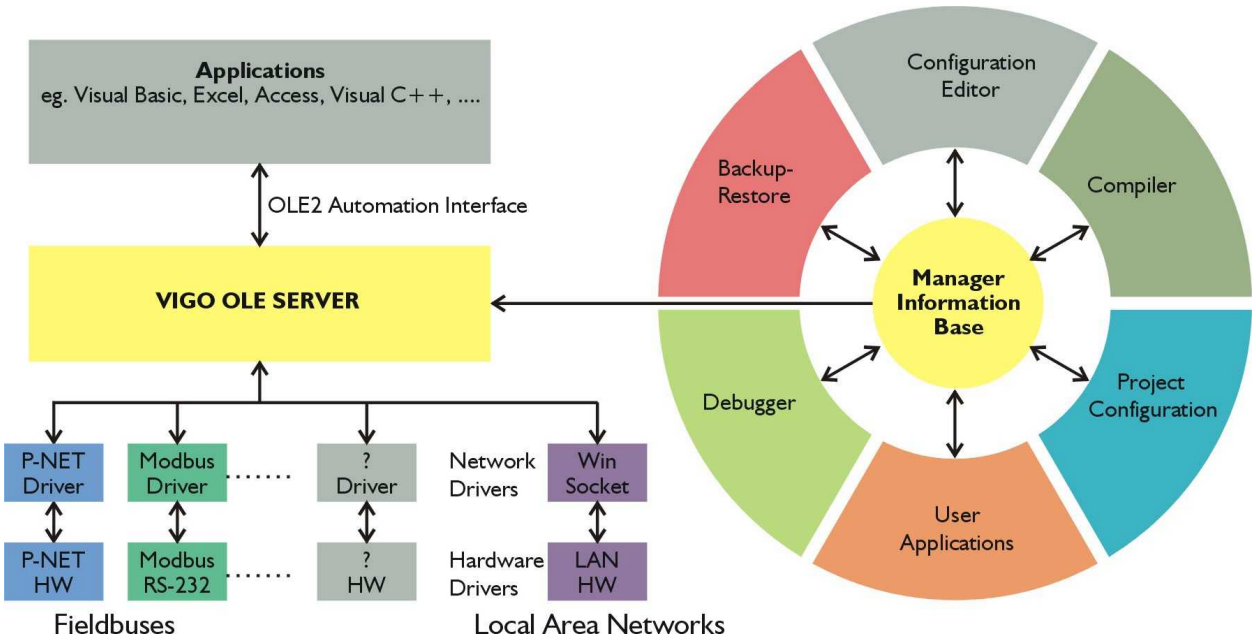

Figure 5. Basic structure of VIGO

modification, monitoring and debugging, using the powerful features of VIGO.

2.6. Plant Simulation

Since a complete description about a particular physical plant is held by, and can be accessed from the MIB, this enables a simulation of the operation of such a project, to be performed within the PC. A simulation program would define the functionality, using the devices available within the actual plant. This facility can be extremely useful when designing or commissioning large plants, or to see the effect of specific customised changes. The advantages in terms of enabling highly efficient training programmes to be devised, are also obvious.

2.7 VIGO Structure

VIGO deals with communication between field devices and local PC applications and/or remote workstation applications via a variety of LAN protocols supported by Windows 95/98/NT. For the real-time fieldbus interfaces, VIGO is structured in an open way, to enable a variety of fieldbus protocol drivers to be incorporated, and used in parallel. By necessity therefore, the real-time aspects of VIGO require that its operation is as fast as possible. This is why it is a true 32 bit protected mode package, fully utilising the fast and safe data exchange features of OLE2 Automation.

3. Visual VIGO

Visual VIGO is used to visualise and operate process plants, factories, building automation etc. on a PC. The PC is connected on-line locally or remotely (via direct MODEM access or Internet connection) through the VIGO fieldbus management system and the P-NET fieldbus. The process plant is visualised by designing one or more *work floors*, where machines and instruments can be placed. The operator can "walk" through the plant by opening "doors" to the different work floors and departments of the plant. The different machines can be opened for studying internal details.

3.1 Object oriented components

Visual VIGO is based on an object oriented principle, where Visual VIGO *components* can contain other components, which again can contain other components etc. Visual VIGO components can illustrate a complete machine, a sub-section of a machine, an actuator, a sensor, a control function, a value, a set point, a record of data etc. The appearance of a Visual VIGO component is based on images. Visual VIGO components can take different states. These states can be illustrated using different images. The images can originate from a picture taken with a camera, generated by any standard drawing program or computed by the Visual VIGO components.

3.2 Development goals

The following has been main goals for Visual VIGO*:*

Ease of use for designers as well as operators.

Visual VIGO components can be customised by system integrators.

Visual VIGO components can be developed by third parties, and added to existing systems without compilation.

Visual VIGO is a true member of VIGO and as such, Visual VIGO components can inherit the functionality of VIGO. The principle of using the right-hand mouse button in Visual VIGO is the same as in VIGO, where a selected channel or variable shows a menu with the relevant tools or properties. The right-hand mouse button menu is also used to open relevant documentation, such as Help files and hardware manuals including those associated with sensors, actuators and related connection diagrams.

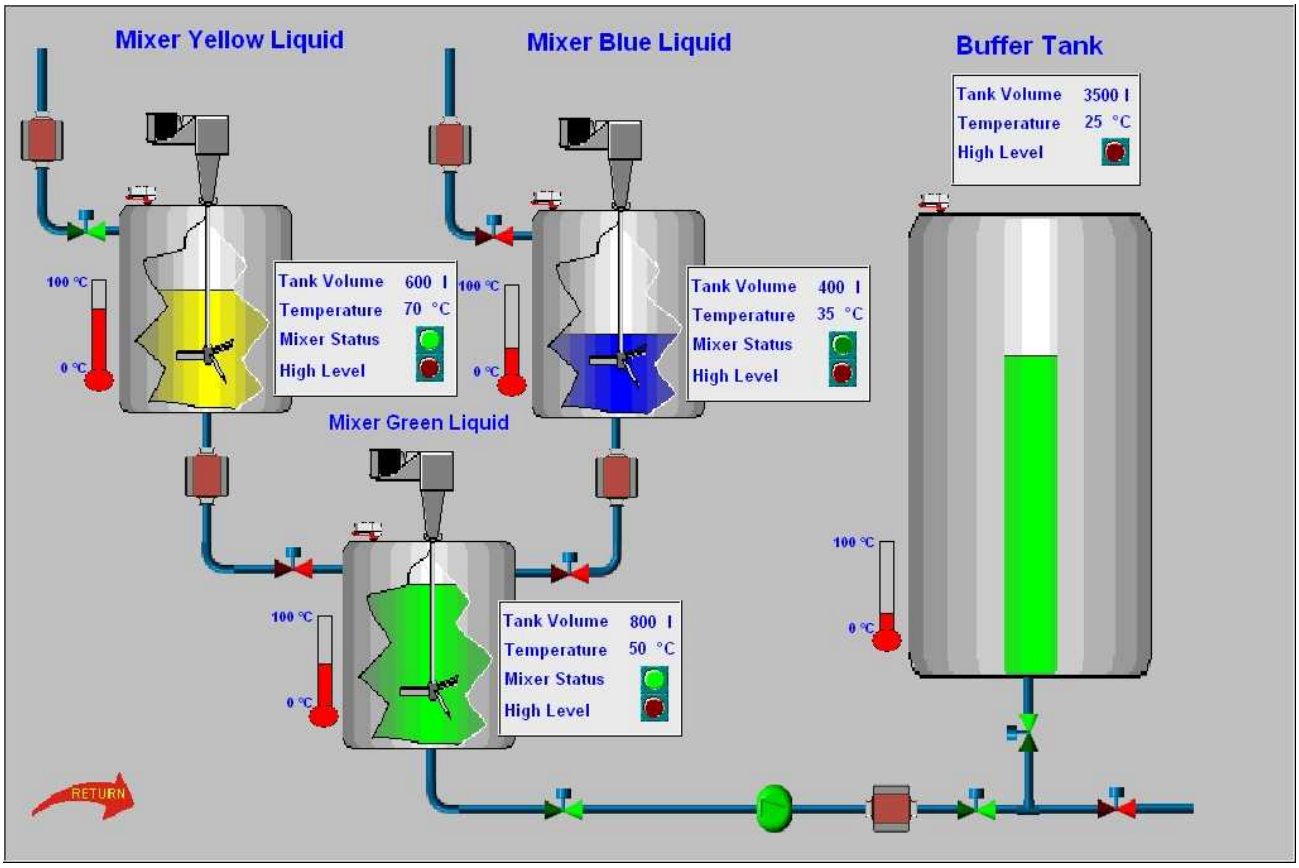

Figure 6. Conceptual PC screen showing use of components to visualise a process

3.3 Ease of use

One of the main goals of Visual VIGO has been that it is so easy to use, that the need for reading the manual is minimised. This is achieved by using the same familiar standard methods for editing elements as with other standard Windows applications. As an example, the following functions can be found in the right-hand mouse button menu: Move, Resize, Copy, Paste, Delete, Group, Ungroup, Order, Undo, Redo, Edit of properties and Help.

From the right mouse menu, windows can be opened to inspect and change the configuration of the Visual VIGO components. The configuration can relate to properties of the Visual VIGO components or parameters in a related I/O channel in the hardware.

In Visual VIGO, the use of file names and folder names is reduced as much as possible. The designer is asked for only two types of folder names:

A whole Visual VIGO design is saved under a folder name.

A *Component box* with all its components is also saved under a folder name.

Some of the images in a Visual VIGO design may be based on pictures. All pictures contained in a design can be shown in a *Picture browser*, and be selected from the browser without using file names. The pictures in Visual VIGO are automatically given unique IDs for avoiding accidental overwriting of pictures with the same file name.

3.4 Components

A Visual VIGO design is created by copying Visual VIGO components into the design from a Component box or from another Visual VIGO design.

The components are independent. New components can be copied into the *Component box*, **without** any compiling of programs: just copy and paste the components. Adding new Visual VIGO components in this way makes it possible to introduce new Visual VIGO components without the risk of changing the functionality of existing components. Visual VIGO components can be easily exchanged between different Visual VIGO users. This is performed by first copying the components to be exchanged into a Component box.. From the Component box, all components can be packed into one file, and sent by email. Visual VIGO components are automatically given unique IDs to avoid accidental overwriting of components having the same file name. The base functionality of components are contained within the Visual VIGO Class Library, which has been developed using the Delphi language. New types of components having a new functionality can be developed using Delphi by utilising the Visual VIGO class library. The Component box and its components are not required when the design is being used in *Run mode*.

Visual VIGO is designed to be utilised at different levels for:

- Operators using a Visual VIGO design created by others.
- Designers that create new Visual VIGO designs based on standard components.
- Designers that create new components by grouping a set of standard components and then copying them to the *Component box*.
- Component designers that create new components with a new appearance, but having the same functionality. For example, a Valve component can be converted into a Pump component just by exchanging the images.
- Component programmers that develop completely new component functionality.

3.5 Communication Efficiency

Visual VIGO keeps track of which components are visible, and which are hidden behind other work floors. In this way, time spent in updating values and reading variables over the network that are not visible on the screen, can be avoided. The consequent bandwidth requirement for the network is thus heavily reduced. In the same way, the components can release memory allocated for images and thus reduce the memory requirement. VIGO objects can be released when the component is hidden, and as such can, for example, close Modem connections.

4. Summary

The common usage of PC's, fieldbuses and other communication networks in today's world of process automation, building management and machine control in industrial environments, warrants a stable, extensible link between the human user and the process. The real-time and object oriented extensions that VIGO provides to a somewhat enclosed operating system such as Windows, enables a useful mix of locally and remotely located information to be assembled and controlled from a single, or network of workstations, at low cost. Visual VIGO completes the human-machine link, in offering the means to create convenient object based graphical components to display such information, which can vary from the simple clicked push button or indicating the state of a valve, to displaying tabular or graphical logged data.

References

[1] PROCES-DATA A/S, *VIGO – The Fieldbus Management System for Windows 95/98 and NT4*, Document No. 502 086 04

[2] PROCES-DATA A/S, *Visual VIGO*, Document No. 5550255-01

[3] PROCES-DATA A/S, *VIGO 4.2*, Download from URL www.Proces-Data.dk