Automation Net PVI

TM700

Perfection in Automation

www.br-automation.com

Requirements

Training modules:	TM600 – The Basics of Visualization
Software:	Windows NT/2000/XP
Hardware:	None

Table of contents

1. INTRODUCTION

B&R Automation Net offers the user a flexible and easy solution for implementing communication between control systems and between the PC and B&R control systems.

The possibilities of **Automation Net PVI** – from now on referred to simply as PVI = **P**rocess **V**isualization **I**nterface – range from simple data exchange between a visualization and a B&R controller, up to complex client / server applications that utilize the full scope[i] of the PVI.

Automation Studio uses all of the PVI functions - from simple variable exchange (watch window) up to the transfer of entire projects to the B&R controller. The PVI even handles seemingly mundane functions such as setting the time on the controller from Automation Studio or reading out various memory information.

Fig. 1: Automation Net PVI

Open access to the **Automation Net PVI** makes numerous interfaces available to the user for connecting **Windows Client Software** to PVI. These range from **programming** with the help of PVI functions to **setting parameters** and **configuring** the PVI application.

1.1 Objectives

Participants will get to know the possibilities offered by the Automation Net PVI for special visualization applications.

Windows can be used to create designs for simple visualizations and service tools up to complex, networked multi-client / server visualizations.

The various possibilities for programming, setting parameters and configuring will also be discussed.

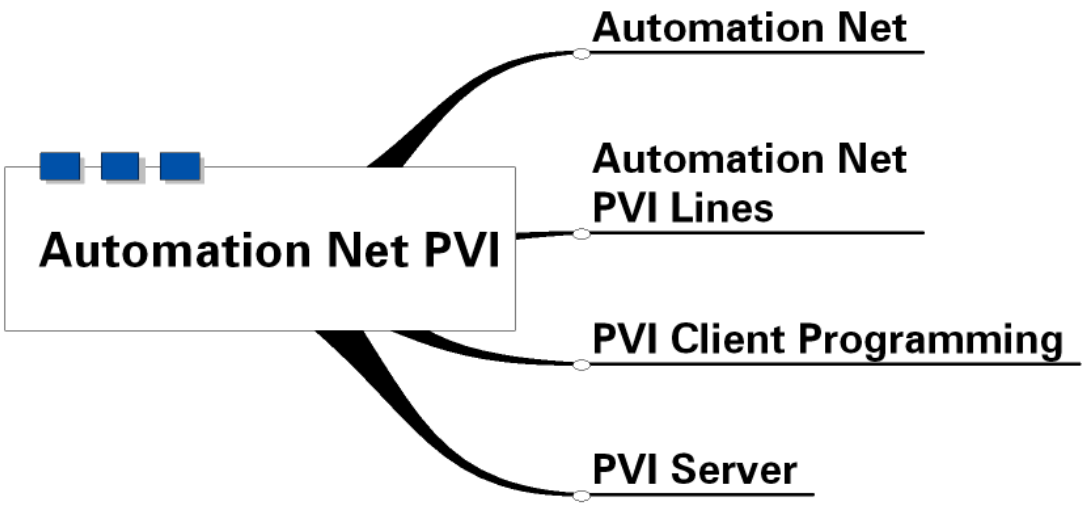


Fig. 2: Overview

2. B&R AUTOMATION NET

B&R Automation Net offers complete, system-wide **communication** between Windows-based industrial PCs, drives, controllers, operating panels and programming tools.

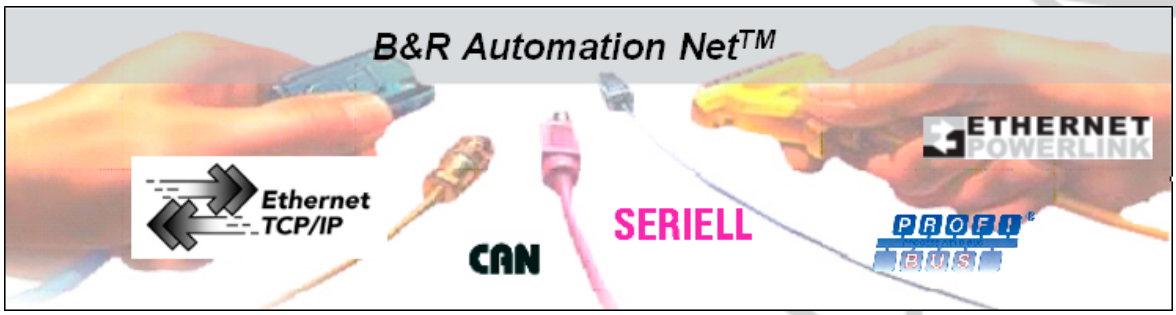


Fig. 3: Automation Net

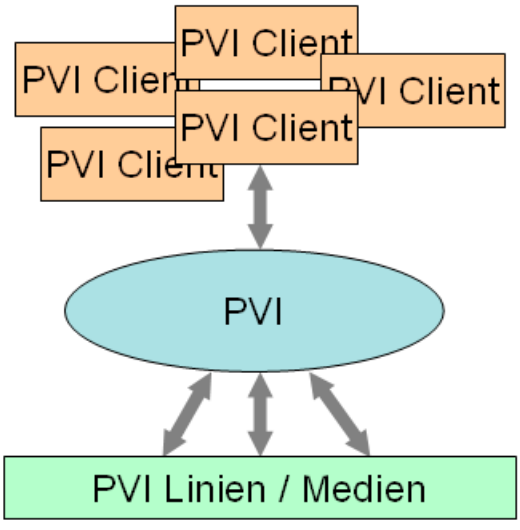
2.1 Access to Automation Net

The controller provides the user with various libraries for exchanging media-independent data between intelligent controllers (CPUs).

PVI, which is used in **Windows** to exchange data between the PC and intelligent controllers (CPUs) independent of the media, platform and protocol.

The PVI concept can be generally divided into 3 areas:

- PVI clients (Windows applications)
- PVI component (**PVI Manager**) for managing the process data
- PVI lines for managing the communication types (protocol) and communication medium



Multiple PVI applications can run at the same time, independent of the communication type that is being used.

This means that the communication is not only limited to the exchange of process data, but additional PVI services can also be used for various applications.

2.2 Automation Net Windows interface

The PVI base system is the central access channel to B&R Automation Net when using Windows NT/2000/XP or Windows CE.

PVI offers a common interface to the world of the B&R industrial PCs for all Windows-based software packages.

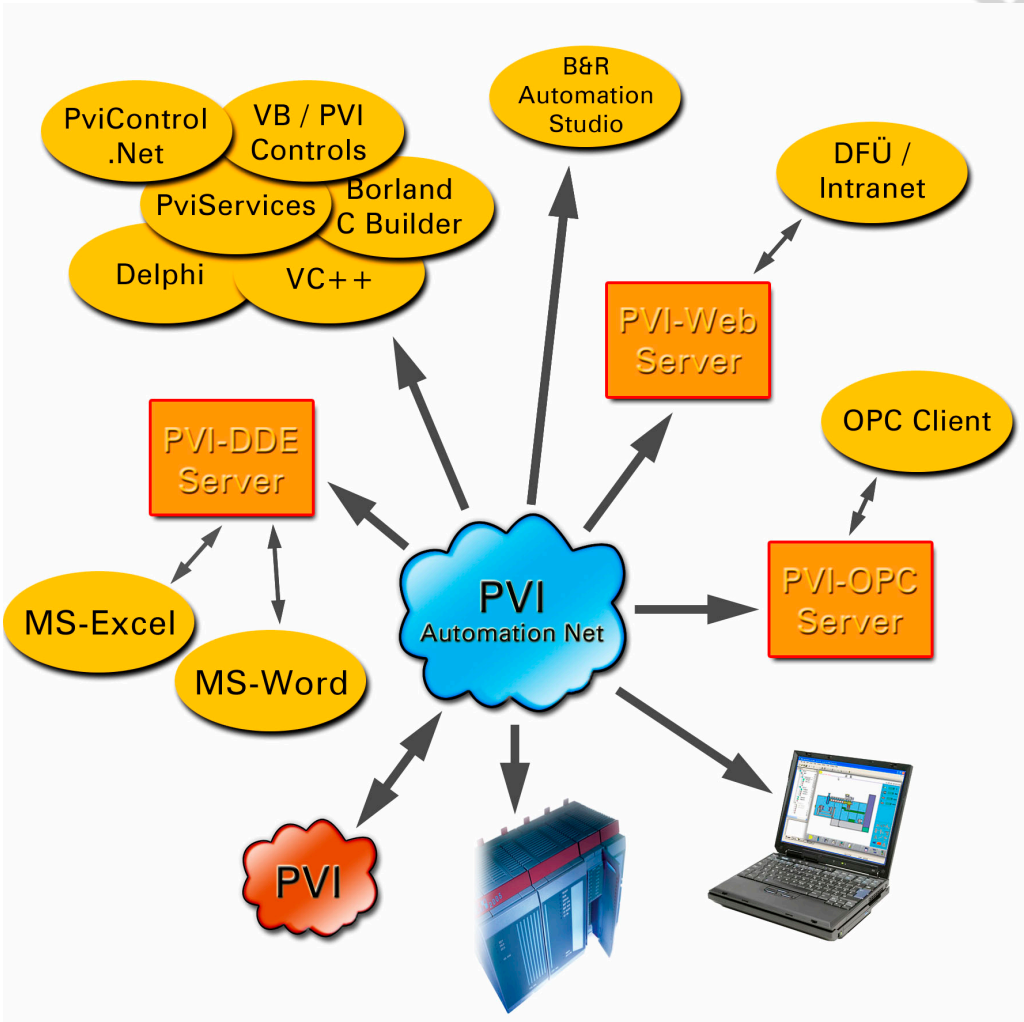
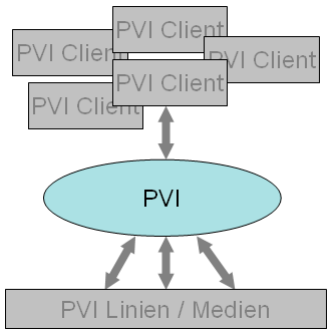


Fig. 4: Windows interface to Automation Net

2.3 PVI Manager



The PVI Manager (central component of PVI) handles the management of all types of process data, from simple process variables to lists, programs or data modules.

The PVI Manager organizes the process data according to chronology as well as direction. That means the PVI Manager coordinates the data transfers from the user configuration (direction, protocol, medium, device, etc).

Special attention is given to asynchronous management in order to be able to fit in e.g. network delays for other tasks or coordination of event processing between other tasks.

PVI Manager features:

- Central PVI component
- Object-oriented, hierarchical management of all process data
- Management of process data based on both timing and direction
- Management of multiple processes and stations via client / server architecture
- Event-driven data acquisition
- Data conversion, linearization, hysteresis
- Dynamic connection description for the PVI objects
- Independent of the programming language and Windows platform
- Scalable runtime system

2.3.1 PVI object hierarchy

All PVI Manager processes are managed in an object structure.

A process object is defined by its object name, the object type and the connection description.

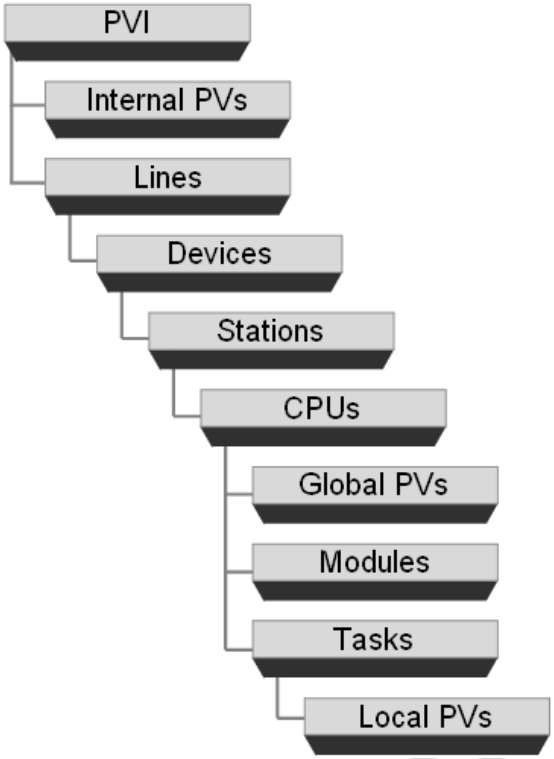


Fig. 5: PVI object hierarchy

It is also possible for a process object to be defined multiple times in a hierarchy (e.g. when communication with multiple controllers is necessary).

Example:

Object name and connection description for controller #1:

@/Pvi/LnIna2/TcpIp/Station/CPU1 ***CD="/DA=2 /DAIP=10.0.0.2"***

Object name and connection description for controller #2:

@/Pvi/LnIna2/TcpIp/Station/CPU2 ***CD="/DA=3 /DAIP=10.0.0.3"***

2.4 PVICOM interface

The PVICOM interface (client interface) establishes access to the PVI at the lowest level.

This represents the "tightest" and, regarding performance, the most optimal PVI interface.

The PVICOM interface is also used by all other Windows-based components which have PVI access (PVI OPC, PVI DDE, PVI services, etc.).

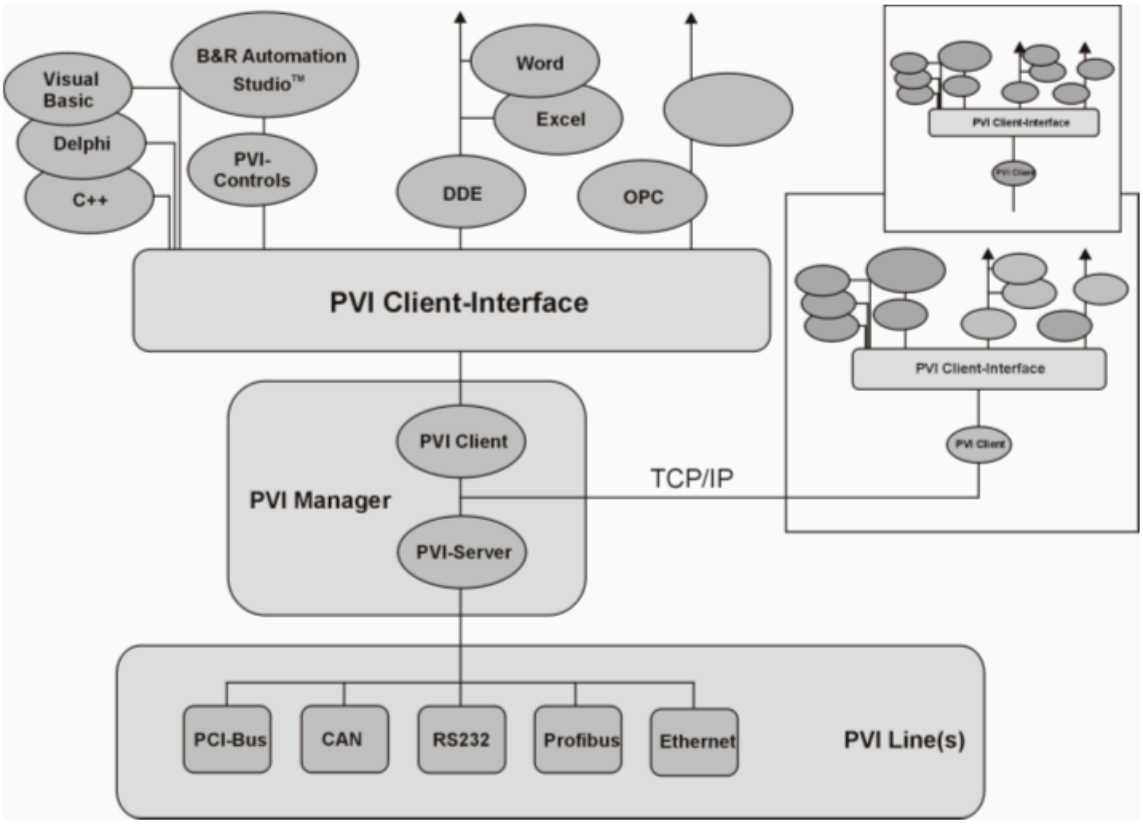


Fig. 6: PVI interface

The PVICOM interface acts according to the client/server principle. The PVI Manager acts as the server, and the PVICOM applications are the clients. Servers and clients can be on the same machine (local communication type) or on different computers (remote communication type).

2.5 PVI installation packages

The PVI installation packages are part of the Automation Software CD. There are 2 packages:

- PVI Runtime
- PVI Server & Runtime / Development

The **PVI Runtime** package contains the main components of the PVI, all lines (protocols) and media (interface drivers).

Furthermore, the **PVI Server & Runtime** or **Development** package contains all of the DLLs and programming modules necessary for programming, all available servers (OPC, WEB, DDE, etc) the PviServices, PVIControls.NET DLLs and controls and the PVI help files.

The latest versions of the PVI installation packages can also be downloaded from the B&R homepage.

2.6 PVI license

PVI can be run on all B&R PCs without an additional Runtime license.

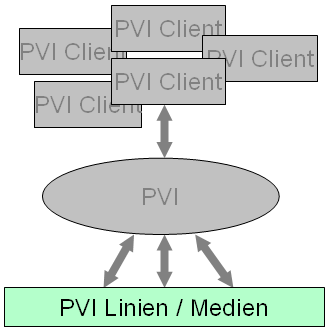
A dongle is required on PCs from other manufacturers (LPT, USB, company license DLL). A PVI application can run on these PCs without functional limitations for a maximum of 2 hours in trial mode (e.g. PviTransfer, etc).

- 5S0500.02 (LPT)
- 5S0500.02U (USB)
- 5S0500.99 (company license DLL)

An additional dongle is required for PviControls.NET.

- 5S0510.02 (LPT)
- 5S0510.02U (USB)
- 5S0510.99 (company license DLL)

3. AUTOMATION NET – PVI LINES



Access to the PVICOM interface does not depend on the protocol or medium being used.

The basic task of a PVI line is to connect PVI objects to objects outside of PVI. The line is also responsible for communicating with B&R controllers and determines the communication protocol to be used to do so.

Supported protocols / PVI lines:

- INA2000 line (System2000 online protocol)
- NET2000 line
- CANdirect line
- MININET line
- ARCNET OS9 line
- MTC line / ADI line

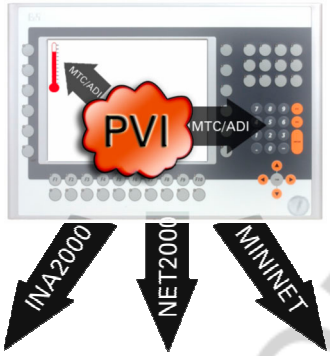


Fig. 7: Overview of PVI lines

3.1 INA2000 line

The INA2000 protocol (**I**ndustrial **N**etwork **A**rchitecture System **2000**) corresponds to the Automation Studio online protocol for the control generation SG3 and SG4.

3.1.1 INA2000 services

In addition to the exchange of variables, all of the online services used in Automation Studio are also available to the user (module transfer, memory manipulation, info services, etc).

3.1.2 INA2000 topology

The INA2000 protocol can be used to establish both a simple point ⇔ point connection as well as a networked multi-master / multi-client architecture.

The Automation Runtime operating system allows you to "route" INA2000 frames that are addressed to another controller. When doing so, the medium can also be changed at any time.

The only requirement for the controller is the configuration of the interface for the INA2000 protocol.

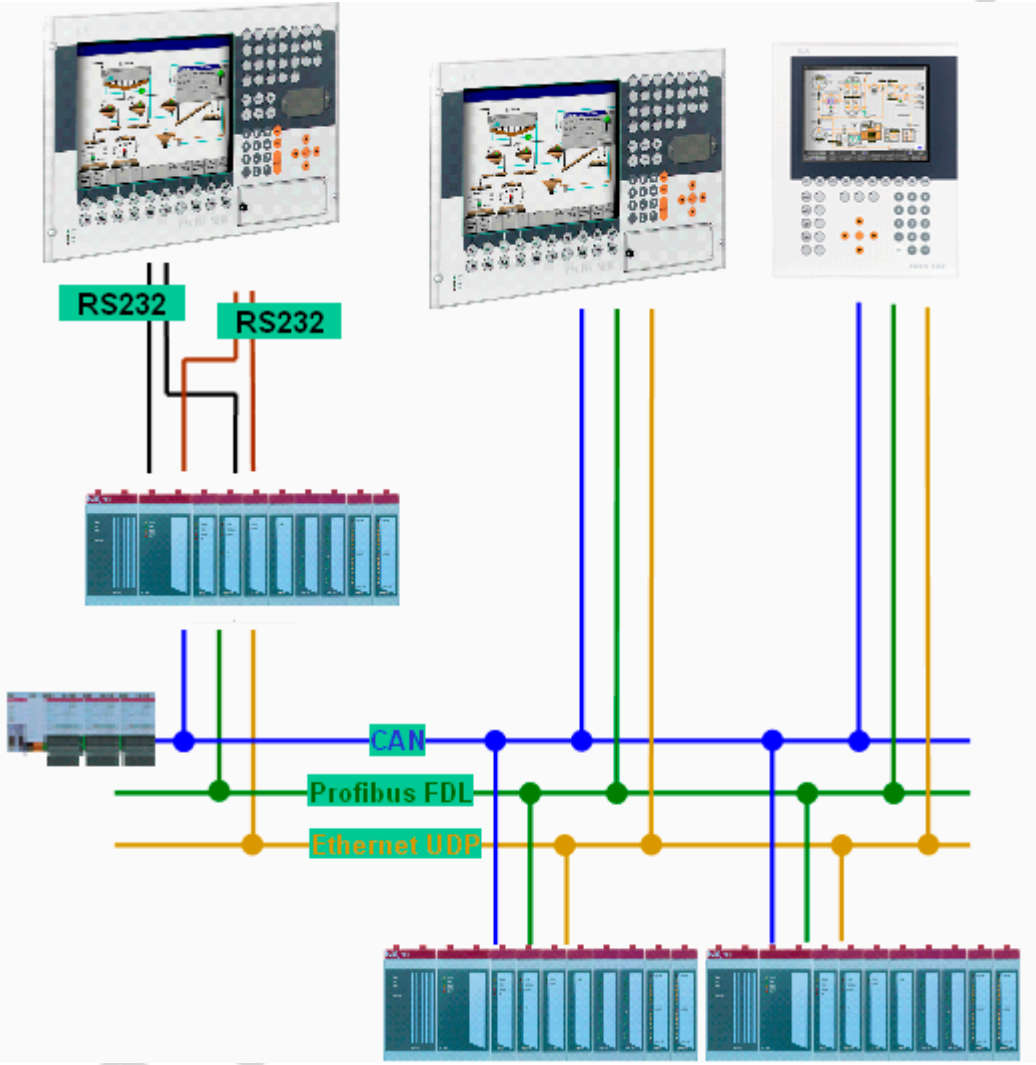


Fig. 8: INA2000 topology

3.1.3 INA2000 media

In principle, all of the interfaces available on the controller (except for RS485) can be configured for INA2000 communication.

Either a point ⇔ point connection or a network connection to one or more controller(s) is possible, depending on the media being used.

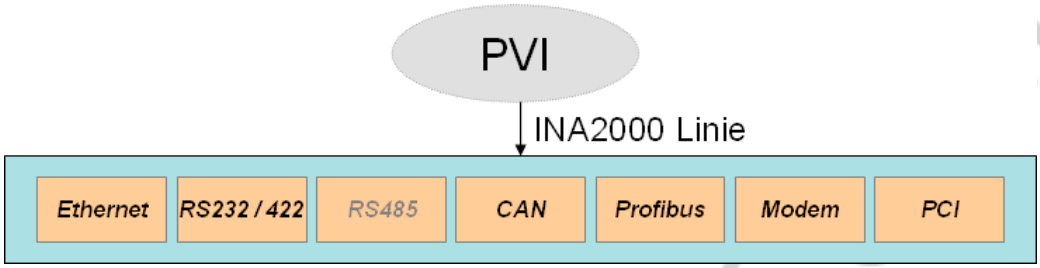


Fig. 9: INA2000 media

Information about setting up the online interface in Automation Studio can be found in the Training Module TM211 – Automation Studio Online Communication.

3.2 NET2000 line

Master – slave communication to SG3 and SG4 control systems can be established using the NET2000 line.

Caution:

Only one station (PC) master is allowed in an INA2000 network. Multi-master communication is not possible.

3.2.1 NET2000 services

Only variable services are supported in the NET2000 line.

3.2.2 NET2000 topology

The NET2000 protocol can be used to establish a simple point ⇔ point connection via RS232 / RS422 or PCI (LS251) as well as master – slave communication via a RS485 interface.

Note:

A NET2000 slave function block is required on the controller.

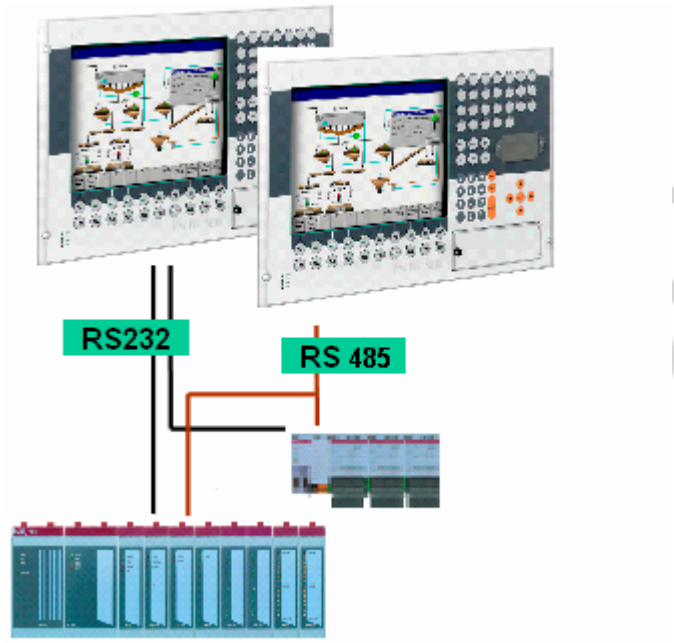


Fig. 10: NET2000 topology

3.2.3 NET2000 media

All of the RS232, RS422 and RS485 interfaces on SG3 / SG4 can be used for NET2000 communication.

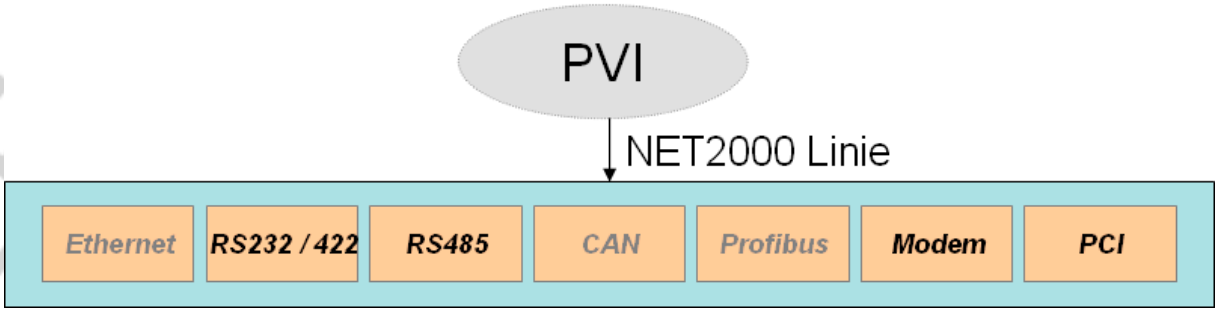


Fig. 11: NET2000 media

3.3 CANdirect line

The CANdirect line can be used to establish event-controlled communication with "unintelligent" CAN nodes.

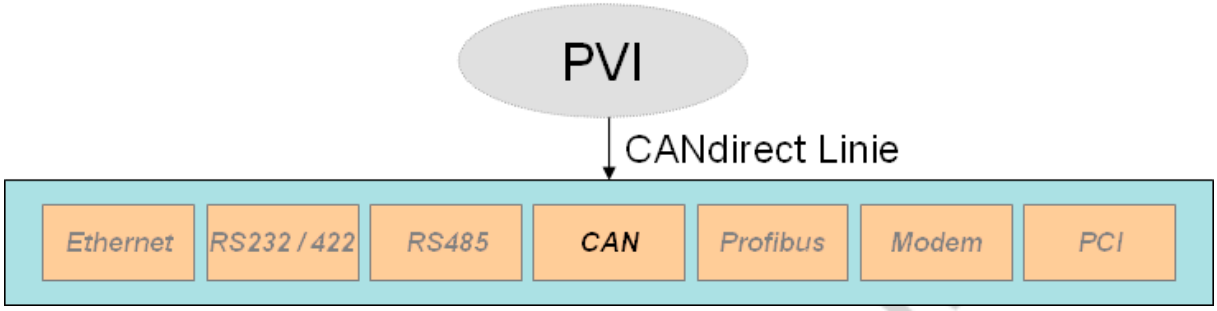


Fig. 12: CANdirect media

3.4 MININET line

The Mininet line can be used to exchange data with SG2 control systems via a serial interface.

Communication to multiple controllers is possible.

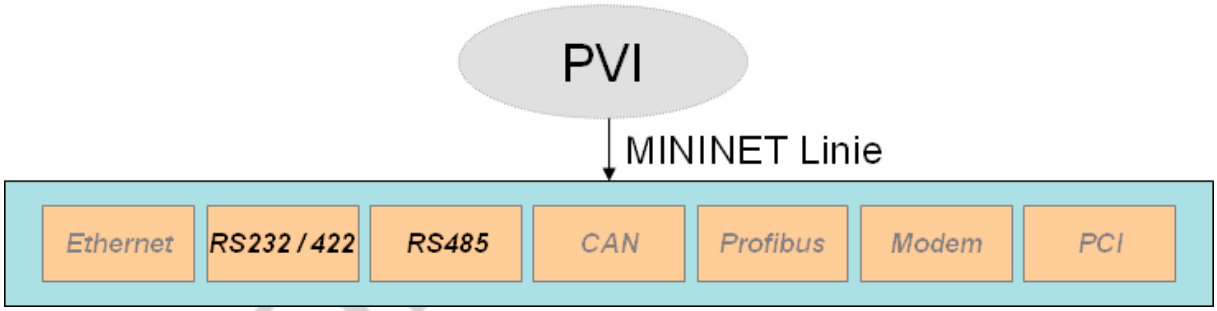


Fig. 13: MININET media

3.5 ARCNET OS9 line

The Arcnet OS9 line can be used to exchange data with SG2 control systems (MIDI / MULTI) via a point ⇔ point connection or a network connection.

Note:

This line can only be used in Windows NT up to Service Pack 5.

3.6 MTC line / ADI line

The MTC (IPC Maintenance Controller) and ADI (APC Automation Device Interface) lines can be used to evaluate the functions available for the corresponding device types via PVI variable objects.

Fig. 14: MTC / ADI line

The MTC line provides Maintenance Controller (MTC) functions for the B&R Provit 5000 industrial PCs.

The ADI line provides functions for accessing display, keys and LEDs on the following devices:

- Automation PC 620
- Panel PC 700
- Power Panel 100 BIOS
- Mobile Panel 100 BIOS

With the help of pre-defined variable names, these lines can be used to read or to set properties of the PC hardware such as:

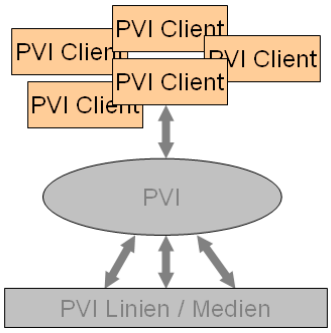
- CPU and IO board temperature
- Fan RPMs and runtime
- Version information about the hardware and firmware
- Evaluation of keyboard and LEDs
- Display – brightness, contrast and background lighting

Note:

The functions and possibilities depend on the hardware being used (IPC, APC, display, etc)

4. PVI CLIENT PROGRAMMING

Communication via the PVICOM interface is handled with the functions in the PVI communication library, "PviCom.dll". The PVI communication library is a DLL (Dynamic Link Library) based on the Windows standard.



Depending on the programming environment, the functions of the PVICOM interface are used directly or are provided for the user controls or classes, which encapsulate the respective functions.

4.1 PVICOM.DLL

The following 32-bit programming environments are supported:

- Visual C++ (Version 6.0 and up)
- Visual Basic (Version 6.0 and up)
- Borland C++ Builder (Version 3 and up)
- Borland Delphi (Version 4 and up)

PviCom.dll is accessed via PVI functions. The corresponding definition files or declaration files are available for each programming environment.

- PviInitialize(...)
- PviCreate(...), PviLink(...)
- PviReadRequest(...)
- PviReadResponse(...)
- PviWriteRequest(...)
- PviWriteResponse(...)
- PviDelete(...), PviUnlink(...)
- PviDeinitialize(...)

Application notifications are made using Windows Post Messages or Callbacks.

Note:

Programming in Windows CE is performed using Embedded Visual Studio.

4.2 PVIServices

PviServices components are aimed at users who want to use communication and diagnostics services based on the PVICOM components within the Microsoft .NET development platform for B&R controllers.

Object-oriented processing of the PVICOM functions results in a clear and logical image of process and control-specific data.

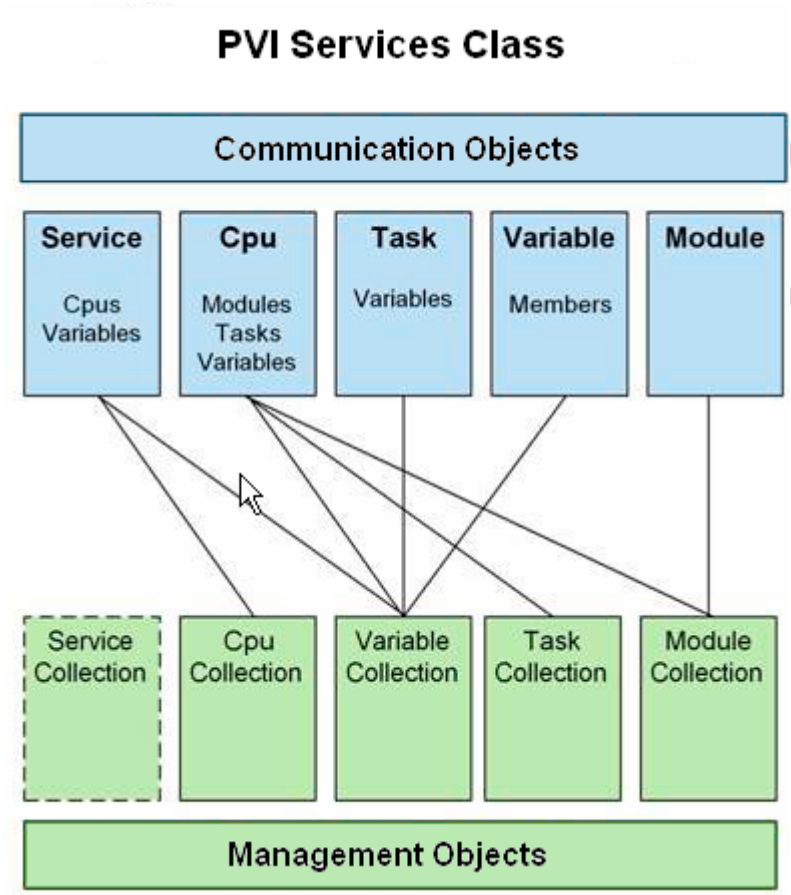
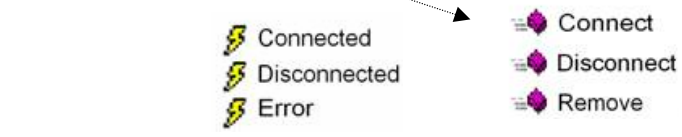


Fig. 15: PVI Services - classes

A communication object basically represents an object located on the controller (e.g. task or process variable). Each of these communication objects contains:

- Basic functions
- Basis properties
- Basis events



These basis services guarantee consistent and uniform work with the communication objects. For example, the "Connect" function means that a connection should be established to the respective process object on the controller for both a task object as well as a variable object.

Example...

```
// Definition of global communication objects
Service service;
Cpu cpu;
Variable variable;
/// <summary>
/// Generate and connect service object
/// </summary>
private void btConnectPLC_Click(object sender, System.EventArgs e)
{
  if ( service == null )
  {
    service = new Service("service");
    service.Connected+=new PviEventHandler(service_Connected);
  }
  service.Connect();
}
/// <summary>
/// Connect CPU object if service object connection successful
///</summary>
private void service_Connected(object sender, PviEventArgs e)
{
  if ( cpu == null )
  {
    // Create CPU object and add the event handler
    cpu = new Cpu(service,"cpu");
    cpu.Connected+=new PviEventHandler(cpu_Connected);
    // Set the connection properties for a serial connection
    cpu.Connection.DeviceType = DeviceType.Serial;
    cpu.Connection.Serial.BaudRate = 57600;
    cpu.Connection.Serial.Channel = 1;
  }
  // Connect CPU
  cpu.Connect();
}
```

Note:

The PVI Services component can be used in both Win32 and Windows CE applications.

Only the INA2000 line with its media is supported from the PVI Services.

4.3 PVIControls.NET

PVIControls.NET is a component based on the PVIServices and .NET Framework 1.1 for the Visual Studio.NET development environment for Win32-based operating systems.

The user is provided with a simple interface between the programming environment Visual Studio .NET and the PVI.

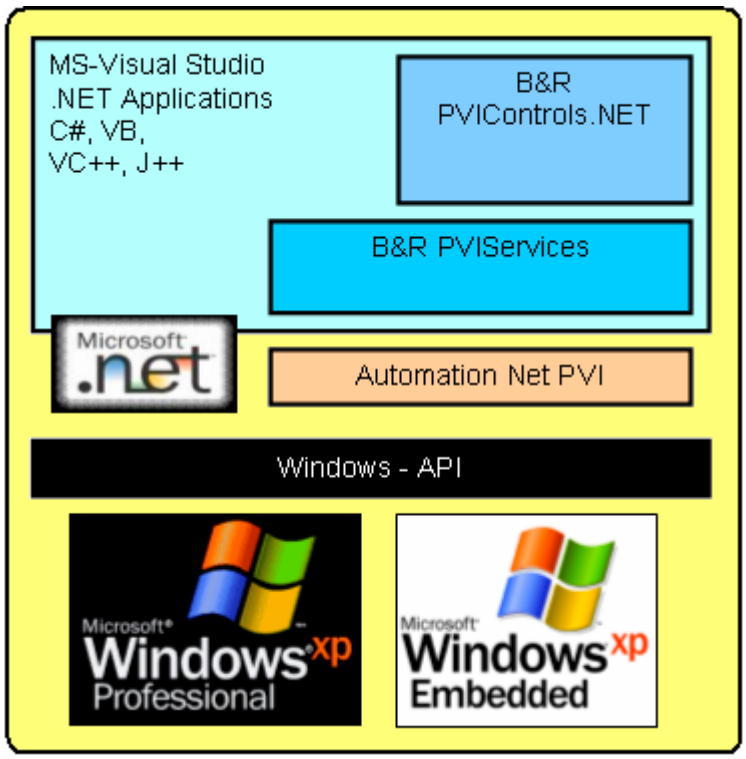


Fig. 16: PVIControls.NET

4.3.1 PVIControls.NET components

PVIControls.NET is made up of the following components:

- ACConfigurator – management of the control variables and resources
- ACComponent – central component on every page for development time and runtime

4.3.2 The ACConfigurator

All of the data required for the ACComponents is managed in XML files in the AC Configurator. Language-dependent resources are saved and managed in separate files.

- Variable configuration and variable properties
- Text, scaling and format resources
- Connection configuration, interface configuration
- Alarm and trend configuration

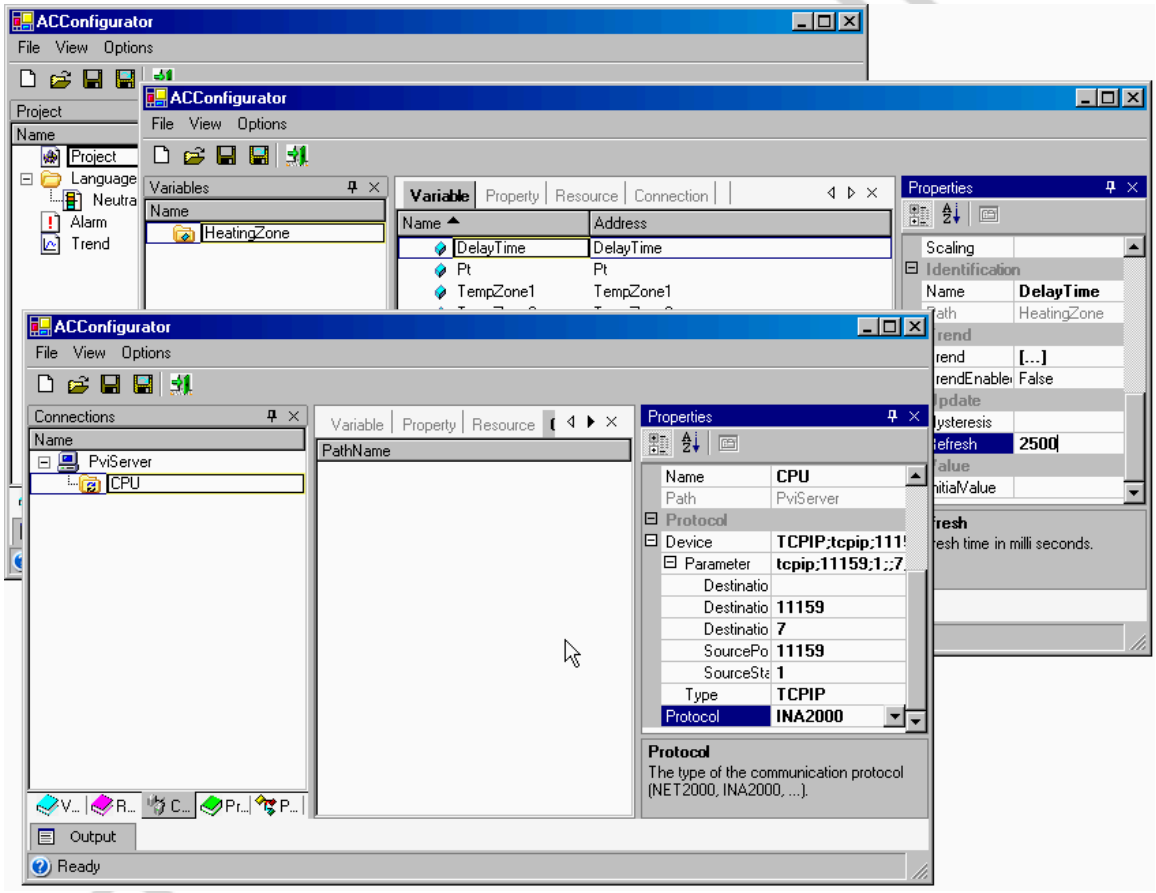


Fig. 17: AC Configurator

The variables for the logical groups are linked with a physical PVI object (e.g. task object) in the configuration.

4.3.3 ACComponent

The **ACComponent** is the core component for PVIControls.NET applications. The main function is to perform the simplest possible exchange of data between process variables and the control elements in a visualization, which then process the data.

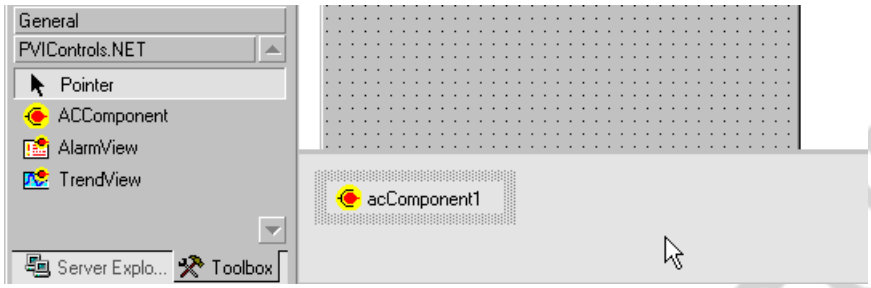
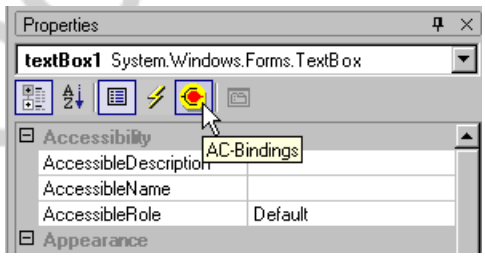


Fig. 18: ACComponent

During the design phase, ACComponent provides support in defining links between the properties of control elements and process variables or resources. Property views that have been specially integrated in VisualStudio.NET allow the available visualization application resources to be selected and assigned to the corresponding control element properties.



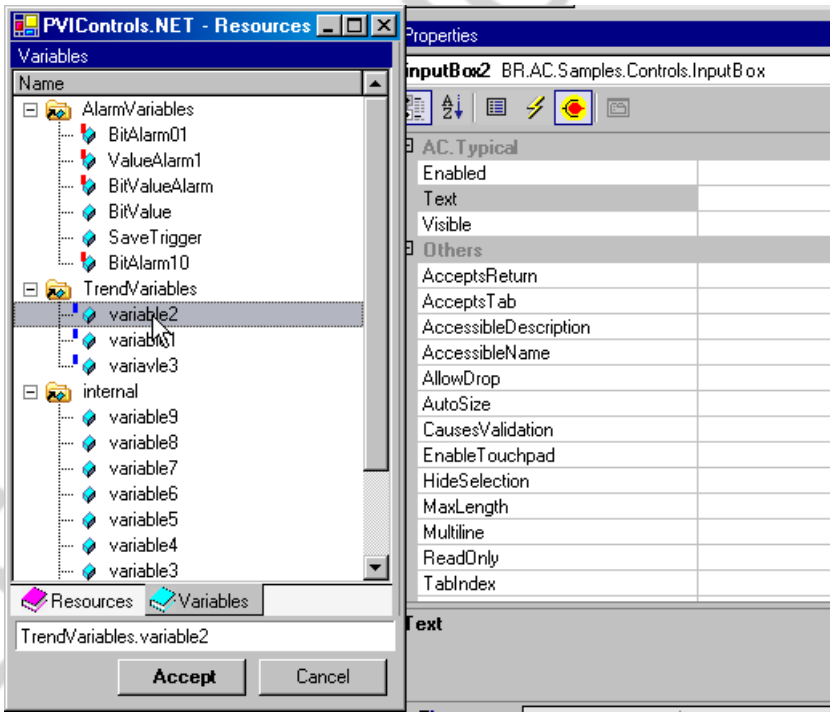


Fig. 19: ACComponent link view

These links are automatically enabled at runtime, i.e. data exchange takes place between control element properties and process data.

4.3.4 AlarmView control

The AlarmView control displays historic or current alarm states recorded by the PviAlarm Server.

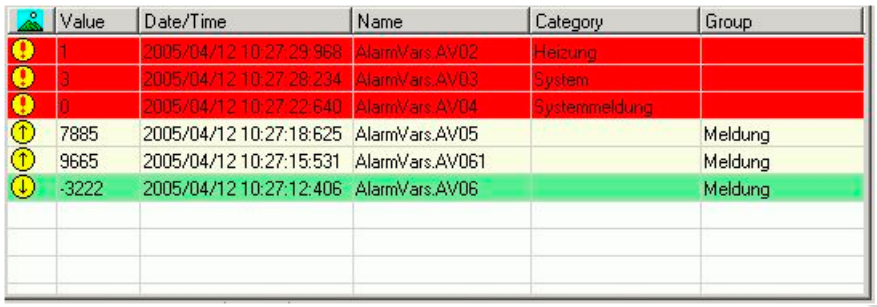


Fig. 20: AlarmView control

The view and the alarm lists that should be used are configured via the properties of the AlarmView control.

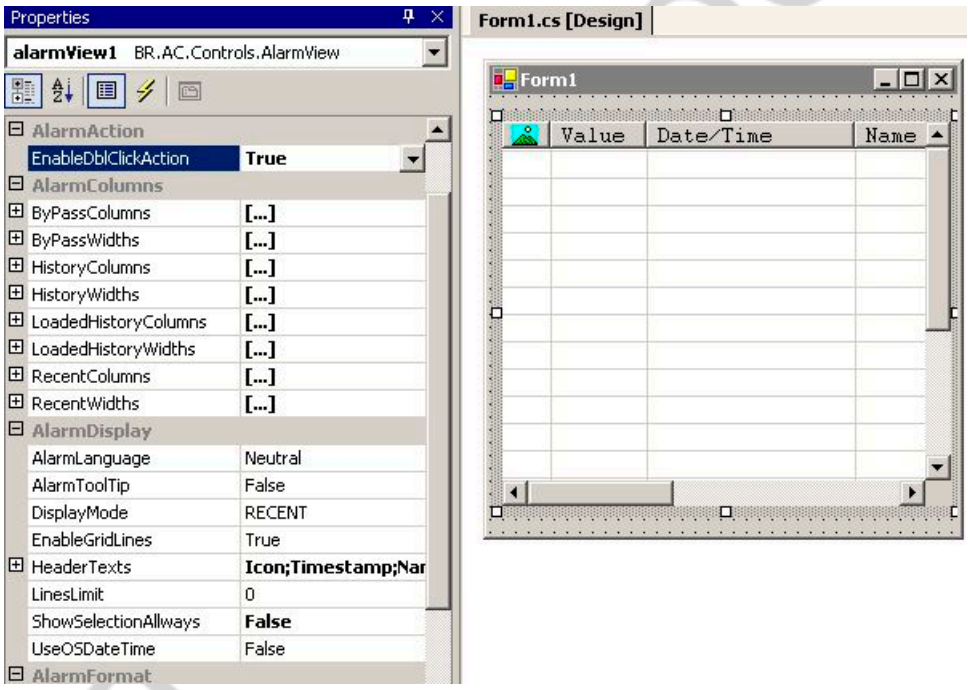


Fig. 21: AlarmView configuration

4.3.5 TrendView control

The TrendView control uses a curve display to show variable states recorded by the PviTrend server.

- Historic administration of the trend configuration in the Trend server
- Display can be configured for historic or online view
- Any number of channels
- X(t), Y(t), Y(X), superimposed or separated
- Automatic or manual scaling for different channels
- Line, layer or bar display
- Trend scales can be shown in multiple languages
- Visibility and channel can be configured freely
- Measurement and reference cursor
- Zooming, scroll functions
- Operation via touch system is possible

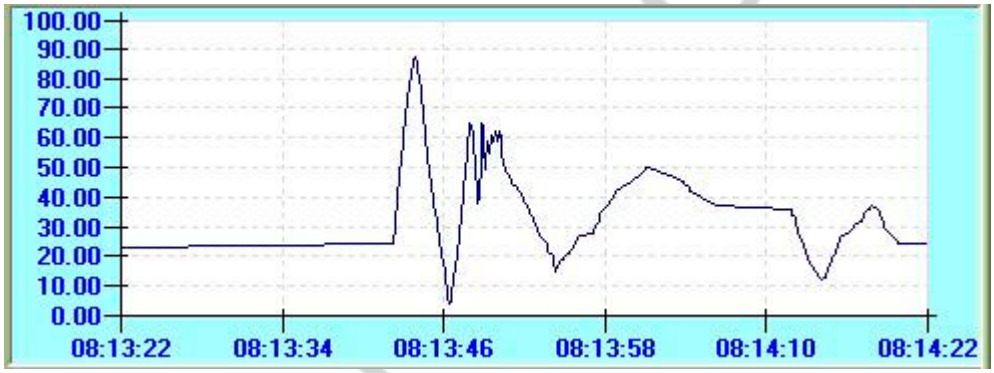


Fig. 22: TrendView Control

The display and assignment of the trend data points are configured via the properties of the TrendView control.

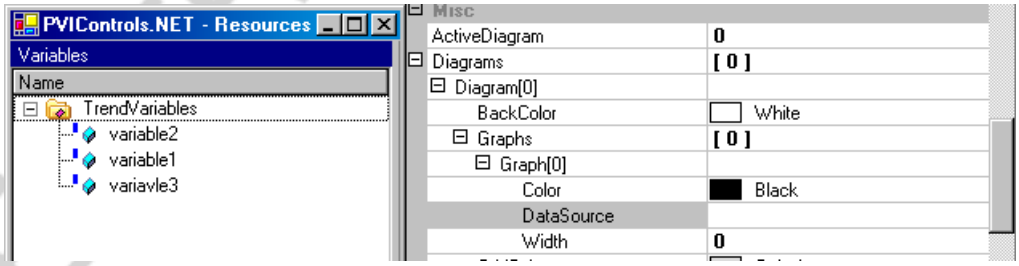


Fig. 23: TrendView control – Assigning the trend data points.

4.4 PviControls (VB6.0)

PviControl provides users in Visual Basic 6.0 with an easy-to-use interface to the PVI. In addition to configurators for managing variables and resources, it also offers complete ActiveX controls.

"Configuring instead of programming"

Direct access to the PVICOM interface means less programming effort for the user and lower runtime costs.

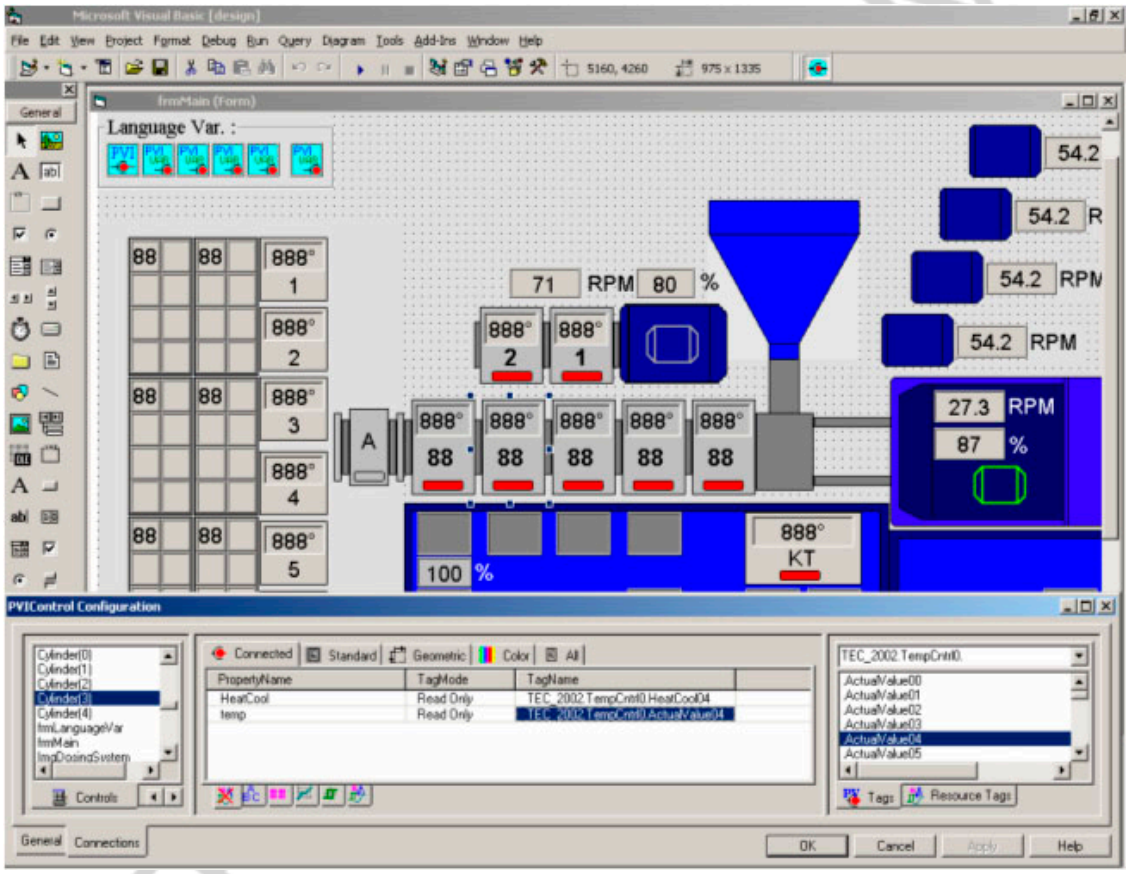


Fig. 24: PviControl concept

The PviControl consists of the following components:

- PviConfigurator – Management and configuration of variables
- PviRessourceEditor – Management of language-dependent text, formats, scaling, etc.
- PviActiveX controls – Unicode-capable controls for displaying and entering values or text, buttons, list displays, etc.
  Displays alarms and trend curves
- Alarmserver – Management and recording of alarm states
- Trendserver – Management and recording of trend variables

4.4.1 PVI Configurator

In the PviConfigurator, variables and their attributes are placed in logical groups and connected to a PVI object.

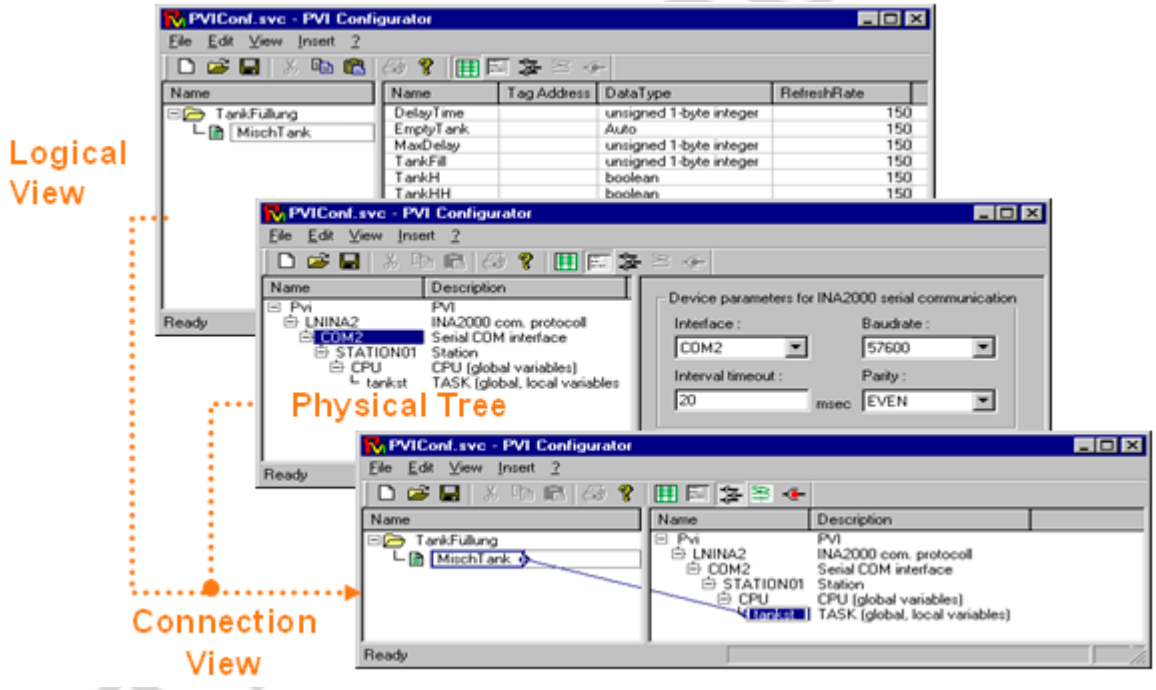


Fig. 25: PviControl configurator

PviConfigurator features:

- Dialog-controlled definition of interface parameters
- Structured variable declaration and assignment
- Variables imported from Automation Studio and PG2000 projects
- Assignment of logical variable groups to a PLC connection
- Definition of variable properties
- Definition of trend & alarm data points
- Dialog menus used to guide through setting general trend & alarm parameters

4.4.2 PviRessource Editor

All of the text, scalings, formats and hystereses in the visualization are configured in the PviRessource Editor.

These resources are displayed during runtime - depending on the language. Configuration is possible for static resources as well as dynamic resources that are switched during runtime by using a variable.

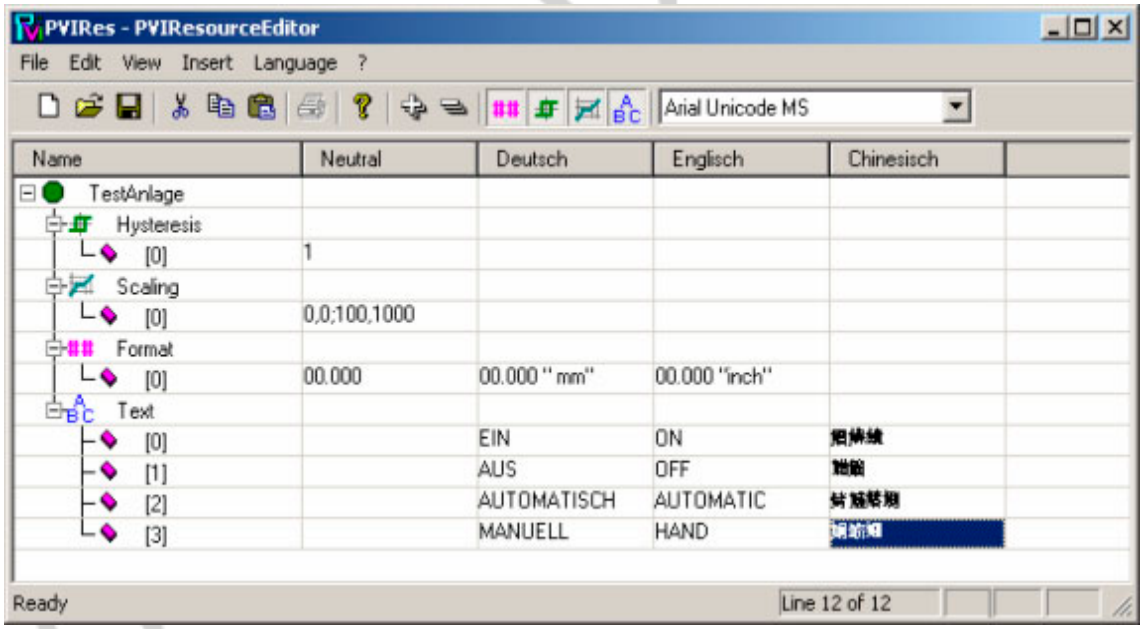


Fig. 26: PviControl Resource Editor

PviRessourceEditor features:

- Structured input of text, scaling and format
- Reference-controlled access to all resources
- ASCII export and import (CSV format) for external translation
- Clear division of dynamic and static text
- Unicode-capable (incl. export/import)

4.4.3 ActiveX controls

In addition to the individual configurators, the PviControl components also offer some ActiveX controls that can be used for displaying and operating the visualization. Unlike the standard Visual Basic controls, these controls are UNICODE-capable and have a few additional functions.

PviActiveX control

The PviActiveX control is the central component on any Visual Basic form during development and runtime.

- Linking the VB-Controls properties to variables and resources
- The PviActiveX control organizes the process variables during runtime
- Automatic refresh of the process variables in the process image
- Evaluation of configuration and runtime errors
- Implementation of language switching

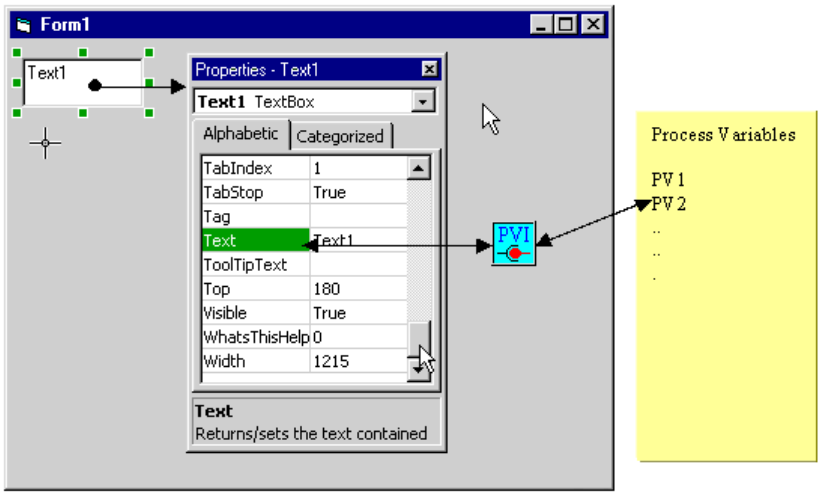


Fig. 27: PviActiveX control

PviTextBox control

In addition to the standard VB Textbox control, the PviTextBox control also offers some expansions, which allow the user to configure the behavior of the control for both inputting and outputting values and text.

- Configurable alignment
- Definition of the input and output behavior
- Limit value monitoring
- Color change for the input modes can be defined variably
- Confirmation of the write task via event function
- Activation of the TouchScreen KeyPad controls for numeric and alphanumeric input
- Switching the key layout
- Different TouchPad sizes

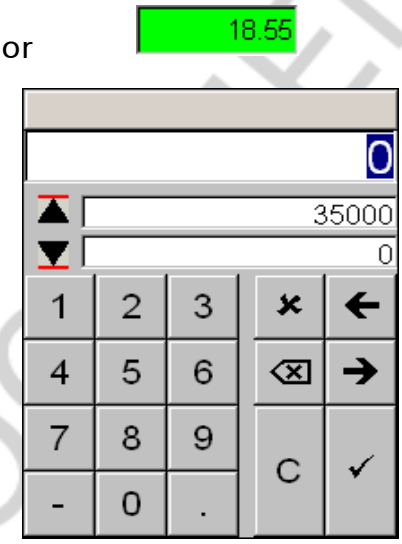


Fig. 28: PviTextBox control - TouchPad

PviButton control

The PviButton control can be used as button as well as label control.

- Configurable for switching and key functions
- "Caption" property also suitable for Unicode characters
- Separate bitmaps for the states "pressed", "released" or "disabled"

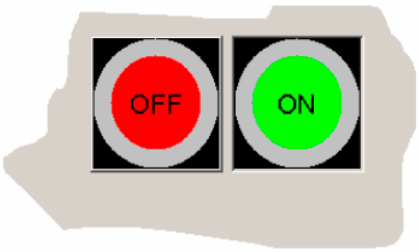


Fig. 29: PviButton control

PviListBox control

During runtime, the PviListBox control can be linked with dynamic text resources or filled with text.

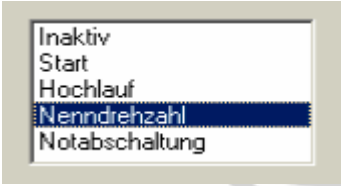
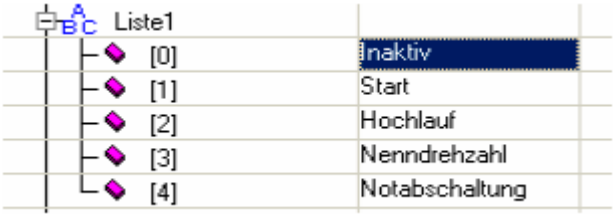


Fig. 30: PviListBox control

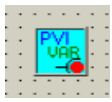
- Connecting a text resource to the "List" property



- Using the PviListBox control as combo box
- Evaluating the list index using configurable completion behavior via the "Value" property

PviVarAccess control

The PviVarAccess control is used to receive hidden value changes in the VB code or to write the control variables connected to the control.




```
Private Sub TempActValue_Changed(ByVal Value As Variant)

        If (Value = True) Then
                MotorSetRPM.Value = 1000
        End If

End Sub
```

PviAlarmView control

The PviAlarmView control displays historic or current alarm states recorded by the PviAlarm Server.

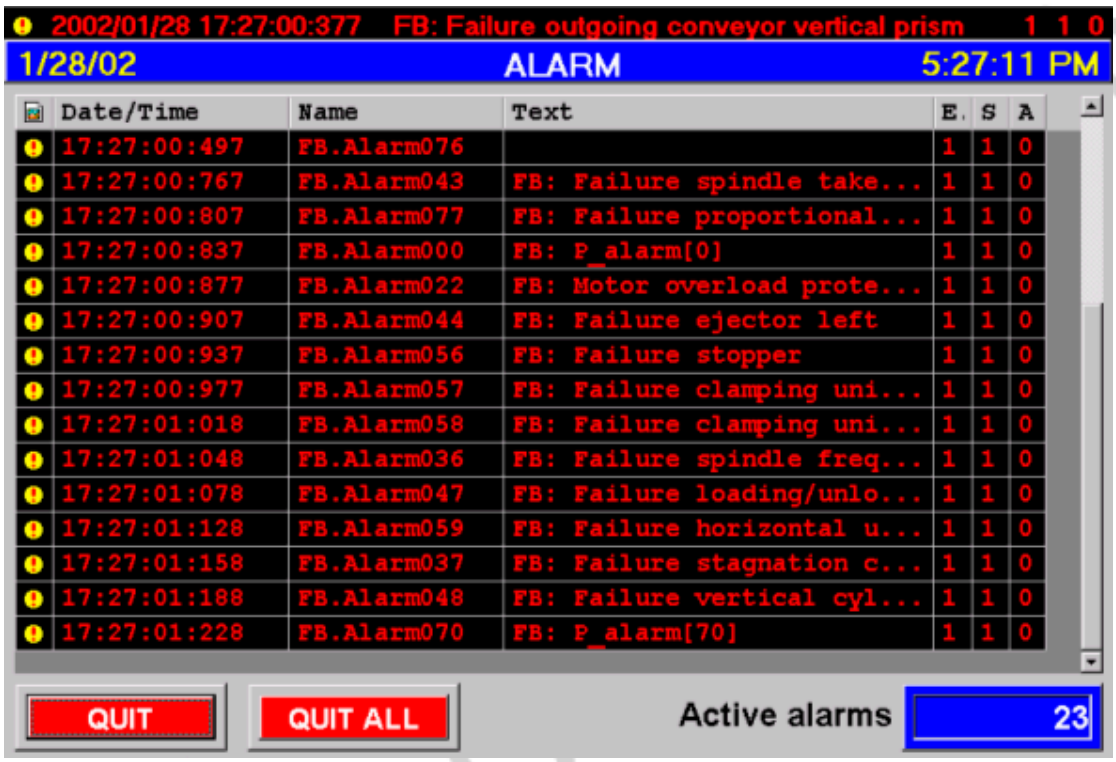


Fig. 31: PviAlarmView control

PviAlarmView properties

- Configurable display and column properties
- Displays active alarm lists, alarm history
- Current alarm status
- AlarmAPI for creating a separate alarm system

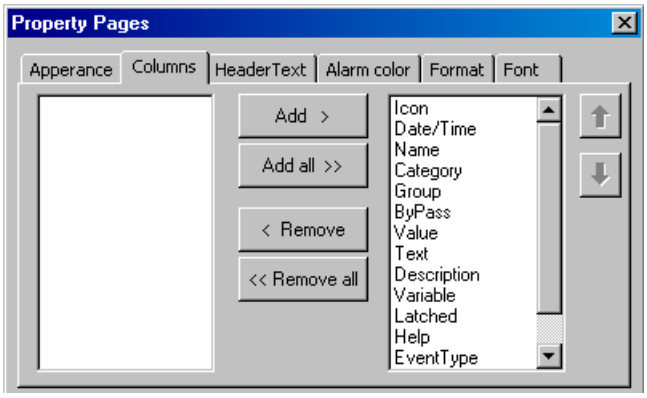


Fig. 32: PviAlarmView properties

PviTrend control

The PviTrend control uses a curve display to show variable states recorded by the PviTrend server.

- Historic administration of the trend configuration in the Trend server
- Display can be configured for historic or online view
- Any number of channels
- X(t), Y(t), Y(X), superimposed or separated
- Automatic or manual scaling for different channels
- Line, layer or bar display
- Trend scales can be shown in multiple languages
- Visibility and channel can be configured freely
- Measurement and reference cursor
- Zooming, scroll functions
- Operation via touch system is possible

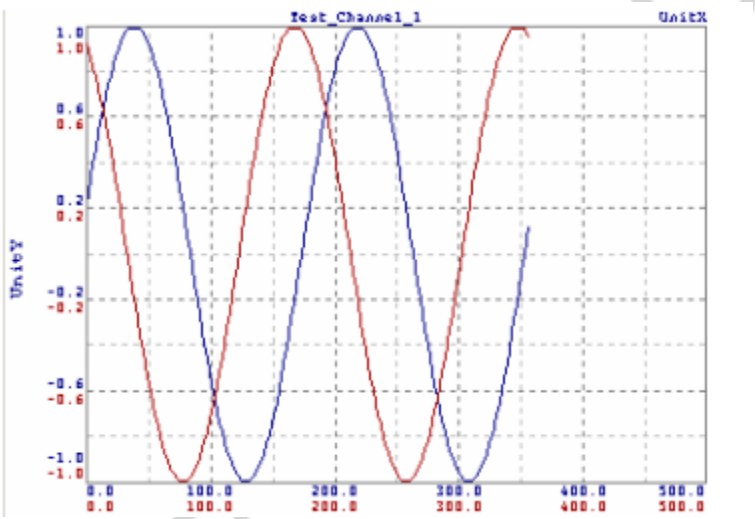


Fig. 33: PviTrend control

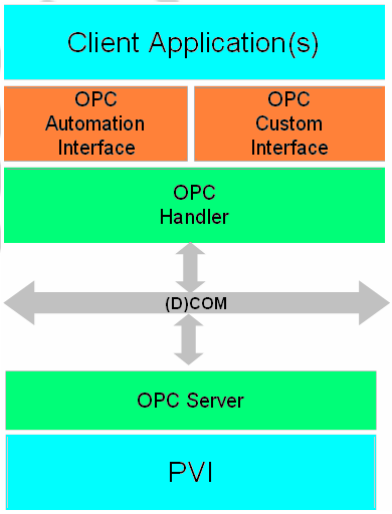
5. PVI SERVER

The PVI server is used to apply interfaces and functions specified for the Win32 client to functions for the PVICOM interface.

5.1 PVI OPC server

OPC (OLE for process control) is an industrial standard that was created with the participation of numerous worldwide leaders in automation and hardware manufacturers in cooperation with Microsoft. The OPC Foundation is the organization that manages this standard.

OPC is based on Microsoft's OLE (Object Linking and Embedding) and COM (Component Object Model) technology and comprises a set standard interfaces, features and methods used by automation clients for process control and manufacturing.



The OLE/COM technologies determine how individual software components work together and exchange data. OPC provides a general interface for communication with various process control devices, independent of the control software used in the process.

The standardized interface enables the user to select any SCADA package that supports OPC or to create his own OPC client based on VC++ or VB.

The following components are available after installing the OPC server:

- OPC server
- OPC configurator
- OPC sample client / OPC diagnostic client

5.1.1 OPC configurator

In the OPC configurator, all control variables are managed in logical groups. These ***Items*** are connected to a PVI object (e.g. a task object).

OPC configurator features:

- User interface for configuration similar to Explorer
- Monitor view for displaying tag data (Tag = data point, consisting of multiple properties: value, quality, etc)
- Tag multiplier for quickly creating multiple tags with the same properties
- Importing Automation Studio export files
- Online import of variables
- Import and export with CSV (Comma Separated Value) file format
- Support of multiple local and remote PVI connections
- Support for logical groups (folders)
- Advanced OPC data quality and data conversion
- Internal simulation possibilities for configuration and test
- Flexible engineering units and signal range

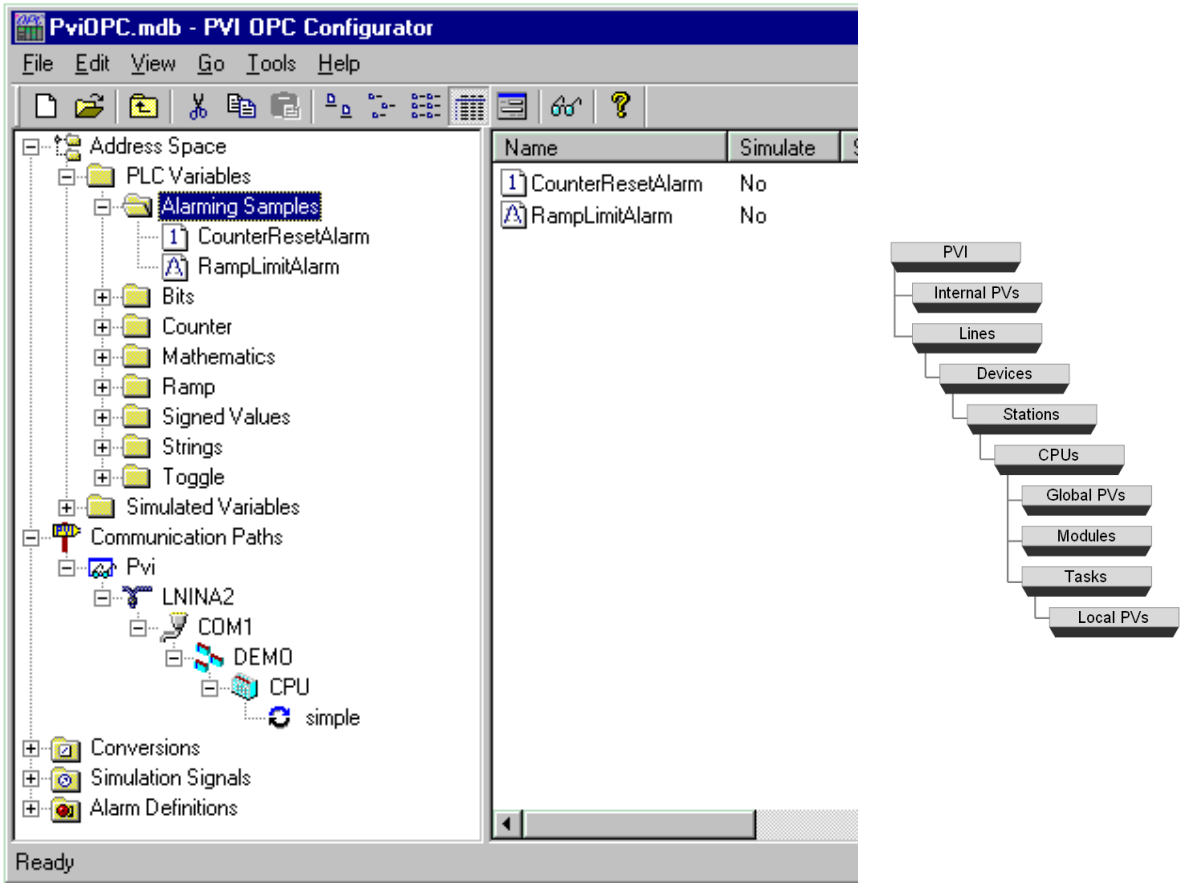


Fig. 34: PVI OPC configurator

5.1.2 OPC server

OPC servers can be addressed using a number of different programming languages, including C++, Visual Basic and script languages.

However, the main area of use for accessing OPC is when using SCADA packages (***S**upervisory **C**ontrol **A**nd **D**ata **A**cquisition*).

The connection between the OPC client and OPC server is established during the project setup. Once this is done, all of the items in the OPC database can be selected and connected to control elements in the visualization.

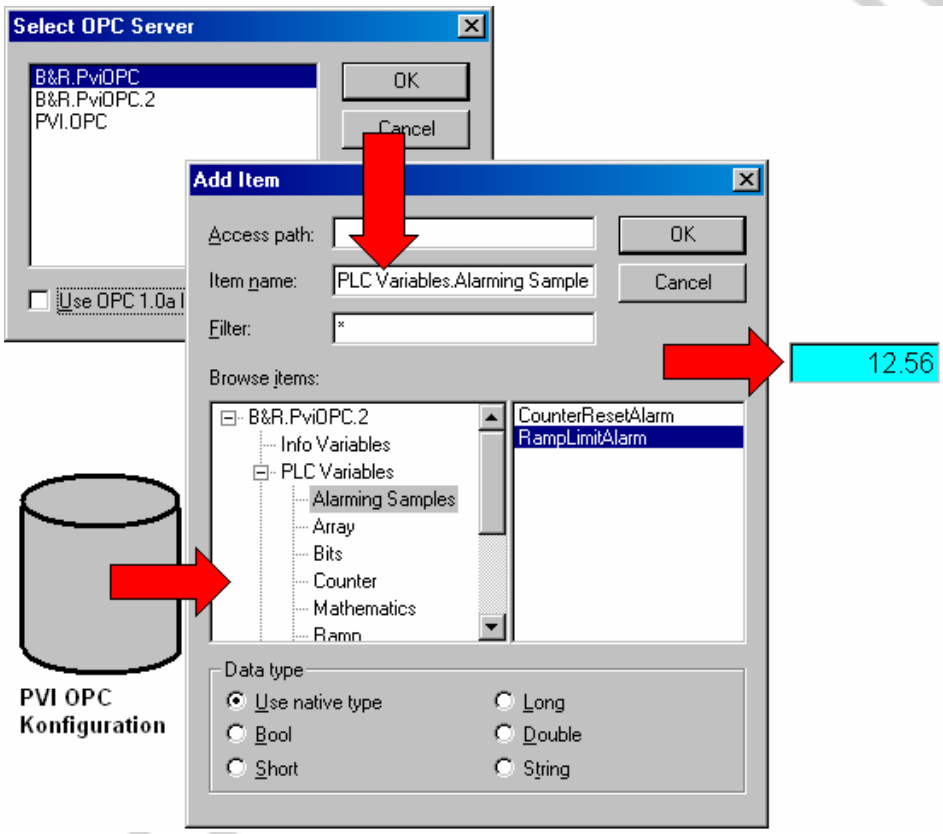


Fig. 35: B&R PVI OPC server

Note:

The PVI OPC server supports OPC Data Access Version 1.0 and 2.0 as well as OPC Alarms & Events Version 1.0 and can be used with DCOM for Intranet and Internet applications.

5.1.3 OPC Sample Client

The OPC Sample Client is used to test the existing OPC configuration.

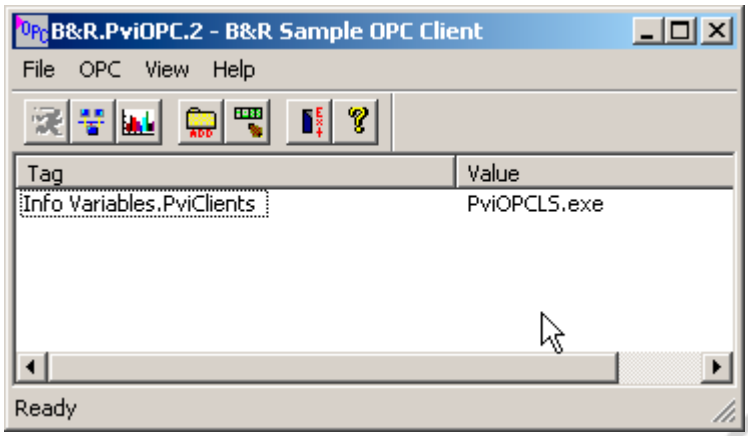


Fig. 36: OPC Sample Client

5.2 PVI DDE server

The PVI DDE server is a PVI application, which allows other Windows applications (clients) such as EXCEL or WORD to access data from PVI variable objects via a standardized interface.

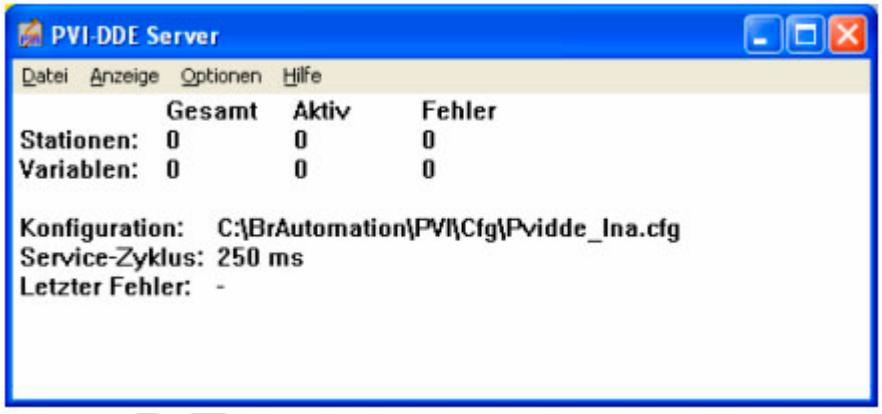


Fig. 37: PVI DDE server

Communication between applications and the DDE Server is made with the help of the DDEML (Dynamic Data Exchange Manager Library) from Windows. According to the definition, the DDE Server is always operated as a server and all applications are operated as clients.

DDE communication is normally initiated by the client which sends a message presently in memory to all DDE Servers (not only PVI DDE Servers).
The ID for the data connection given with the message is determined by the DDE Server. After establishing the data connection, the data provided by the DDE Server (PVI DDE Server process variables) can be read.

The ID for the data connection and the item is made up of three strings, the application, the topic and the item.

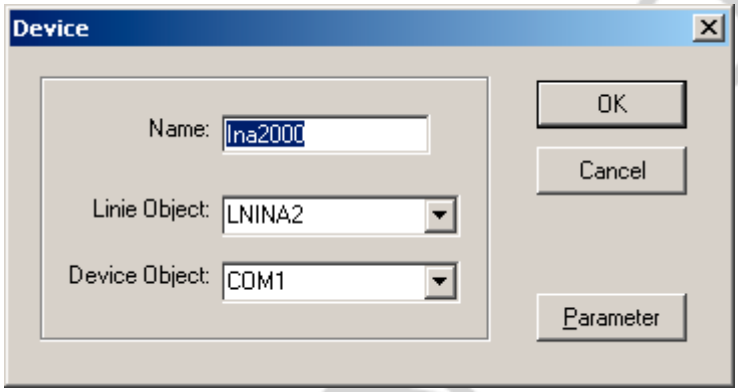
5.2.1 Application, topic and item

Application, topic and item are defined as follows for the PVI DDE Server:

- Application: Server name (PVIDDE or compatible mode NET2DDE).
- Topic: Symbolic station name.
- Item: Symbolic variable name (name of the variable in the DDE Server).

5.2.2 Device list (initialization file: PviDDE.ini)

Each entry in the device list defines a PVI line object and a PVI device object. The device list (selection of the line, device) is defined in the DDE server using a dialog box.



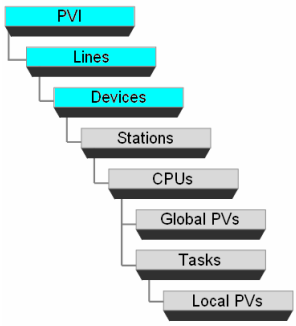
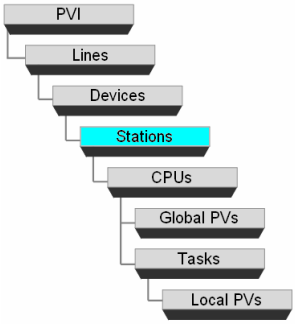


Fig. 38: DDE device configuration

5.2.3 Station List

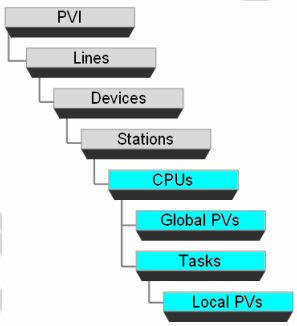
Each entry in the station list defines a PVI station object. Each station entry is also assigned an entry in the device list. In this way, a PVI line object and a PVI device object is also defined.
The station list is defined in the configuration file. The symbolic name allows DDE to address the station entry as a topic.
The station list is located in the configuration file:




```
*******************************************************************************
***                      PVIDDE Server Configuration File                   ***
*******************************************************************************
```

```
TI=250                          * time (ms) for refresh cycle
cpu_INA:#Ina2000/PVITEST{},ERR="E#";   * DDE server station entry for INA2000
```

5.2.4 Variable list



Each entry in the variable list defines a PVI variable object. An optional CPU and task object can be defined. Additionally, each variable entry is assigned an entry in the station list.
The variable list is defined in the configuration file. A symbolic name can be assigned which allows DDE to access the variable entry as an item.

```
Item1:CPU{}/pvitest/Pvar1{PV1},
RF=1,DA=cpu_INA;
Item2:CPU{}/pvitest/Pvar2{PV2}, RF=4,DA=cpu_INA;
Item3:CPU{}/pvitest/Pvar3{PV3}, RF=4,DA=cpu_INA;
Item4:CPU{}/pvitest/Pvar4{PV4}, RF=4,DA=cpu_INA;
```

5.2.5 Displaying a value in EXCEL

After specifying the application (PVIDDE), the topic (Station) and the data (item) in a cell, the value is automatically supplied and displayed.

	A
1	=PVIDDE|cpu_INA!Item1

Fig. 39: DDE displaying values in Excel

5.2.6 Writing data from Excel

The write access from Excel to a control variable is made via a VBA macro:

```
Private Sub cmdWriteItem1_Click()
   channelnumber = Application.DDEInitiate("PVIDDE", "cpu_INA")
   Set SendValue = Worksheets("Table1").Range("A2")
   Application.DDEPoke channelnumber, "Item1", SendValue
   Application.DDETerminate channelnumber
End Sub
```

Note:

The PVI DDE Server can only carry out write and read services for PVI variable objects. Other services or services of other PVI objects cannot be carried out.

5.3 PVI WEB server

The PVI WEB server makes it possible to access a controller's variables from anywhere in the world via HTML pages.

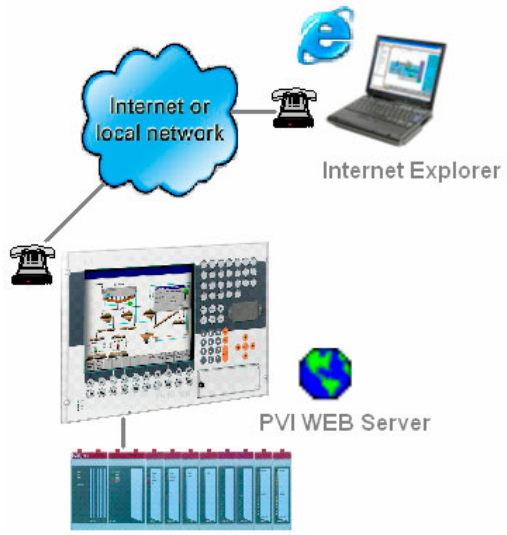


Fig. 40: PVI WEB server

PVI WEB server features and configuration:

- The PVI WEB server reads out the configuration file and registers all of the PVI objects in this list on the PVI manager
- Definition of the start page (HTML page)
- Password protection when writing data points
- Definition of the HTTP port

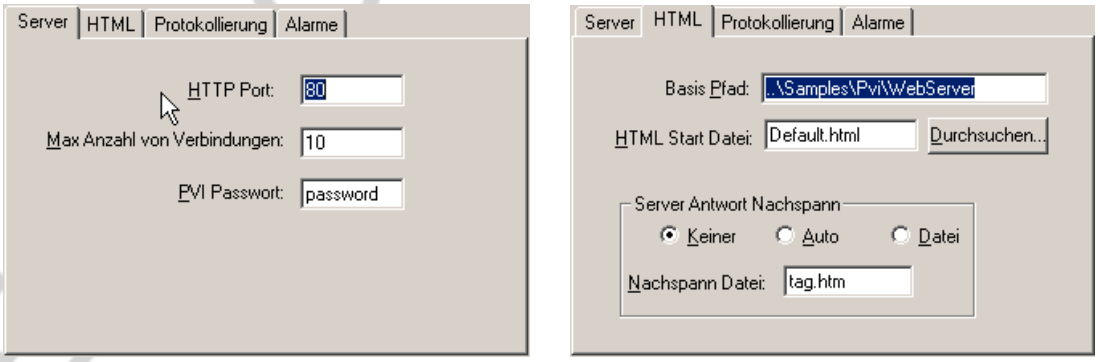
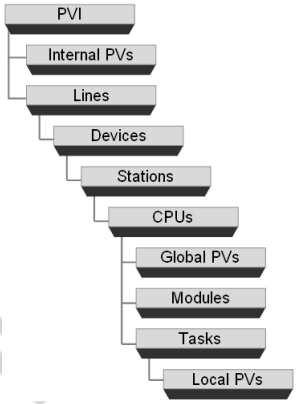


Fig. 41: PVI WEB server configuration

5.3.1 Configuration file:

The PVI objects are defined in a separate configuration file "PviWebServer.cfg", according to the PVI object hierarchy.



[@/Pvi/LNINA2] line
[@/Pvi/LNINA2/COM2] device
[@/Pvi/LNINA2/COM2/PVITEST] station
[@/Pvi/LNINA2/COM2/ PVITEST /CPU] CPU
[@/Pvi/LNINA2/COM2/ PVITEST /CPU /PV1] global variable (or task)

5.3.2 Access to variables on the HTML page

Read and write access to the process variables is possible from an HTML page.

Reading a variable:

```
<PVI FORMAT=%#0.8x>ReadVar @/Pvi/LNINA2/COM2/PVITEST/CPU/PV1</PVI>
```

The following statements must be added to the HTML source in order to read variables cyclically:

```
<META HTTP-EQUIV="REFRESH" CONTENT=5>
```

Time is specified in seconds.

Writing a variable:

```
<FORM action="PviWrite" method="get">
variable name: <INPUT type="text" name="Name" size=32><BR>
new value: <INPUT type="text" name="Value" size=32><BR>
password: <INPUT type="password" name="Password" size=32><BR>
<INPUT type="submit" value=" Write! "><BR>
</FORM>
```

5.3.3 Access to the PVI WEB server

In order to connect a remote or local Web browser to the PVI Web Server, the user has to enter the URL '**http://[Computer name]/**' (e.g. '**http://pc1.company.com/**').

If server and client are located on the same computer '**http://localhost/**' will work as well.

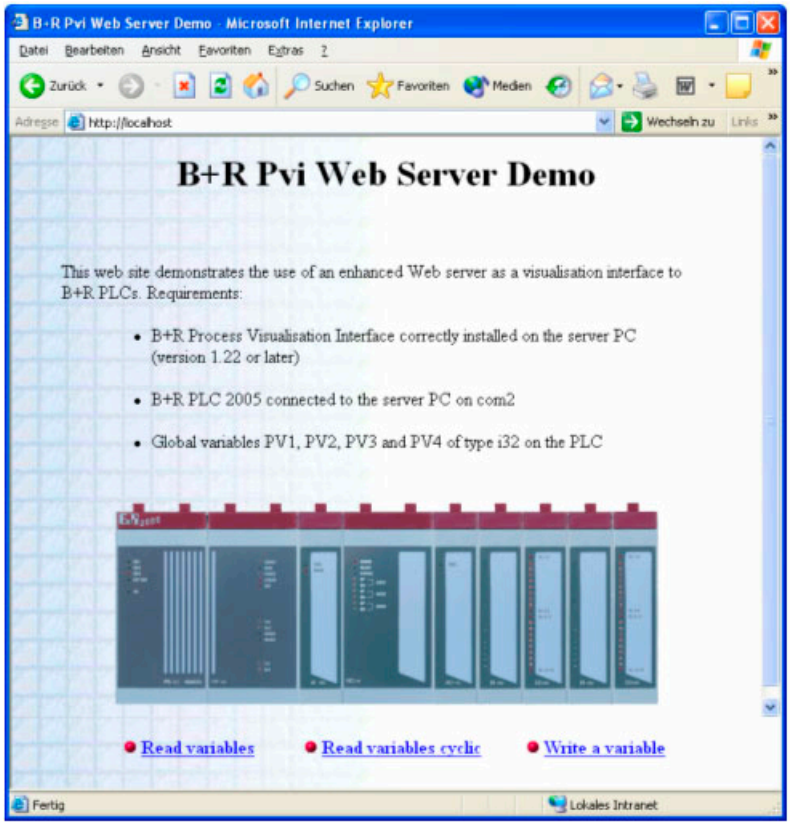


Fig. 42: PVI WEB server demo

6. SUMMARY

Automation Net PVI offers users a wide range of possibilities to manage visualizations in Windows.

The multitude of PVI components allow users to meet their own demands for data exchange between the Windows application and the controller.

These demands are able to be met thanks to the standard interfaces for various PVI servers (OPC, DDE, WEB). However, standard access to the PVICOM interface also provides users with programming functions to implement their own applications.

The PVI concept allows for new PVI components. This ensures that any future developments to the Windows operating system and the new programming techniques that may result (e.g. Visual Studio.NET) will always be covered.

Fig. 43: Automation Net PVI

Notes

Overview of training modules

TM200 – B&R Company Presentation **
TM201 – B&R Product Spectrum **
TM210 – The Basics of Automation Studio
TM211 – Automation Studio Online Communication
TM212 – Automation Target **
TM213 – Automation Runtime
TM220 – The Service Technician on the Job *
TM221 – Automation Components and Sources of Errors *
TM223 – Automation Studio Diagnostics
TM230 – Structured Software Generation
TM240 – Ladder Diagram (LAD)
TM243 – Sequential Function Chart (SFC) *
TM245 – Instruction List (IL) *
TM246 – Structured Text (ST)
TM247 – Automation Basic (AB) *
TM248 – ANSI C
TM250 – Memory Management and Data Storage
TM260 – Automation Studio Libraries I

TM400 – The Basics of Motion Control
TM402 – Dimensioning Motion Control Systems *
TM410 – The Basics of ASiM
TM440 – ASiM Basic Functions
TM441 – ASiM Multi-Axis Functions
TM445 – ACOPOS ACP10 Software
TM450 – ACOPOS Control Concept and Adjustment
TM460 – Starting up Motors *

TM600 – The Basics of Visualization
TM610 – The Basics of ASiV
TM630 – Visualization Programming Guide
TM640 – ASiV Alarm System
TM650 – ASiV Internationalization
TM660 – ASiV Remote
TM670 – ASiV Advanced

TM700 - Automation Net PVI
TM710 - PVI Communication
TM711 - PVI DLL Programming
TM712 - PVIServices
TM730 - PVI OPC

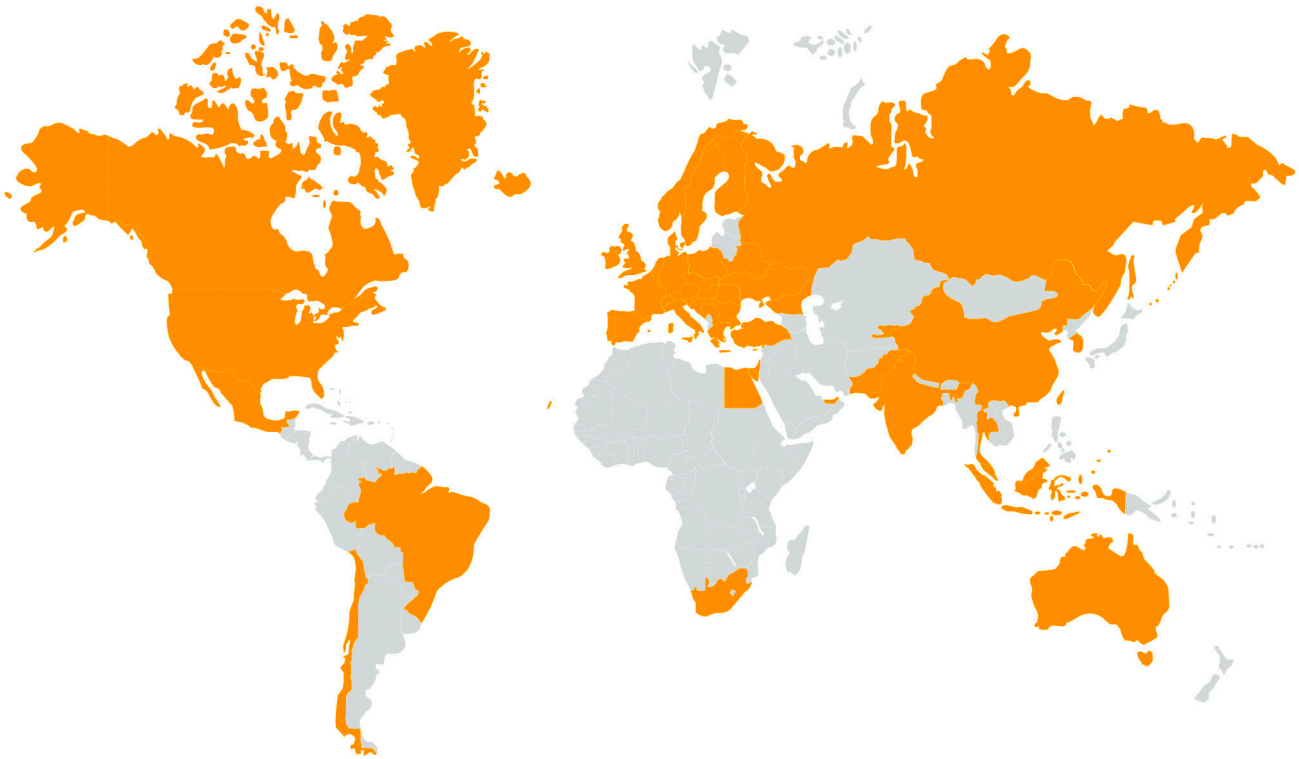
TM800 – APROL System Concept
TM801 – APROL Engineering Basics
TM810 – APROL Setup, Configuration and Recovery
TM811 – APROL Runtime System
TM812 – APROL Operator Management
TM813 – APROL XML Queries and Audit Trail
TM830 – APROL Project Engineering
TM840 – APROL Parameter Management and Recipes
TM850 – APROL Controller Configuration and INA
TM860 – APROL Library Engineering
TM865 – APROL Library Guide Book
TM870 – APROL Python Programming *
TM880 – APROL Report *

*) upon request
**) see Product Catalog

CORPORATE HEADQUARTERS

Bernecker + Rainer Industrie-Elektronik Ges.m.b.H.
B&R Straße 1
A-5142 Eggelsberg
Tel.: +43 (0) 77 48 / 65 86-0
Fax: +43 (0) 77 48 / 65 86-26
info@br-automation.com
www.br-automation.com

120 offices in more than 50 countries - **www.br-automation.com/contact**



Australia · Austria · Belarus · Belgium · Brazil · Bulgaria · Canada · Chile · China · Croatia · Cyprus · Czech Republic
Denmark · Egypt · Emirates · Finland · France · Germany · Greece · Hungary · India · Indonesia · Ireland · Israel · Italy · Korea
Kyrgyzstan · Malaysia · Mexico · The Netherlands · Norway · Pakistan · Poland · Portugal · Romania · Russia · Singapore
Slovakia · Slovenia · South Africa · Spain · Sweden · Switzerland · Taiwan · Thailand · Turkey · Ukraine · United Kingdom · USA