TABLE OF CONTENTS

VERSION 4.0.17 –JANUARY 2009

1. INTRODUCTION

1.1 ABOUT RELIANCE

Reliance is a SCADA/HMI software package designed for the monitoring and control of industrial processes in real time. *Reliance* takes advantage of many years of experience with the development of monitoring and control systems in various fields of industry (e.g. food processing, chemical, heating and the power industry). It allows users to develop complex applications with the ability of connecting to other information systems.

Reliance 4 brings significant improvements implemented according to our users' suggestions and our experience with the development of large applications. Thanks to the modifications made, application development is now much faster and easier than ever before.

1.2 RELIANCE 4 STRUCTURE

Reliance 4 consists of several software modules described briefly in the following chapters.

1.2.1 Development environment: Reliance Design

In the context of *Reliance*, a development environment is a program intended for creating visualization projects. *Reliance Design*, the development environment of *Reliance*, is available in two versions, *DESKTOP* and *ENTERPRISE*.

- *Reliance Design DESKTOP* is a basic version of the program meant for developing applications of type "1 computer - any number of devices". A device may be a PLC (Programmable Logic Controller) or other similar I/O device used for industrial automation and control. The *DESKTOP* version does not enable the user to develop network and Web applications. Thus, the resulting visualization project only allows communications between a single computer and any number of devices.
- *Reliance Design ENTERPRISE* incorporates all the features of the *DESKTOP* version. In addition, it empowers the user to develop network and Web applications for any number of computers with "view-only" and/or "full control" access.

1.2.2 Runtime software: Reliance View, Reliance Control, Reliance Server, Reliance Control Server

In the context of *Reliance*, runtime software is a program designed to run a visualization project on the end user's computer. Runtime software is a common term for the following programs: *Reliance View, Reliance Control, Reliance Server* and *Reliance Control Server*. These programs have an identical core and some basic features – acquisition of data from communication drivers (native drivers, OPC and DDE servers), acquisition of data and alarms/events from other instances of the runtime software (data servers), generating and processing alarms/events, logging data and alarms/events, executing scripts and recipes, and more.

Reliance View, Reliance Control

In addition to the basic features of the runtime software, *Reliance View* and *Reliance Control* allow displaying visualization windows with real-time data,

displaying and acknowledging current alarms/events, displaying and printing historical alarms/events, displaying and printing historical data as historical trends and reports, user administration, and more.

- *Reliance View* is "view-only" software. It only allows the user to monitor the visualized industrial process, but not to control it. It is intended for computers with "view-only" access, e.g. at workplaces of managers who need to view historical trends of process variables (tags) and the current status of the process but have no need to control it.
- *Reliance Control* is full-featured software that allows the user to monitor and control the visualized industrial process. That means that the user can send commands for changing process parameters, set-points or outputs. It is intended for computers with "full control" access. However, "full control" access can be limited on a per-user basis.

Reliance Server

In addition to the basic features of the runtime software, *Reliance Server* is a data server for client instances of the runtime software and *Reliance Web Client*s. It provides clients with data and alarms/events, and executes commands received from the clients. It cannot display visualization windows. Therefore, *Reliance Server* is used as a non-visual data concentrator and data server. It is especially suitable for unmanned computers.

Reliance Control Server

In addition to the basic features of the runtime software, *Reliance Control Server* provides all the features of *Reliance Control* and *Reliance Server*. It is intended for computers with enough performance to simultaneously handle the operator's requests and requests from client instances of the runtime software and Web clients.

1.2.3 Reliance Web Client

In the context of *Reliance*, a Web client is a program designed to run a visualization project over the Web. *Reliance Web Client* is a Java applet/application (i.e. a program written in the Java programming language) intended to run a visualization project inside a Java-enabled Web browser (e.g. Microsoft Internet Explorer 4 and higher) or as a stand-alone application. The Web client provides a powerful and easy way to access the

technology from remote locations over the Internet/intranet. In order to be operational, the Web client needs to be connected to a data server (*Reliance Server* or *Reliance Control Server*). The Web client is a thin client – it requires no installation but has only some of the features of the runtime software. It allows displaying visualization windows with real-time data, controlling the visualized industrial process, displaying and acknowledging current alarms/events, displaying historical alarms/events, displaying historical data as historical trends and reports, displaying custom reports, and more.

1.2.4 Communication drivers

In the context of *Reliance*, a communication driver is a program designed to provide communication with a hardware device. If the device is a PLC or other similar I/O device then the driver provides transmission of process data from the device to the runtime software and transmission of commands (e.g. commands for changing process parameters, set-points or outputs) from the runtime software to the device. Communication drivers for some popular I/O devices (e.g. *Teco*, *Modicon*, *Wago*, *Sauter*, *Rittmeyer* and *Johnson Controls*) are part of *Reliance* (so-called native communication drivers that can only be used with *Reliance*). In addition, *Reliance* can communicate with any device for which an OPC or DDE server is available (*Reliance* is an OPC and DDE client). *Reliance* also supports the GSM technology (sending and receiving SMS messages).

1.3 HARDWARE AND SOFTWARE REQUIREMENTS

Reliance 4 can be operated on computers running MS Windows Vista/XP/2000/NT operating systems. MS Windows 9x/ME operating systems are not supported. The complete installation of *Reliance 4* requires c. 300 MB of free hard disk space.

Reliance Web Client is platform- and Web browser-independent. It means that it may be used on operating systems such as Microsoft Windows, Linux, MacOS, Solaris or FreeBSD and in web browsers such as Microsoft Internet Explorer, Firefox (Mozilla), Opera or Konqueror. The only condition is Java Runtime Enviroment (JRE 6.0) installed on the computer.

2. DESCRIPTION OF THE DEVELOPMENT ENVIRONMENT

The *Reliance Design* development environment is a program intended for creating visualization projects.

The user interface of the program consists of several windows. These are the main window, containing the main menu, toolbars and the component palette, the ***Component Manager***, ***Window Manager***, ***Layer Manager*** and ***Information Window.***

2.1 STARTING THE DEVELOPMENT ENVIRONMENT

The *Reliance Design* development environment can be started from the ***Start*** menu or by using a shortcut on the Windows desktop located in the *Reliance 4* folder.

2.2 DESCRIPTION OF MENU COMMANDS

This chapter describes commands that are available in the main menu. The detailed description of a command may include the default key shortcut assigned to the command. To configure key shortcuts, use the ***Environment Options*** dialog (see the chapter 2.3.3.2 KEY SHORTCUTS).

2.2.1 File menu

The ***File*** menu contains commands for creating, opening, saving and closing visualization projects and visualization windows. The menu also contains a list of most recently open projects.

New Project... Is used to create a new visualization project (see the chapter 2.4.1 CREATING A NEW PROJECT).

Open Project... (*Ctrl+O*) Is used to open an existing visualization project.

Close Project Is used to close the open visualization project.

New Window... Is used to create a new window.

Save Window (*Ctrl+S*) Is used to save the active window.

Save Window As... Is used to save the active window using a new name.

Close Window (*Ctrl+F4*) Is used to close the active window.

Close All Windows Is used to close all open windows.

Exit Is used to exit the program.

2.2.2 Edit menu

The ***Edit*** menu contains a set of commands for editing components in the active window. To use most of the commands, select the component(s) to which a command is to be applied and then select the command from the menu.

Undo (*Ctrl+Z*) Is used to reverse the last action taken. However, some actions cannot be undone.

Redo (*Shift+Ctrl+Z*) Is used to reverse the last *Undo* command.

Cut (*Ctrl+X*) Is used to delete the currently selected component(s) from the window and place it to the clipboard.

Copy (*Ctrl+C*) Is used to copy the currently selected component(s) to the clipboard.

Duplicate (*Ctrl+D*) Is used to duplicate the currently selected component(s). The contents of the clipboard remain unchanged.

Paste (*Ctrl+V*) Is used to paste the contents of the clipboard into the active window. The contents of the clipboard remain unchanged.

Delete (*Del*) Is used to delete the currently selected component(s) from the window.

Select All (*Ctrl+A*) Is used to select all the components in the active window.

Unselect All (*Esc*) Is used to unselect all the currently selected components in the active window.

Next Component (*Tab*) Is used to select the next component in the Z-order.

Previous Component (*Shift+Tab*) Is used to select the previous component in the Z-order.

Group (*Ctrl+G*) Is used to group the currently selected components, i.e. combine them into a single unit called a group. A group maintains a fixed spatial relationship between individual components and cannot be resized.

Ungroup (*Ctrl+U*) Is used to break the currently selected group into individual components.

Alignment

Left Is used to align the left edge of all selected components with the left edge of the leftmost selected component.

Right Is used to align the right edge of all selected components with the right edge of the rightmost selected component.

Top Is used to align the top edge of all selected components with the top edge of the topmost selected component.

Bottom Is used to align the bottom edge of all selected components with the bottom edge of the bottommost selected component.

Center Vertically Is used to align the vertical centerline of all selected components with the centerline of the group of components selected.

Center Horizontally Is used to align the horizontal centerline of all selected components with the centerline of the group of components selected.

Space Equally, Vertically
Is used to equally space all selected components vertically between the topmost selected component and the bottommost selected component.

Space Equally, Horizontally
Is used to equally space all selected components horizontally between the leftmost selected component and the rightmost selected component.

Size

Shrink to Smallest, Horizontally
Is used to adjust the width of all selected components to the smallest selected component.

Grow to Largest, Horizontally
Is used to adjust the width of all selected components to the largest selected component.

Shrink to Smallest, Vertically

Is used to adjust the height of all selected components to the smallest selected component.

Grow to Largest, Vertically

Is used to adjust the height of all selected components to the largest selected component.

Order

Bring to Front (*Shift+PgUp*)

Is used to move the currently selected component(s) in front of all other unselected components in the window. This is called changing the Z-order of a component(s).

Bring Forwards (*Ctrl+PgUp*)

Is used to move the currently selected component forwards by one level. This is called changing the Z-order of a component.

Send to Back (*Shift+PgDn*)

Is used to move the currently selected component(s) behind all other unselected components in the window. This is called changing the Z-order of a component(s).

Send Backwards (*Ctrl+PgDn*)

Is used to move the currently selected component backwards by one level. This is called changing the Z-order of a component.

Transformation... Is used to proportionally resize or (and) move the currently selected component(s).

Align Position to Grid Is used to align the currently selected component(s)'s top left corner to the closest grid point while preserving the component(s)'s size.

Align Size to Grid Is used to align the currently selected component(s)'s bottom right corner to the closest grid point while preserving the component(s)'s top left corner position.

Lock Components (*Ctrl+L*)

Is used to lock the position and size of all the components in the window.

Component Properties... (*Alt+Enter*)

Is used to bring up the property editor of the currently selected component (see the chapter 2.9 COMPONENTS).

2.2.3 View menu

The ▸ ***View*** menu contains commands for showing and hiding windows that belong to the development environment.

Component Manager (*F11*)

Is used to show/hide the ***Component Manager*** (see the chapter 2.6.12 COMPONENT MANAGER).

Window Manager (*F12*)

Is used to show/hide the ***Window Manager*** (see the chapter 2.6.13 WINDOW MANAGER).

Layer Manager Is used to show/hide the ***Layer Manager*** (see the chapter 2.6.14 LAYER MANAGER).

Information Window Is used to show/hide the ***Information Window*** (see the chapter 2.7 INFORMATION WINDOW).

2.2.4 Managers menu

Device Manager... Is used to bring up the ***Device Manager*** (see the chapter 2.6.3 DEVICE MANAGER).

Data Table Manager...

Is used to bring up the ***Data Table Manager*** (see the chapter 2.6.4 DATA TABLE MANAGER).

Trend Manager... Is used to bring up the ***Trend Manager*** (see the chapter 2.6.5 TREND MANAGER).

Real-Time Trend Manager...
Is used to bring up the ***Real-Time Trend Manager*** (see the chapter 2.6.6 REAL-TIME TREND MANAGER).

Report Manager... Is used to bring up the ***Report Manager*** (see the chapter 2.6.7 REPORT MANAGER).

Custom Report Manager...
Is used to bring up the ***Custom Report Manager*** (see the chapter 2.6.8 CUSTOM REPORT MANAGER).

Recipe Manager... Is used to bring up the ***Recipe Manager*** (see the chapter 2.6.9 RECIPE MANAGER).

Picture Manager...
Is used to bring up the ***Picture Manager*** (see the chapter 2.6.10 PICTURE MANAGER).

Script Manager... Is used to bring up the ***Script Manager*** (see the chapter 2.6.11 SCRIPT MANAGER).

Project Structure Manager...
Is used to bring up the ***Project Structure Manager*** (see the chapter 2.6.2 PROJECT STRUCTURE MANAGER).

2.2.5 Project menu

Start (*F9*) Is used to start the visualization project. The development environment runs the runtime software of the specified type and passes the project name and location as parameters.

Create Shortcut... Is used to bring up the ***Create Shortcut to Project*** dialog (see the chapter 2.4.4 CREATING A SHORTCUT TO A PROJECT).

Export for Remote Users...

Is used to bring up the ***Export for Remote Users Wizard***.

Information... Is used to bring up the ***Project Information*** dialog. The dialog contains information on the number of tags and data points within the project (see the chapter 3.3 POINT RATING OF RELIANCE VISUALIZATION PROJECTS).

Options... Is used to bring up the ***Project Options*** dialog (see the chapter 2.4.2 PROJECT OPTIONS).

2.2.6 Tools menu

Configure Toolbars... Is used to bring up the ***Configure Toolbars*** dialog (see the chapter 2.3.1 CONFIGURING THE TOOLBARS).

Configure Component Palette...

Is used to bring up the ***Configure Component Palette*** dialog (see the chapter 2.3.2 CONFIGURING THE COMPONENT PALETTE).

Environment Options...

Is used to bring up the ***Environment Options*** dialog (see the chapter 2.3.3 ENVIRONMENT OPTIONS).

2.2.7 Windows menu

Next (*F6*) Is used to change the active window to the next open window.

Previous (*Shift+F6*)
Is used to change the active window to the previous open window.

Open Window List... (*Alt+0*)
Is used to bring up the ***Open Window List***. Any window in the list can then be selected and activated.

Duplicate... Is used to bring up the ***Duplicate Window*** dialog to duplicate the active window (see the chapter 2.5.2 DUPLICATING A WINDOW).

Properties... (*Alt+Enter*)
Is used to bring up the property editor of the active window (see the chapter 2.5.3 WINDOW PROPERTIES).

2.2.8 Help menu

Contents (*F1*) Is used to access the help system.

Reliance on Internet Is used to open the *Reliance* Web site (www.reliance.cz) where you can find information about *Reliance*, including news and announcements, feature descriptions, and product downloads.

About Reliance 4 Design...
Is used to view the version and license information for the *Reliance Design* program.

2.3 SETTING UP THE DEVELOPMENT ENVIRONMENT

2.3.1 Configuring the toolbars

Toolbars

The main window of the development environment contains 5 toolbars.

Component palette	Contains icons representing the components you can use to design your application interface. To rearrange the icons on the palette, use the ***Configure Component Palette*** dialog (see the chapter 2.3.2 CONFIGURING THE COMPONENT PALETTE).
Standard	Is designed for standard file operation commands (opening, closing, saving, etc.).
Edit	Is designed for commands used when working with components.
Managers	Is designed for commands for opening individual managers.
Project	Is designed for project operation commands.

Commands

A toolbar can be configured by dragging any command available in the command list to the toolbar area with the mouse. Commands are well-arranged in groups according to their function. However, their position on individual toolbars is fully configurable.

Options

Show tooltips	Determines whether to show a brief description of the command after positioning the mouse cursor over a toolbar button.
Show key shortcuts on tooltips	Determines whether to show key shortcuts on tooltips.

2.3.2 Configuring the component palette

The component palette contains icons representing the components you can use to design your application interface. Within the palette, the icons are divided into individual pages. By default, the palette contains the pages *Standard*, *Vectors*, *Control*, *Teco*, *Sauter*, *Johnson Controls*, *IP Cameras* and *Other*. Individual pages can be renamed and if needed, a new page may be added or an existing one removed. Installed components can be moved by dragging with the mouse into the page selected.

Add	Is used to add a new page to the component palette, or install a new component from the »Components« directory.
Rename	Is used to rename the selected page of the component palette.
Show	Is used to show or hide the selected component.
Delete	Is used to remove the selected page from the component palette, or remove the selected component from the selected page.
Move Up	Is used to move the selected page or selected component up by one position.
Move Down	Is used to move the selected page or selected component down by one position.

2.3.3 Environment Options

The ***Environment Options*** dialog enables you to define paths, key shortcuts and other settings of the development environment.

2.3.3.1 Paths

Project directory	Specifies the path for storing visualization projects.
Project backup directory	Specifies the path for storing project backups.

Picture Library directory
: Specifies the path where the picture library is located.

2.3.3.2 Key shortcuts

Here it is possible to configure key shortcuts for all the commands available in the main menu.

2.3.3.3 Picture Manager

Editors

Here it is possible to associate the picture formats supported by *Reliance* with external picture editors. Thus, you can view or edit a picture with the specified editor directly from the ***Picture Manager*** by invoking the *Edit Picture* command. After closing the editor, you return to the ***Picture Manager***. By default, all the picture formats are associated with Microsoft Paint (mspaint.exe).

Editor
: Specifies the path and name of the picture editor's executable file.

Parameters
: Specifies the parameters to pass to the picture editor at startup. By default, *Reliance* passes the picture name as a parameter to the editor.

Working directory
: Specifies the path for storing temporary files.

Thumbnails

Here you can configure the appearance of picture thumbnails displayed in the ***Picture Manager*** if the *Thumbnails* display style is used. You can specify the width and height of the thumbnails. The thumbnails can also display the picture size and file size.

2.3.3.4 Script Manager

General

Set cursor to beginning of edited script

Determines whether to set the cursor to the beginning of a script's code after opening the script in the editor.

Insert header into new script

Determines whether to insert a header to the beginning of a newly created script's code. The header is a group of comment lines that contain basic information such as project name, user name logged on to Windows, date and time of script creation.

Code Insight

Automatic functions Enables you to configure features that make code writing easier.

Code completion Determines whether the code editor should help you complete the code you type in the editor. If this option is active, the editor automatically displays the list of properties and methods appropriate to any *Reliance*-defined object after entering the object's name followed by a period. You can then select the item from the list and press *Enter* to add it to your script. However, you can always invoke the list of properties and methods by pressing *Ctrl+Spacebar* (if the cursor is positioned right after the period following the object's name), even if this option is not active.

Function parameters Determines whether the code editor should give you help on parameters of functions used in the code you type in the editor. If this option is active, the editor automatically displays the list of parameters (in a pop-up window) appropriate to any predefined

function after entering the function's name followed by a left parenthesis. However, you can always invoke the list of parameters by pressing *Ctrl+Shift+Spacebar* (if the cursor is positioned after the left parenthesis following the function's name), even if this option is not active.

Reliance-defined objects

Automatically supply function and procedure parameters
Determines whether the code editor should automatically supply parameters of *Reliance*-defined objects' methods (i.e. procedures and functions). If this option is active and you select a method (by double-clicking it) from the list of methods of a *Reliance*-defined object, the editor lets you select the parameters (e.g. the window to be activated) from the appropriate dialogs and adds them to your script.

2.3.3.5 Visualization windows

Grid and bounds

Size X Specifies the grid width.

Size Y Specifies the grid height.

Display grid in newly created windows
Determines whether to activate the *Display grid* property in newly created windows.

Snap to grid in newly created windows
Determines whether to activate the *Snap to grid* property in newly created windows.

Display bounds in newly created windows
Determines whether to activate the *Display window bounds* property in newly created windows.

Undo and Redo

History step count Specifies the number of edit operations that can be cancelled (adding and deleting components and modification of their size, position and Z-order).

Undo after save changes

Determines whether to enable the user to use the *Undo* command after saving the window.

Other

Delete XML file on remove window

Determines whether to delete a window's *XML* file after removing the window from the project.

2.3.3.6 File types

File types registration

Here, it is possible to associate files with an .rp4 extension with the *Reliance Design* development environment.

Upon registration, the files with an .rp4 extension are marked as *Reliance 4 Project* and they are assigned a specific icon. Thus, for example, a visualization project can be opened by double-clicking on the shortcut to the main file on the Windows desktop.

The registration is done automatically by the installer.

2.4 VISUALIZATION PROJECT

A *Reliance* visualization project is a group of files stored in a specific directory structure (see the chapter 3.2 FILE AND DIRECTORY STRUCTURE).

2.4.1 Creating a new project

To create a new visualization project, choose the ▸ ***File*** ▸ ***New Project*** command. This will bring up the ***Create New Project Wizard*** where you can specify the name, location and other options for the new project.

After the new project is generated, the ***Create New Window Wizard*** appears to prompt you to specify some basic properties for the first visualization window (see the chapter 2.5.1 CREATING A NEW WINDOW).

2.4.2 Project Options

2.4.2.1 Alarms/Events

Database

Database type — Determines whether to use a file-based or SQL-based database.

Archive files/Deleting records — Determines whether to create archive files or maintain the latest alarms/events in the database's current file.

Archive files — Determines the type of archive files. Creating archive files can be either periodic or tag-controlled.

Delete oldest archive files — Determines whether to delete the oldest archive files after exceeding the specified count.

Limit accessible archive file count — Determines whether and how to limit the number of archive files accessible to the user during runtime.

Sounds

These options enable you to configure sounds that can be played when an alarm/event is generated, ends or is active. The sound files must be located in the »Main\MMedia« subdirectory of the visualization project.

Default sounds Specifies the default sounds in the **.wav* format. The default sounds will be used for alarms/events that do not specify a particular sound.

Active alarm/event sound

Determines whether and how often the runtime software should play the specified sound if the list of current alarms/events contains at least one active unacknowledged alarm/event.

Display

Alarm/Event text font Determines the font of alarm/event texts at runtime.

Other

Allow users to disable alarms/events

Determines whether to allow users to temporarily disable and enable alarms/events at runtime. If active, you can specify the access rights required for these operations. This feature can be useful for example in case of a long-term failure of the equipment to prevent alarms/events from being triggered.

2.4.2.2 Runtime

Basic

Starting

Runtime software — Determines the type of runtime software (*Reliance View*, *Reliance Control* or *Reliance Control Server*) to run the visualization project when started from the development environment.

Computer name — Determines the logical computer (defined through the ***Project Structure Manager***) whose settings (e.g. IP address, host name, connected devices, etc.) will be used by the runtime software when starting the visualization project from the development environment.

Control termination — Determines whether the visualization project can be terminated owing to changing the value of the specified tag at runtime. If the control tag's value changes to the specified value, the runtime software displays an information message regarding the termination. After expiration of the specified delay, the visualization project is terminated.

Appearance

Main window background

Determines the appearance of the runtime software's main window.

Language

This option determines the language to be used for all texts in the runtime software's user interface (i.e. menus, toolbars, etc.).

2.4.2.3 Access Rights

In a *Reliance* visualization project, there are 30 access rights available. These rights (named *Right1* to *Right30* by default) can be used to control the access to specific operations.

There is no hierarchy among the access rights – i.e. their sequence within the list has no meaning (Example: If *Right1* or *Right2* is required for setting the value of a tag using the Display component, the user with the *Right3* access right will not be allowed to perform the operation).

It is strongly recommended to rename the access rights on the functional basis.

2.4.2.4 Window records

At runtime, it is possible to make operating records for each visualization window. When activating a window (for which a new record has been made), the new record is signalized by changing an icon on the standard toolbar. This feature is useful for example in productions with a shift operation. Using this feature, the operators may pass on information in record format to one another. For example, if a failure occurs in the technology, a record can be made for the window related to the technology.

Enable window records Determines whether to allow users to access window records at runtime.

Access rights required for record deletion
Specifies the access rights required for deleting an existing window record. To perform other operations on window records (e.g. view or write records), no specific access rights are required. However, the operator must log on to the program.

2.4.2.5 Telephone service providers

Here you can define providers of telephone services by renaming the predefined items. For example, rename *Provider1* to *Vodafone*, *Provider2* to *T-Mobile*, etc. Later, you can choose a provider when configuring modems and communication channels of I/O devices connected to a computer.

2.4.2.6 DDE sharing

The runtime software acts as a DDE server (see the chapter 2.6.3.3 DEFINING A TAG).

Replace invalid values of tags

Determines whether and how to replace the value of a DDE-shared tag when providing it to DDE client programs in case that the value is not valid (e.g. because of a failure in communication with the device).

Array element divider Specifies the text to be used as element divider when sharing an array-type tag.

2.4.2.7 Web

These options enable you to configure the communication between a data server and thin clients (*Reliance Web Client, Reliance Mobile Client*). Since only *Reliance Server* and *Reliance Control Server* can be used as a data server for thin clients, the options have no effect if *Reliance View* or *Reliance Control* is used as the runtime software.

Start Web server and service for thin clients

If this option is active, the runtime software starts to act as a data server for thin clients (i.e. Web and Mobile clients) and enables them to connect to it using the TCP/IP protocol.

Port number

Specifies the TCP/IP port number to be used by thin clients when connecting to a data server. At

the same time, a data server uses the specified port number to open a TCP/IP socket in order for thin clients to connect to it. It is recommended to use the default value.

2.4.2.8 Scripts

Run script

After start project Specifies the script to be run as the first script after starting the visualization project.

Debugging scripts

Terminate script on error Determines whether to terminate a script if an error occurs while working with the *Reliance*-defined objects in the script.

☞ **RECOMMENDATION** It is recommended to activate this option for debugging purposes only. It only relates to the *Reliance*-defined objects (e.g. *RTag*, *RSys*, etc.). It does not apply to other objects or syntax errors of *VBScript*.

2.4.2.8 Components

Display

Mark components with invalid links to tags
Determines whether to graphically mark the components with invalid links to tags (e.g. link to a tag that was later deleted through the ***Device Manager***). If the option is active, such components will be marked with a red border.

Mark components linked to tags with invalid values
Determines whether to graphically mark the components linked to tags with invalid values

(e.g. link to a tag whose value has not yet been returned by the communication driver). If the option is active, such components will be marked with a yellow border.

2.4.2.10 Objects

Complete object names

Character to be used to separate name parts when building complete name of an object (e.g. tag)

Determines the character used when building the complete name of an object (e.g. tag) to separate the two parts of the name. The default separator is the dot character.

🗝 **EXAMPLE** If the default separator (the dot character) is used, the complete name of the tag *Temperature1* belonging to the device *Tecomat1* is displayed as *Tecomat1.Temperature1*.

👁 **WARNING** This character must not be included in the name of any object within a *Reliance* visualization project!

2.4.4 Creating a shortcut to a project

To create a shortcut to the open project, choose the ▸ ***Project*** ▸ ***Create Shortcut...*** command. This command brings up the ***Create Shortcut to Project*** dialog box.

Software — Determines the program to open/run the project.

Computer to run project on — Determines the logical computer (defined through the ***Project Structure Manager***) whose settings (e.g. IP address, host name, connected devices, etc.) will be used by the runtime software when starting the visualization project through the shortcut. This option is only used when the *Software* option specifies the runtime software

Shortcut name — Determines the name of the shortcut.

Shortcut location — Determines the folder to place the shortcut in. The default shortcut location is the »Desktop« folder of the user currently logged on to Windows.

Comment — Can be used to specify an optional description of the shortcut.

2.5 VISUALIZATION WINDOW

A visualization project usually contains one or more visualization windows. A visualization window is intended to contain components from the component palette.

2.5.1 Creating a new window

To create a new window, choose the ▸ ***File*** ▸ ***New Window*** command. This command brings up the ***Create New Window Wizard*** which prompts you to specify the following properties for the new window: *Name*, *Title*, *Window type*, *Dynamic loading*. For the description of window properties, see the chapter 2.5.3 WINDOW PROPERTIES.

2.5.2 Duplicating a window

To duplicate the active window, choose the ▸ ***Window*** ▸ ***Duplicate...*** command. This command brings up the ***Duplicate Window*** dialog box.

Name — Specifies the window's name that must be unique within the project and cannot contain illegal characters. Using the window name, you can work with the window for example in scripts. The window name is also displayed in controls containing a link to the window.

Title — Specifies the text to be displayed in the window's title bar (see the chapter 2.5.3 WINDOW PROPERTIES).

Substitute components' links to tags — Determines whether to substitute components' links to tags while duplicating the window. If the option is active, all links to tags from the device specified by the *Source device* option will be replaced by links to same-named tags from the device specified by the *Target device* option.

2.5.3 Window properties

To view or edit the properties of a visualization window, choose the ▸ ***Window Properties...*** command from the popup menu of the window or double-click the window. This will bring up the ***Window Properties*** dialog box.

ⓘ **NOTE** Configuring certain properties will not take effect in the development environment, but at runtime only.

Basic

Name Specifies the window's name that must be unique within the project and cannot contain illegal characters. Using the window name, you can work with the window for example in scripts. The window name is also displayed in controls containing a link to the window.

Alias Specifies an alternative name for the window. In multi-language projects, alias can be localized (i.e. translated into all project languages).

Title Specifies the text to be displayed in the window's title bar.

Window type

Unlike standard windows, dialog windows have a fixed size and always display a title bar. These properties are optional for standard windows. Also, dialog windows can be closed by the user at runtime. A tray (top, bottom, left or right) is a special type of window that is aligned to the top, bottom, left, or right of the runtime software's main window.

Options

Show title bar Determines whether to display the window's title bar at runtime.

Dynamic loading Determines whether to load the window into memory only before it is activated. If the window becomes completely obscured by other windows it is released from memory. If

this property is not active, the window is loaded into memory right after starting the project and it stays in memory until the project is terminated. It is recommended to leave this property active for most windows.

Enable sizing — Determines whether to enable users to resize the window at runtime.

Stay on top — Determines whether the window should stay on top of the other open windows at runtime – except for windows with the identical feature and trays.

Exclusive mode — Determines whether the window should be displayed in exclusive mode (modally). When a window is displayed in exclusive mode, the user is not allowed to switch to another visualization window or to perform other operations within the runtime software until the window is closed.

Placement

Specifies how the window is to be displayed at runtime.

Maximized — The window fills the area of the runtime software's main window.

Centered — The window has the same size as at design-time and is positioned in the center of the runtime software's main window.

In specified position — The window's position is defined by the specified co-ordinates (the *Startup position and size* or *Position and size* property) of the window's top left corner.

Position

Position and size Determines the window's current position and size (in pixels).

Startup position and size Determines the window's initial position and size (in pixels) to be used at project startup. If *Startup position and size* is not active, *Position and size* is used instead.

Background

Background color Specifies the window's background color.

Grid

- *Show grid* Determines whether to display dots on the window to make the grid visible.
- *Snap to grid* Determines whether to align components in the window to the closest grid point.

Picture Specifies the picture to be displayed on the window's background.

- *Placement* Specifies how the background picture is to be displayed.
  - *Tiled* The picture is drawn as tiles to fill up the entire window area.
  - *Centered* The picture is positioned in the center of the window.
  - *In specified position* The picture's position is defined by the specified co-ordinates (the *Position* property) of the picture's top left corner.
- *Adjust size to picture* Determines whether to adjust the size of the window to the picture.

Scripts – Window

These properties enable you to specify the scripts to be executed after the window is loaded into memory, activated, deactivated, closed and freed. You can also specify the parameters to pass to the scripts.

Scripts – Mouse

These properties enable you to specify the scripts to be executed after clicking or double-clicking individual mouse buttons on the window area. You can also specify the parameters to pass to the scripts.

Actions – Mouse

These properties enable you to specify the actions to be performed after clicking or double-clicking individual mouse buttons on the window area.

Advanced

Show hint	Determines whether to display the specified help hint when the mouse cursor rests momentarily on the window area.
Locked	Determines whether the position and size of all the components in the window are locked.

Security

Access to window	Specifies the access rights required for accessing the window (i.e. activating the window).

2.5.4 Designing a window

To create a new window, choose the ▸ ***File*** ▸ ***New Window*** command. Once created, the new window becomes active and is displayed on top of other open windows.

When designing a window, add individual components from the component palette to the window. If a window is to contain a large number of components that overlap each other, it is recommended to use the layer system. Each component can be located on one of 16 layers. A layer may be hidden (all components located on this layer will be hidden at design-time) or locked (all components located on this layer will be locked at design-time) as needed.

A component selected on the palette by clicking the left mouse button may be placed into a window either with its default size (by clicking on the window area), or with the size defined by dragging the mouse inside the window area (before releasing the mouse button).

☺ **HINT** To add multiple components of the same type, press the *Shift* key while selecting the component on the palette.

To change the position or size of components or component groups, edit functions may also be used – e.g. centering, alignment, changing the Z-order, modification of height and width, transformation, etc. The *Undo* and *Redo* commands allow you to undo and redo up to the last 100 edit actions.

The last step when designing a window is configuring the properties of the components, which affects their appearance and behavior. To configure the properties of a single component, use the component's property editor (double-click the component or choose the ▸ ***Component Properties...*** command from the component's popup menu). The properties of individual components differ depending on their type. For more information on the available components, see the chapter 2.9 COMPONENTS.

☺ **HINT** To configure the properties of multiple components at the same time, select the components and edit their properties through the ***Component Manager***.

2.6 MANAGERS

In the context of *Reliance*, managers are tools designed to define and configure objects that are the building blocks of every *Reliance* visualization project. These are computers, users, devices, data tables, trends, reports, custom reports, recipes, scripts, windows, components and others.

Most of the managers have a uniform appearance and behavior.

Toolbar

Most of the commands on the toolbar are common to all managers (see the chapter 2.6.1 DESCRIPTION OF COMMON COMMANDS).

Top left pane

The top left pane displays the objects in a hierarchical tree diagram in order to show the subordination of one object to another. The diagram allows you to select and edit multiple objects at a time (editing is allowed only if the selected objects are of the same type, e.g. alarm/event). When you select an object in the tree diagram, the manager displays the subordinated objects in the bottom left pane and the object's properties in the right pane. If multiple objects are selected in the tree diagram, the manager displays the subordinated objects and properties of the selected object that has the input focus or the first object in the selection (if no object in the selection is focused).

Bottom left pane

The bottom left pane displays the objects in a list. These are the objects subordinated to the object selected in the top left pane. The list allows you to select and edit multiple objects at a time. When you select an object in the list, the manager displays the object's properties in the right pane. If multiple objects are selected in the list, the manager displays the properties of the selected object that has the input focus or the first object in the selection (if no object in the selection is focused).

Right pane

The right pane displays properties of the object(s) selected in the top left or bottom left pane. The properties can be edited as needed. When you edit a property, the corresponding control changes its color to yellow. This status

is also indicated by a red exclamation mark displayed beside the edited object(s) in the top left and bottom left pane. When the selection is about to change, the manager checks all the edited properties to see if they are correct. If so, the manager assigns the edited properties to the selected object(s) and the red exclamation mark changes its color to blue. Otherwise, the selection remains unchanged. If the edits have been assigned to the selected object(s), they can later be saved to the appropriate files by the *Save All* command or canceled by invoking the *Close* command and choosing not to save the changes. Naturally, you can also edit a property and use the *Save All* command immediately. The manager checks all the edited properties to see if they are correct. If so, the manager assigns the edited properties to the selected object(s) and saves the object(s) to the appropriate files.

2.6.1 Description of common commands

New Folder (*Alt+Ins*) Is used to create a new folder. The type of the newly created folder depends on the object selected when invoking the command.

New Object (*Ins*) Is used to create a new object. The type of the newly created object depends on the object selected when invoking the command.

Undo (*Ctrl+Z*)
Is used to cancel the edits that have not yet been assigned to the selected object(s).

Copy (*Ctrl+C*) Is used to copy the currently selected object(s) to the clipboard.

Cut (*Ctrl+X*) Is used to delete the currently selected object(s) from the structure and place it to the clipboard.

Paste (*Ctrl+V*) Is used to paste the contents of the clipboard into the structure. The contents of the clipboard remain unchanged.

Duplicate (*Ctrl+D*) Is used to duplicate the currently selected object(s). The contents of the clipboard remain unchanged.

Delete (*Del*) Is used to delete the currently selected object(s) from the structure.

View	Is used to change the view style of the objects listed in the bottom left pane of the manager.
Sort	Is used to sort the objects directly subordinated to the object selected in the tree diagram by the selected column of the bottom left pane.
One Level Up (BkSp)	Is used to move the selection one level up in the tree diagram.
Find Object (*Ctrl+F*)	Is used to bring up the ***Find Object*** dialog to search for an object by its name.
Options	Is used to bring up the ***Options*** dialog to view or configure the settings related to the manager.

2.6.2 Project Structure Manager

The ***Project Structure Manager*** is a tool designed for defining the structure of an entire visualization project. It enables you to define the structure consisting of control areas, computers, server connections, devices, data tables, users and other objects, so that it corresponds to a real plant site.

A visualization project always comprises at least one control area. A control area comprises at least one computer. The objects defined through other managers, such as devices, data tables, trends and reports, can be made accessible to a computer by connecting them to the computer, i.e. adding links to the objects to the computer's appropriate folder. To easily understand the architecture of a visualization project, see the examples at the end of this chapter.

2.6.2.1 Defining control areas

In the context of *Reliance*, a control area is an independent unit representing a separate locality, where one or more computers designed for running the visualization project are located. Computers can be defined within a control area. A typical example of a control area is a control room.

Name Specifies the control area's name that must be unique within the project and cannot contain illegal characters.

2.6.2.2 Defining computers

In the context of *Reliance*, a computer, also referred to as logical computer, is an object representing the actual computer on which the visualization project will be running at the end user site.

Basic

Name Specifies the computer's name that must be unique within the project and cannot contain illegal characters.

Identification on network

Full name — Specifies the name of the actual computer within the computer network (also known as hostname).

IP address — Specifies the IP address of the actual computer within the computer network. If the actual computer has a DHCP-assigned IP address, leave the property blank and specify *Full name*.

☞ **RECOMMENDATION** It is recommended to specify the *IP address* property. However, if the actual computer has a DHCP-assigned IP address, the *Full name* property will be used instead. If both of these properties are specified, the *IP address* property will be used in preference to the *Full name* property.

Hardware configuration — Can be used to specify the computer's hardware configuration.

Display

Windows

Initial window — Specifies the visualization window to appear as the first window after starting the visualization project on the computer.

Disabled windows — Specifies a list of visualization windows that should not be available to the user at runtime.

Show event log while starting/terminating project

Determines whether to show the event log that contains a description of the operations being performed while starting and terminating the project.

Show tool bar Determines whether to display the standard toolbar in the runtime software's main window. You can hide the standard toolbar and create your own one.

Show main menu bar Determines whether to display the menu bar in the runtime software's main window.

Show title bar Determines whether to display the title bar in the runtime software's main window.

2 monitors Allows you to configure the behavior of the runtime software when operating on a multi-monitor system.

Alarms/Events

Alarm/event database

Directory Specifies the path where the alarm/event database will be located. If the path is not specified, the „History\AlarmsEvents« subdirectory of the visualization project will be used.

Create archive files Determines whether the runtime software running on the computer should create archive files for the alarm/event database.

Current alarms/events

Show current alarms/events on start alarm/event

Determines whether the list of current alarms/events should be shown when an alarm/event is generated that has the *Show current alarms/events* property configured as active.

Show alarm/event panel

Determines whether to show a special panel (alarm/event panel) in the bottom part of the

runtime software's main window and display alarms/events in the panel.

Auto-hide panel after all alarms/events acknowledged

Determines whether to automatically hide the alarm/event panel after acknowledging all current alarms/events.

Show device name

Determines whether the alarm/event panel should show the name of the device to which a currently displayed alarm/event belongs.

Log-on/Log-off

On log-on user

Run script Specifies the script to be executed after the user logs on.

Play sound Specifies the sound to be played after the user logs on. The sound file (a file in the **.wav* format) must be located in the „Main\MMedia« subdirectory of the visualization project.

On log-off user

Run script Specifies the script to be executed after the user logs off.

Play sound Specifies the sound to be played after the user logs off. The sound file (a file in the **.wav* format) must be located in the „Main\MMedia« subdirectory of the visualization project.

Log-on by HW code sensor

Determines whether a hardware code sensor will be connected to the computer to enable users to log on to the program using a special card with a unique code.

Log-on by a biometric sensor

Determines whether a biometric sensor (e.g. a fingerprint reader) will be connected to the computer to verify users when logging on to the program.

Restrictions

These properties determine the access restrictions applied when no user is logged on to the program.

Disable 'Start' menu Determines whether to disable using the Windows ***Start*** menu.

Hide task bar

Determines whether to hide the Windows task bar.

Hide icons on desktop Determines whether to hide icons on the Windows desktop.

Disable minimizing runtime software's main window

Determines whether to disable minimizing the main window of the runtime software.

Disable moving runtime software's main window

Determines whether to disable moving the main window of the runtime software.

Disable printing alarms/events, trends, reports...

Determines whether to disable printing alarms/events, trends and reports. However, this property only applies to printing invoked by the user. It does not affect online printing alarms/events.

Disable editing trends Determines whether to disable editing trends via the ***Trend Manager***.

Disable editing reports Determines whether to disable editing reports via the ***Report Manager***.

Project termination

Access rights Determines the access rights required for terminating the visualization project at runtime.

E-mail

These properties enable you to configure the settings related to sending e-mail messages by the runtime software.

Outgoing E-mail configuration

SMTP server (name or address) Specifies the name or IP address of the SMTP server to be used for sending e-mail messages.

Port number Specifies the port used by the SMTP server for sending e-mail messages.

Connection timeout (ms) Specifies the maximum time period for the runtime software to connect to the SMTP server.

Sender address Specifies the e-mail address of the sender.

Sender name Specifies the name of the sender.

SMS

These properties allow you to configure the settings related to sending and receiving SMS messages by the runtime software.

Start SMS driver Determines whether to launch the SMS driver when starting the visualization project.

GSM device type Specifies the type of GSM device to be used for sending and receiving SMS messages.

Communication options
Enables you to configure the connection to the computer using the RS-232 interface (serial port number, communication speed in bauds, number of data and stop bits, parity).

SMS service center number
Specifies the telephone number of the SMS service center depending on the provider (T-Mobile, etc.).

Run script on receive SMS
Specifies the script to be executed each time the GSM device receives a SMS. The received SMS is passed to the script as a parameter.

For more information see the Script help. Sending and receiving SMS using *Reliance* is demonstrated in the example located in the »Examples« directory on your hard disk.

Postmort

These properties allow you to configure the Postmort function. If the function is activated, the runtime software records changes in process data of the controlled process on a real-time basis into special data files. Later, the operator can switch from online mode to Postmort mode and replay the process. Thus, for example, it is possible to analyze the cause of a technology failure.

It is highly recommended to use different computers for storing and viewing Postmort records, because both cannot be done at the same time.

Record postmort Determines whether to activate recording the process.

Max. record length (day count)
Specifies the maximum record length, i.e. number of days recorded.

Directory Specifies the path where the files containing Postmort records will be located. If the path is not specified, the »History\Postmort« subdirectory of the visualization project will be used.

Miscellaneous

User profiles

Directory Specifies the path where user profiles will be located. If the path is not specified, the »Settings\Profiles« subdirectory of the visualization project will be used.

Window records

Directory Specifies the path where the window record database will be located (see the chapter 2.4.2.4 WINDOW RECORDS). If the path is not specified, the »History\WindowRecords« subdirectory of the visualization project will be used.

2.6.2.4 Connecting devices

To make a device (e.g. PLC, previously defined via the ***Device Manager***) accessible to the runtime software running on an actual computer, you have to add a link to the device object to the logical computer's *Devices* folder. This is called connecting a device to a computer. After adding a link to the device object to the *Devices* folder, a new object of type communication channel is automatically added under the link.

Basic

Enable — Determines whether to make the device accessible to the computer. If the property is not active, the runtime software acts as if the device were not connected to the computer (i.e. as if the link to the device were not contained in the *Devices* folder) and ignores this device and its subordinated objects (tags, communication zones and alarms/events) when loading the visualization project. It can be useful when debugging the application to speed up starting the project if concentrating only on certain devices.

Online — Determines whether the communication driver attempts to establish communication with the device. This property only applies to communication drivers that support offline mode (in offline mode, the driver does not attempt to establish communication with the device).

MEM file

Log data to MEM file — If this property is active, the runtime software periodically saves real-time data of the device to a MEM file. The MEM file contains a binary image of the device's memory. Thus, the runtime

software running on another computer can obtain real-time data of the device by periodically reading the MEM file.

File name — Specifies the path and name of the MEM file.

Logging interval (s) — Specifies the time period used for saving the MEM file.

Offset in file — Specifies the position in the MEM file at which to write the device's memory image. Thus, it is possible to use a single file to log data from multiple devices.

WARNING Devices' memory images must not overlap.

Communication channel properties

Basic

Data transfer — Specifies the way in which the runtime software accesses the device's data and alarms/events.

Direct — If this option is active (selected), the runtime software obtains data of the device through the device's communication driver.

Network — If this option is active (selected), the runtime software obtains data and alarms/events of the device through a server connection from another instance of the runtime software (data server).

MEM file — If this option is active (selected), the runtime software obtains real-time data of the device by periodically reading a MEM file updated by the runtime software running on another

computer. The MEM file contains a binary image of the device's memory.

If the *Data transfer* property is set to *Direct*, the following properties and options are available:

Basic

The channel properties described below are common to most device types.

Channel type — Specifies the type of channel to be used for communication with the device.

Serial (COM port) — This type of channel is used for communication with serial devices through a serial cable (RS-232 or RS-485/422 interface).

Address — Specifies the device's address and enables you to override the value configured for the device via the ***Device Manager***. In most cases, there is no need to activate this property.

COM port — Specifies the number of COM port used for communication with the device.

Parity — Specifies the parity used for communication with the device.

Comm. timeout (ms) — Specifies the maximum time period between sending a request to and receiving a response from the device by the communication driver. If the device does not respond to the request within the time specified, the communication is recognized as faulty (*Err-timeout*) and the request is sent to the device again. If the device does not

respond repeatedly, it is recognized as a failure in communication with the device.

☞ **RECOMMENDATION** Failures in communication with the device slow down the communication with all the devices connected on the same serial cable. If the device is known to be out of service, it is recommended that you temporarily deactivate the *Online* property.

Comm. speed Specifies the speed (in bauds) used for communication with the device.

👁 **WARNING** The maximum possible communication speed depends on the data transmission line and connection quality.

Dial-up (modem) This type of channel is used for communication with serial devices through a dial-up connection.

Address See the same-named property in the section describing the *Serial (COM port)* channel type.

Comm. timeout (ms) See the same-named property in the section describing the *Serial (COM port)* channel type.

Repeat call count Specifies the number of repeated attemps to establish a dial-up connection.

Limit connection time Determines whether and how to limit the duration of a single dial-up session.

Provider Specifies the telephone service provider to be used to establish a dial-up connection. Based on this property, the appropriate modem is chosen by the device's communication driver.

Phone number — Specifies the number of a telephone line used by the device's modem. Several devices can share the same modem. You can either specify a static value or a string-type tag in order to be able to change the telephone number at runtime.

WARNING By default, *Teco* devices use even parity. For a dial-up connection, it is necessary to configure the device to the same parity as that of the modem.

Network (Ethernet) — This type of connection can only be used for communication with Ethernet-enabled devices.

Address — See the same-named property in the section describing the *Serial (COM port)* channel type.

IP address — Specifies the device's IP address and enables you to override the value configured for the device via the ***Device Manager***. In most cases, there is no need to activate this property.

Comm. timeout (ms) — See the same-named property in the section describing the *Serial (COM port)* channel type.

TCP/UDP port — Specifies the TCP or UDP port to be used for communication with the device.

Network (Ethernet)/Serial (RS-232, RS-485)

This type of connection is used for communication with serial devices connected to the Ethernet network via a Serial-to-Ethernet converter.

IP address — Specifies the IP address of the converter and enables you to override the value configured for the device via the ***Device Manager***. In most cases, there is no need to activate this property.

Comm. timeout (ms) — See the same-named property in the section describing the *Serial (COM port)* channel type.

TCP/UDP port — Specifies the TCP or UDP port to be used for communication with the converter.

Advanced

Control — Specifies the link to a digital-type or integer-type tag to be used to control communication with the device.

Status — Specifies the link to an integer-type tag that should receive the communication status.

Run script on

Start communication — Determines whether to run the specified script after communication with the device is established for the first time.

Error in communication — Determines whether to run the specified script after an error occurs in communication with the device.

Restore communication — Determines whether to run the specified script after communication with the device is restored.

☺ **HINT** In these scripts, it is possible to take special actions (e.g. set and reset the value of an internal tag in order to trigger an alarm).

Do not log communication events (error, restore)

Determines whether to log information about communication-related events to the alarm/event database.

Do not show communication events (error, restore)

Determines whether to show information about communication-related events in the list of current alarms/events.

Comm. error timeout (ms)

Specifies the time period between the moment that the communication driver first time indicates an error in communication with the device and the moment that the *Error in communication* event is generated by the runtime software.

Driver

Connect to driver

Locally — If this option is active (selected), the communication driver starts on the same computer as the runtime software. This is the preferred option.

On remote computer — If this option is active (selected), the communication driver starts on another computer using the DCOM service. The driver must be registered on both computers. It is also necessary to configure security settings for the driver in Windows. Using this option is not recommended.

Time

Regular synchronization

Determines whether and when to synchronize the system time of the device with the system time of the computer. The synchronization is performed every day at the specified time.

Tag-controlled synchronization

Determines whether and when to synchronize the system time of the device with the system time of the computer. The synchronization is performed on the leading edge of a digital-type tag (the off-to-on transition). The *Reset tag* property determines whether to reset the control tag (i.e. set the tag to the off state) after detecting the leading edge.

If the *Data transfer* property is set to *Network*, the following properties and options are available:

Server connection group Specifies the link to a server connection group that contains a server connection to be used as the data source for the device.

Enable sending commands

Determines whether the client runtime software can send commands (e.g. commands for changing process parameters, set-points or

outputs) to the device through the server connection.

Data update interval (ms) Specifies the time interval used for updating the device's real-time data.

If the *Data transfer* property is set to *MEM file*, the following properties and options are available:

File name Specifies the path and name of the MEM file.

Update interval (s) Specifies the time period used for updating the device's memory image by reading the MEM file.

Offset in file Specifies the position in the MEM file from which to read the device's memory image.

2.6.2.5 Connecting data tables

To make a data table (previously defined via the ***Data Table Manager***) accessible to the runtime software running on an actual computer, you have to add a link to the data table object to the logical computer's *Data tables* folder. This is called connecting a data table to a computer.

SQL connection Specifies the SQL connection defining the SQL server and database where the data table's data will be located. It only makes sense for SQL-based data tables. SQL connections can be defined via the ***Project Options*** dialog.

Primary directory Specifies the primary path where the data table's files will be located. If the path is not specified, the „History\Data« subdirectory of the visualization project will be used. If the *Log data* property is active, the primary path should always point to a local drive.

Standby directory Specifies the path to be used as a standby location where the runtime software will search for the data table's files that could not be found on the primary path. The runtime software never logs historical data to this location. If the path is not specified, the „History\Data« subdirectory of the visualization project will be used. If you do not intend to use the standby directory in the way described above, specify the same value as for the primary directory.

🗝 **EXAMPLE** The standby directory is sometimes used in network applications. There is usually a server computer with *Reliance Server* that logs historical data and alarms/events to databases on a local drive. Then there are client computers with *Reliance View* or *Reliance Control* that usually download and maintain a limited number of the data tables' archive files on a local drive, for example the last 30 days. The complete historical data is always on the server computer. The idea is that the operator on a client computer most frequently views recent data, which is stored locally, so the access to the data is faster than over the network. For a data table connected to a client computer, the primary directory is set to a local drive and the standby directory is set to a shared directory on the server that contains the complete historical data. When the operator on a client computer views historical data via the trend viewer in *Reliance View* or *Reliance Control*, the trend viewer searches for the data in preference in the primary directory (i.e. local drive). When the operator continues to view older data and there are no more archive files in the primary directory, then the standby directory is used by the trend viewer to search for the archive files.

Connection Specifies the way in which the runtime software accesses the data table.

Direct Is always active (selected) for a computer that has the *Log data* property active. It can also be active (selected) for a client computer that only views the data table updated by another instance of the runtime software (running on another computer). In both cases, the data is accessed directly (there is no local copy of the data).

Network Must be active (selected) for a client computer that is intended to download the data table's files to a local drive through a server connection with a server computer (the client computer maintains a local copy of the data). This is the preferred way of accessing historical data from multiple computers.

Direct/Network Is only used in some very specialized applications.

Log data

Determines whether the runtime software running on the computer samples real-time data and logs the samples (i.e. historical data) to the data table.

WARNING This property should only be active for one of the computers accessing the data table. Usually, it is the one communicating with the device directly (i.e. via a communication driver). The other computers should only view the data table directly (*Connection* = *Direct*) or download the data table's files to a local drive through a server connection (*Connection* = *Network*).

Create archive files Determines whether the runtime software running on the computer should create archive files for the data table.

Delete oldest archive files Determines whether to delete the oldest archive files so that the number of archive files does not exceed the specified value. The oldest files are deleted when creating a new archive file.

In addition to the above, if the *Connection* property is set to *Network* or *Direct/Network*, the following properties and options are available:

Server connection group Specifies the link to a server connection group that contains a server connection to be used as the data source for the data table.

Limit downloaded archive file count Determines whether and how to limit the number of the data table's archive files downloaded to the client computer. When a new client computer is added to a visualization project and the project is deployed to an actual computer, there is no local copy of the data table when the visualization is launched for the first time. When the client instance of the runtime software connects to a data server, it only downloads the specified number of archive files to a local drive (e.g. the last 6 months), not the complete historical data stored in the data table on the server computer.

Data update interval (s) Specifies the time interval used by the client instance of the runtime software for requesting the data server for updates of the data table.

2.6.2.6 Connecting trends

To make a trend (previously defined via the ***Trend Manager***) accessible to the runtime software running on an actual computer, you have to add a link to the trend object to the logical computer's *Trends* folder. This is called connecting a trend to a computer.

2.6.2.7 Connecting reports

To make a report (previously defined via the ***Report Manager***) accessible to the runtime software running on an actual computer, you have to add a link to the report object to the logical computer's *Reports* folder. This is called connecting a report to a computer.

2.6.2.8 Connecting custom reports

To make a custom report (previously defined via the ***Custom Report Manager***) accessible to the runtime software running on an actual computer, you have to add a link to the custom report object to the logical computer's *Custom Reports* folder. This is called connecting a custom report to a computer.

2.6.2.9 Connecting recipes

To make a recipe (previously defined via the ***Recipe Manager***) accessible to the runtime software running on an actual computer, you have to add a link to the recipe object to the logical computer's *Recipes* folder. This is called connecting a recipe to a computer.

Directory — Specifies the path where the files containing stored recipes will be located. If the path is not specified, the »Settings\Recipes« subdirectory of the visualization project will be used.

2.6.2.10 Defining printers for printing alarms/events

Name — Specifies the printer's name that must be unique within the logical computer and cannot contain illegal characters. In addition, it specifies the name of the printer in the operating system.

Alarm/event online print Determines which types of alarms/events should be printed online (i.e. printed immediately when they occur). To make this feature operational, it is necessary that the printer support printing a single line without ejecting the paper.

WARNING Laser printers and regular ink printers cannot be used for this purpose!

2.6.2.11 Defining modems

In a *Reliance* application, a modem can be used to establish a dial-up connection to a remote modem to which one or more I/O devices are connected.

Basic

Name Specifies the modem's name that must be unique within the logical computer and cannot contain illegal characters.

Provider Specifies the telephone service provider used by the modem.

Control Determines whether to control the modem by the specified integer-type tag.

Only for callbacks Determines whether the modem can also be used for outgoing calls or only for callbacks.

COM port Specifies the number of COM port used for communication with the modem.

Comm. speed Specifies the speed (in bauds) used for communication with the modem.

Advanced

Process DCD signal Determines whether the modem processes the DCD signal.

Dial timeout (s) Specifies the timeout of the modem's dial operation.

Command timeout (s) Specifies the timeout of commands sent to the modem.

Initialization string no. 1, 2

Can be used to specify strings used for initializing the modem.

Dial command, *Hang-up command*, *Reset command*

Can be used to specify basic commands for communication with the modem.

2.6.2.12 Defining a server connection group

In the context of *Reliance*, server connections are the most common way to transfer data of a device (e.g. PLC) between instances of the runtime software running on different computers. A server connection always involves two computers: a client and a server. A server is a computer running a data server (*Reliance Server* or *Reliance Control Server*) that has the data available (either through a communication driver or another server connection or a MEM file). A client is a computer running the runtime software (*Reliance View, Reliance Control, Reliance Server* or *Reliance Control Server*) that needs to get the data from a server. The data is transferred through a so-called socket using the TCP/IP protocol. A socket is a communication channel identified by a unique number called a port.

To define a server connection between two computers, use the ***Project Structure Manager***. First, select the *Server Connection Groups* folder of the client computer and invoke the *New Object* command from the toolbar. This will add a new server connection group to the folder and automatically select it. Now you can add one or more server connections to the group by invoking the *Connect Objects* command and choosing the server computer(s) from the list in the ***Select Computer*** dialog. If you add more than one server connection to the group, you define a redundant configuration. This means that you define one primary and one or more standby (secondary) connections. A primary connection is the one with the highest priority. In the event of a communication failure on the current connection, the runtime software on the client computer automatically attempts to re-establish communication on a connection with lower priority. As soon as a connection with higher priority is again available, communication on the lower-priority connection is terminated. Therefore within each group, there is never more than a single connection being used for communication.

Name Specifies the server connection group's name that must be unique within the client computer and cannot contain illegal characters.

Connection priority Specifies the list of connections in the sequence of priority. To increase/decrease the priority of the selected connection, move it up/down in the list.

Higher priority connections test interval (s)
Specifies the time interval at which the runtime software on the client computer checks for the availability of a higher-priority connection.

2.6.2.13 Defining a server connection

Basic

Name Specifies the connection's name that must be unique within the group and cannot contain illegal characters.

Server computer Specifies the server computer used as a data source for the client computer.

Control communication Specifies whether to control communication on the connection by the specified digital-type tags on client and server computers.

Run script on

Start communication Determines whether to execute the specified script when communication on the connection is established.

Terminate communication

Determines whether to execute the specified script when communication on the connection is terminated.

Transfer alarms/events Determines whether to transfer alarms/events (alerts, commands and system messages) of the devices provided through this server connection. If this property is active, alarms/events of these devices are not generated on the client computer, but alarms/events generated on the server computer are accepted. Thus, both the client and server computers have the same alarms/events in the alarm/event database (each computer has its own local database). The only difference is the receipt time of the alarms/events. This property should always be active.

Limit downloaded archive file count Determines whether and how to limit the number of the alarm/event database's archive files downloaded to the client computer. When a new client computer is added to a visualization project and the project is deployed to an actual computer, there is no local copy of the alarm/event database when the visualization is launched for the first time. When the client instance of the runtime software connects to a server, it only downloads the specified number of archive files to a local drive (e.g. the last 30 days), not the complete historical alarms/events.

Transfer window records Determines whether to transfer window records through this server connection. If this property is active, all information related to window records (adding a new record, editing, ending or deleting a record, etc.) is synchronized through the server connection. This feature is only enabled if the global option *Enable window records* is enabled (see the chapter 2.4.2.4 WINDOW RECORDS).

Advanced

Communication on a server connection between instances of the runtime software is started at the client side. Unless communication at the client side is controlled by a tag, the client runtime software attempts to establish

communication immediately after starting the visualization project; otherwise only in case the communication control tag is equal to 1.

The following situations may occur while establishing communication on a server connection:

1) The server computer has not been found within the network.
2) The server computer has been found within the network. However, the runtime software (data server) is not running on it.
3) The server computer has been found within the network, the runtime software (data server) is running on it, however communication at the server side is currently disabled (this may occur when communication at the server side is controlled by a tag).
4) The server computer has been found within the network, the runtime software (data server) is running on it, and communication at the server side is enabled. In this case, communication is established successfully.

The first situation is usually caused by an incorrectly specified IP address (or hostname, if a constant IP address cannot be specified) of the server computer in the visualization project, or the fact, that the computer is not running. Immediately after the attempt to establish communication, the client runtime software will qualify this situation as an unsuccessful attempt to find the server computer within the network.

Next attempt to establish communication will be made after expiration of the time period defined by the *Idle delay after failure to find the server computer on the network (s)* property. If this situation repeats more than *x*-times (the figure *x* is defined by the *Number of failed attempts to find the server computer on the network before using a standby connection* property), a standby connection is activated (according to priorities), which is defined in the same server connection group.

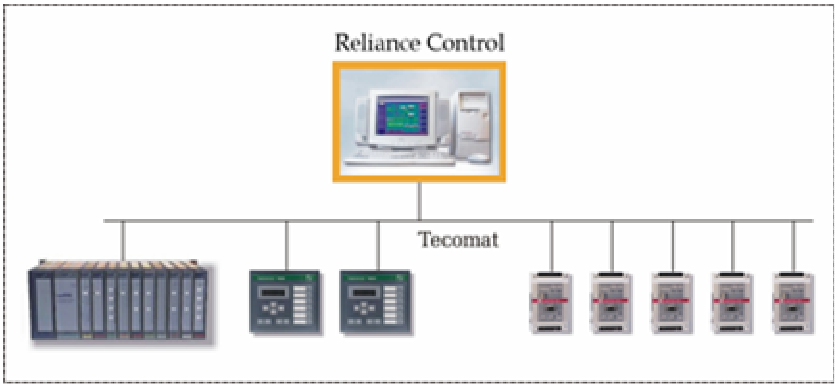
The second and third situation will be qualified as an unsuccessful attempt to establish communication by the client runtime software after expiration of the time period defined by the *Connection timeout (s)* property, and the attempt will be repeated. If this situation repeats more than *x*-times (the figure *x* is defined by the *Connection timeout count before using a standby connection* property), a standby connection is activated (according to priorities), which is defined in the same server connection group.

2.6.2.16 Examples

This chapter contains a few examples to demonstrate various structures of visualization projects; the dashed line indicates individual control areas (e.g. control rooms).

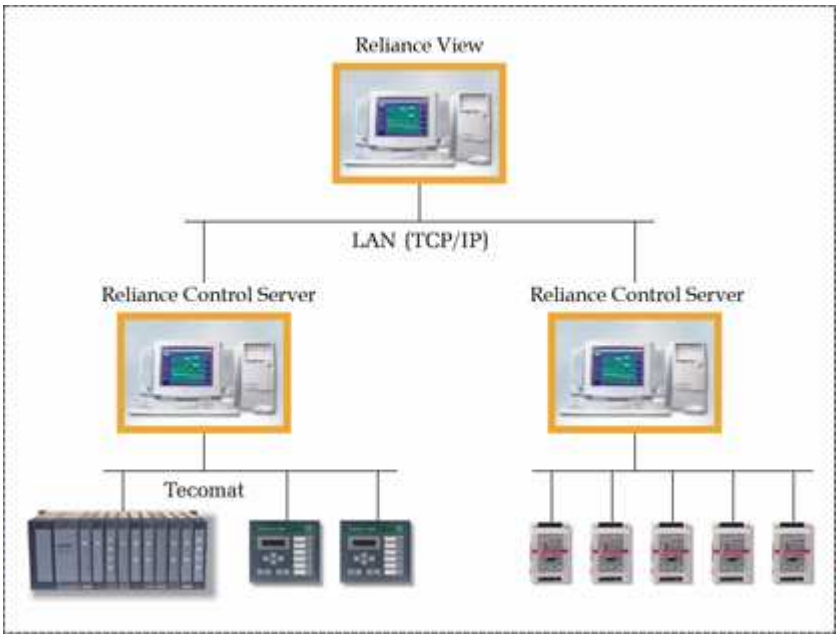
Example 1

This example demonstrates a project structure typical for minor applications of *1 computer – n devices* type. The visualization project contains a single control area. The control area consists of a single computer and several *Tecomat* devices.



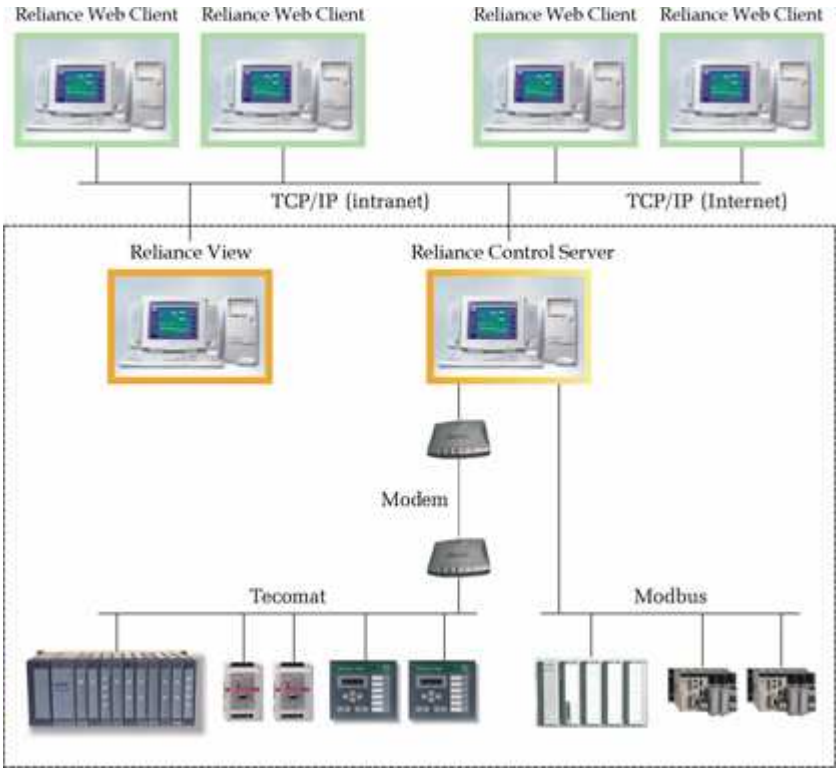
Example 2

This example demonstrates an application that uses network communication between instances of the runtime software. The visualization project contains a single control area. The control area consists of three computers running the runtime software that exchange process data through a LAN network using the TCP/IP protocol.



Example 3

This example demonstrates an application that uses network communication between instances of the runtime software and distribution of process data to *Reliance Web Client*s. The visualization project contains a single control area. The control area consists of two computers – a computer running *Reliance Control Server* and a computer running *Reliance View*. The computer running *Reliance Control Server* is connected to a network of *Tecomat* devices through a switched telephone line and to a network of *Modbus* devices through a serial cable. *Reliance Control Server* also enables the *Reliance Web Client*s to connect to it from a local intranet network or over the Internet. *Reliance View* obtains process data from *Reliance Control Server* through a LAN network.



Example 4

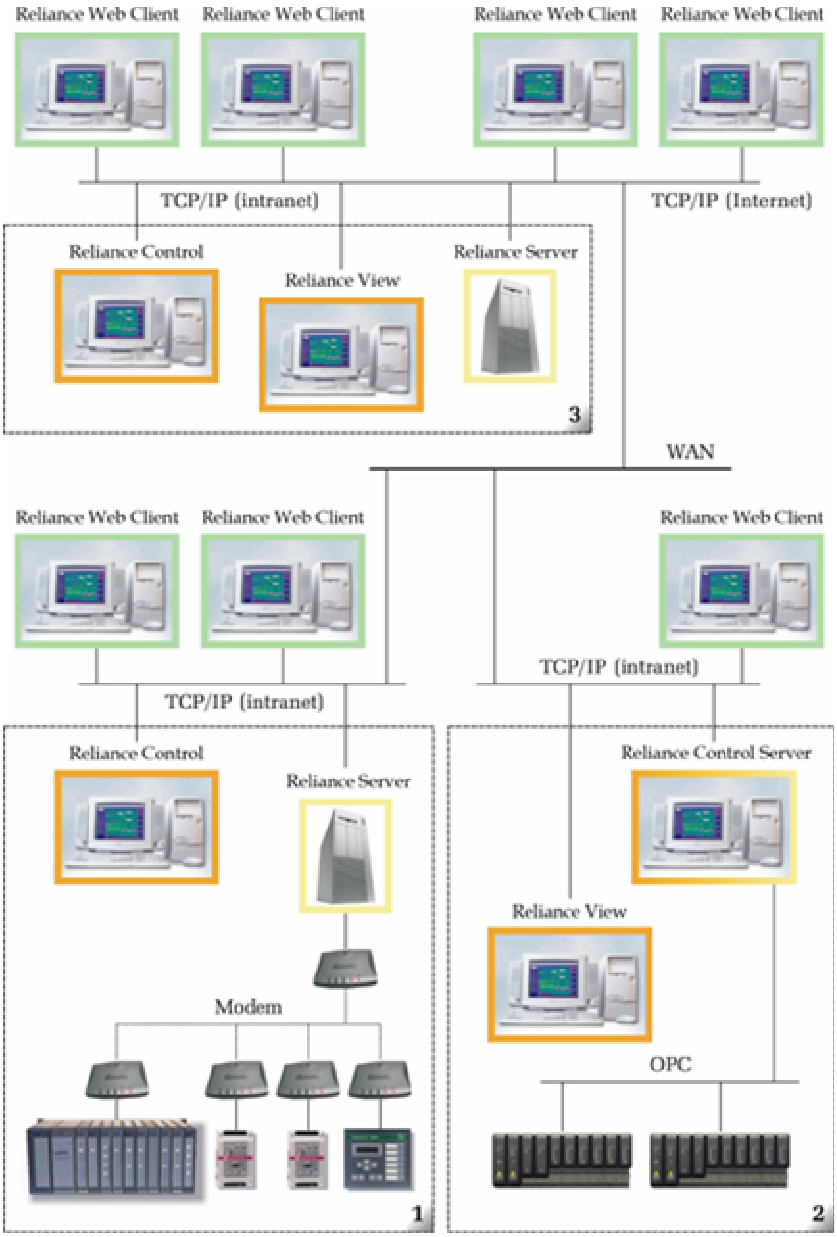
This example demonstrates a major application divided into three control areas.

The control area no. 1 consists of two computers. The computer running *Reliance Server* is connected to a network of *Tecomat* devices through a switched telephone line. *Reliance Control* running on the other computer obtains data of the devices from *Reliance Server* through a LAN network and allows monitoring and controlling the devices.

The control area no. 2 consists of two computers. The computer running *Reliance Control Server* is connected to the devices through a serial cable. *Reliance View* running on the other computer obtains data of the devices from *Reliance Control Server* through a LAN network and allows monitoring the devices.

The control area no. 3 is a central control area collecting process data from subordinated locations. It consists of three computers. The computer running *Reliance Server* is connected to the other control areas through a WAN network. *Reliance Server* obtains process data from the data servers running on computers within the other control areas. *Reliance View* and *Reliance Control* running on the other computers within this control area obtain data of the devices from *Reliance Server* through a LAN network and allow monitoring (both *Reliance View* and *Reliance Control*) and controlling (only *Reliance Control*) the devices located in the other control areas (1 and 2). When a command is sent to a device from *Reliance Control*, it is passed to *Reliance Server* within this control area through the LAN network and then it is passed to the data server within the appropriate control area.

Reliance Server and *Reliance Control Server* also enable *Reliance Web Client*s to connect to them from a local intranet network or over the Internet.


Reliance Web Client
Reliance Web Client
Reliance Web Client
Reliance Web Client
TCP/IP (intranet)
TCP/IP (Internet)
Reliance Control
Reliance View
Reliance Server
3
WAN
Reliance Web Client
Reliance Web Client
Reliance Web Client
TCP/IP (intranet)
TCP/IP (intranet)
Reliance Control
Reliance Server
Reliance Control Server
Reliance View
Modem
OPC
1
2

2.6.3 Device Manager

In the context of *Reliance*, a device is an object representing a physical device (e.g. PLC or other similar I/O device used for industrial automation and control) or a so-called virtual device. Within each device, you can define tags and alarms/events linked to the tags. The values of the tags are used to generate alarms/events (e.g. alerts), can be displayed via visual components, can be used to control the appearance and behavior of components, can be logged to data tables, processed by scripts, etc. If a device represents a physical device, the runtime software obtains the tags' data from the device's communication driver or another instance of the runtime software (e.g. through a server connection from a data server). Otherwise, the device's tags only exist in the memory of the computer on which the visualization project is running and are referred to as internal tags.

Every visualization project contains a predefined virtual device named *System*. This device is automatically accessible to each computer defined within a visualization project (i.e. you need not connect this device to a logical computer via the ***Project Structure Manager***). It is intended for defining private internal tags. It cannot be shared among the computers in any way (i.e. cannot be provided through a server connection, etc.).

2.6.3.1 Defining a device

Name — Specifies the device's name that must be unique within the project and cannot contain illegal characters.

Other device properties (e.g. the address) may vary according to device type. For information on properties of the most common device types, see the following sections.

Defining a Teco device

Basic

Address — Specifies the device's address.

IP address — Specifies the device's IP address on the Ethernet network. It is used by the communication driver

for network communication with Ethernet-enabled devices, such as TC700, or serial devices connected to the Ethernet network via a Serial-to-Ethernet converter. In the latter case, the property specifies the IP address of the converter.

Model Can be used to specify the device's model. In most cases, it is not required.

Databox

Enable reading/writing Databox

Determines whether to enable the communication driver to read/write data from/to the Databox (supplementary memory of *Tecomat* devices).

Offset Specifies the integer-type tag to be used to control the initial address of the data block to be read from the device.

Length Specifies the integer-type tag to be used to control the length of the data block to be read from the device.

Control Specifies the digital-type tag to be used to control the reading. The reading starts on the leading edge of the tag (the off-to-on transition).

Buffer Specifies the array-type tag to be used to store the data block read from the Databox.

Status Specifies the integer-type tag to be used to store the current status of the reading operation. It may contain the following values:

0... ready
1... the read operation in progress
2... the read operation completed successfully
3... the read operation failed
11. the write operation in progress

12. the write operation completed successfully
13. the write operation failed

Defining a Modbus device

Address Specifies the device's address.

IP address Specifies the device's IP address on the Ethernet network.

Swap bytes Determines whether the communication driver should exchange (swap) the byte order within the data of tags defined in *User registers* and *Input registers* when the data is read/written from/into the I/O device. This property must be active if the I/O device stores most-significant byte first (big-endian form) since *Reliance* stores least-significant byte first (little-endian form).

Swap words Determines whether the communication driver should exchange (swap) the word order within the data of tags of type *DoubleWord*, *LongInt*, *Float* and *DoubleFloat* when the data is read/written from/into the I/O device. It must be active if the I/O device stores most-significant word first.

Supported communication functions

Determines which communication functions the I/O device supports. The communication driver will only use the selected (active) functions. For example, if the I/O device is known to not support the *Preset Multiple Regs 16* function, you must deactivate the respective option.

Defining an OPC device

OPC (OLE for Process Control) is a worldwide standard for exchanging process data between computer programs. Defining an OPC device allows *Reliance* to connect to any OPC server that meets the specification OPC DA

1.0 or 2.0. It means that if the manufacturer of a hardware device (or a third party company) develops an OPC server for the device, then *Reliance* (which is an OPC client) can use the OPC server to obtain process data from the device and send commands to the device.

Each OPC server has two unique identifiers within the Windows operating system – so-called *ProgID* and *GUID*. *ProgID* contains an identification string of the program (e.g. *Matrikon.OPC.Simulation.1*). *GUID* contains a unique identification number generated by each OPC server manufacturer and this number should be unique throughout the world (no other program should use this number as *GUID*).

OPC server Prog ID — Specifies a unique identifier of the OPC server. It is supplied automatically after selecting the OPC server. To select an OPC server, click the ellipsis (...) button.

OPC server GUID — Specifies a unique identifier of the OPC server. It is supplied automatically after selecting the OPC server. To select an OPC server, click the ellipsis (...) button.

Import tags from a remote computer — Enables you to configure importing tags from an OPC server running on a different computer or device.

2.6.3.2 Importing and exporting tags

Reliance allows users to import tags from external files (exported for example from development tools used for PLC programming).

Teco

Tags of *Teco* PLCs and controllers may be imported from a file in the **.pub* (xPro) or **.tdr* format (Mercur, Epos for Windows). A structure-type tag (which can be defined via Mercur or Epos) is imported as a group of tags corresponding to the structure fields. Names of the tags are generated

according to the syntax ***StructureName_FieldName***. An array-type tag (which can be defined via Mercur, Epos) is imported as a group of tags corresponding to the elements of the array. Names of the tags are generated according to the syntax ***ArrayName_ElementIndex***.

OPC

To import tags into the current OPC group directly from the OPC server, click the *Import from OPC server* button. However, the server must support the *IOPCBrowseServerAddressSpace* interface. Tags may be imported to the current OPC group also from a file in the **.csv* format or exported in this format. Each line must have the following structure:

OPC ItemID; Tag name; Tag data type; Comment

2.6.3.3 Defining a tag

Basic

Name	Specifies the tag's name that must be unique within the device and cannot contain illegal characters.
Eng. name	Specifies an optional engineering name for the tag.
Units	Specifies units of measurement (engineering units) for the tag.

Other properties of tags (e.g. data type) differ according to device type.

Advanced

String character count	Specifies the number of characters for tags of type *String* and *Array of String*.
Dec. place count	Specifies the number of decimal places displayed. It only makes sense for floating point-type tags, integer-type tags that have the *Analog value correction* property active and derived array-type tags.

Array element count Specifies the number of elements of array-type tags.

Matrix row count Specifies the number of matrix rows. This property is only used for array-type tags defined within *AMiT* devices. In such case, *Reliance* uses array-type tags to represent matrix-type data used in the I/O devices.

Matrix column count Specifies the number of matrix columns. This property is only used for array-type tags defined within *AMiT* devices. In such case, *Reliance* uses array-type tags to represent matrix-type data used in the I/O devices.

Save latest value Determines whether to save the latest value of the tag before terminating the visualization project; when the project is started next time, the saved value is used to initialize the tag in the memory of the runtime software.

Initial value Determines whether to initialize the tag with the specified value when starting the visualization project. This value may later be replaced with the value saved as the latest (see the *Save latest value* property).

Log write commands to alarm/event database
Determines whether to log information about all "write" operations performed on the tag to the alarm/event database. The *Text* property is optional. If it is not specified, a default text will be used.

Format Specifies the format used to display the tag's value at runtime (*Decimal*, *Hexadecimal*). The hexadecimal representation can only be used for integer-type tags (i.e. tags of type *Byte*, *Word*, *DoubleWord*, *Integer*, and *LongInt*) and derived array-type tags.

Correction

Analog value correction Determines whether to use a correction when computing the value of the tag. If this property is active, the runtime software uses the corrected value and the original value is not available anymore.

Negate value if digital Determines whether to negate the value of the tag. It only makes sense for tags of type *Bool* and *Array of Bool*. If this property is active, the runtime software uses the negated value and the original value is not available anymore.

Limits

These properties let you define high and low limits (critical and warning) for the tag. Limits may be either static or dynamic. The value of a static limit is defined as a constant property. The value of a dynamic limit is controlled by the specified numeric-type tag. A limit can be used to generate an alarm/event when the tag's value exceeds or falls below the limit.

Sharing

The runtime software acts as a DDE server.

DDE Determines whether the runtime software provides other programs (the so-called DDE clients, e.g. MS Excel, MS Word) with the real-time value of the tag using the DDE (Dynamic Data Exchange) standard.

When defining a link to the tag in a DDE client program, use the following syntax:

{=runtime_software|DdeServer!DdeItem}

runtime_software....the runtime software's file name without extension (*R_View*, *R_Ctl*, *R_CtlSrv*, *R_Srv*)
DdeItem ..... the name specified as the *DDE item* property

🗝 **EXAMPLE**

{=R_CtlSrv|DdeServer!Water_Temperature}

Defining a tag of a Teco device

Register type	Specifies the type of register where the tag is stored in the device.
Address	Specifies the register address where the tag is stored in the device.
Tag data type	Specifies the data type (format) for the tag.

Defining a tag of a Modbus device

Register type	Specifies the type of register where the tag is stored in the device. A *Modbus* device has 4 types of registers – Outputs (*Coils*), Inputs (*Inputs*), User registers (*Holding registers*) and Input registers (*Input registers*).
Address	Specifies the register address where the tag is stored in the device.
Tag data type	Specifies the data type (format) for the tag. Available data types depend on the value of the *Register type* property.

Defining a tag of an OPC device

Tags in an OPC device cannot be added directly into the *Tags* folder. First, you have to add a special folder called an OPC group. Then you can either define new tags within the group or import tags to the group from an OPC server.

Group properties

Name — Specifies the group's name that must be unique within the device and cannot contain illegal characters.

Update interval (ms) — Specifies the time interval used for updating the tags defined within the group.

Dead band (%) — Specifies the percentage of change in the value of a tag defined within the group in order for the OPC server to send the new value of the tag to the runtime software.

OPC group state — Determines the group's state which affects whether the tags defined within the group will periodically be updated by the OPC server.

- *Inactive* — The group's tags will never be updated.
- *Active* — The group's tags will be updated.
- *Tag controlled* — The group's tags will only be updated when the value of the selected control tag is equal to 1.

Tag properties

Tag data type — Specifies the data type (format) for the tag.

OPC ItemID — Specifies the identifier used for the tag within the OPC server.

2.6.3.4 Defining a communication zone

Communication zones let you fully control the communication with I/O devices. However, defining them is optional (communication zones are not required for communication).

Name Specifies the zone's name that must be unique within the device and cannot contain illegal characters.

Defining a communication zone of a Teco device

Register type Specifies the type of register where the zone is located.

Address Specifies the register address where the zone begins.

Length (byte count) Specifies the length of the zone in bytes.

Reading interval (ms) Specifies the time interval used for reading the zone by the communication driver.

Control reading Determines whether to control reading the zone by the specified digital-type tag.

Defining a communication zone of a Modbus device

Register type Specifies the type of register where the zone is located.

Address Specifies the register address where the zone begins.

Length (element count) Specifies the length of the zone in elements.

Reading interval (ms) Specifies the time interval used for reading the zone by the communication driver.

Control reading Determines whether to control reading the zone by the specified digital-type tag.

2.6.3.5 Defining an alarm/event

Basic

Name — Specifies the alarm's name that must be unique within the device and cannot contain illegal characters.

Text — Specifies the text of the alarm/event.

Tag — Specifies the link to the tag related to the alarm/event. The tag must belong to the same device as the alarm/event.

Condition — Specifies the condition that generates the alarm/event.

- *Data change* — The alarm/event is generated when the value of the tag changes in the specified way (*Any change*, *Increment*, *Decrement*).
- *Leading edge* — The alarm/event is generated when the value of the digital-type tag changes from 0 to 1 (the off-to-on transition). The alarm/event remains active until the tag changes its value back to 0.
- *Trailing edge* — The alarm/event is generated when the value of the digital-type tag changes from 1 to 0 (the on-to-off transition). The alarm/event remains active until the tag changes its value back to 1.
- *High critical limit* — The alarm/event is generated when the value of the tag becomes equal or greater than the high critical limit of the tag.
- *High warning limit* — The alarm/event is generated when the value of the tag becomes equal or greater than the high warning limit of the tag.

Low warning limit The alarm/event is generated when the value of the tag becomes equal or less than the low warning limit of the tag.

Low critical limit The alarm/event is generated when the value of the tag becomes equal or less than the low critical limit of the tag.

Miscellaneous

Type An alarm/event can be of one of the following types: alert, command and system message. Alarms/events of all types are stored in a single table.

Priority Affects the order in which the sounds triggered by the alarm/event's start and end should be played. The alarm/event with higher value of this property has higher priority when playing the sounds.

Advanced

Options

Log to alarm/event database

Determines whether to log the alarm/event to the alarm/event database.

Show in current alarms/events

Determines whether to show the alarm/event in the list of current alarms/events.

Require acknowledgement Determines whether acknowledgement by the operator is required for the alarm/event before it can be removed from the list of current alarms/events.

Verify user on acknowledge

Determines whether to verify the operator when acknowledging the alarm/event. If this property is active, the alarm/event can only be acknow-

ledged upon successful verification of the operator's identity.

Related objects

Window — Determines whether the specified window is related to the alarm/event. If this property is active, the operator can bring up the window by choosing a special command.

Digital tag — Determines whether the specified digital-type tag is related to the alarm/event. The tag is set to 1 and kept on this value by the runtime software while the condition that generated the alarm/event persists. If the condition is not present, the tag is set to 0 and kept on this value by the runtime software.

Actions

On start

Run script — Determines whether to run the specified script when the alarm/event is generated.

Play sound — Determines whether to play the specified sound file (in the **.wav* format) when the alarm/event is generated. If this property is active and the sound file is not specified, the runtime software uses the default sound (see the chapter 2.4.2.1 ALARMS/EVENTS).

Show current alarms/events

Determines whether to activate the list of current alarms/events when the alarm/event is generated. This feature can be disabled individually for each computer defined in the project (see the chapter 2.6.2.2 DEFINING COMPUTERS).

Online print — Determines whether to print the alarm/event on the printer (intended for online printing,

see the chapter 2.6.2.10 DEFINING PRINTERS FOR PRINTING ALARMS/EVENTS) when the alarm/event is generated.

On end

Run script — Determines whether to run the specified script when the condition that generated the alarm/event ceases to exist.

Play sound — Determines whether to play the specified sound file (in the **.wav* format) when the condition that generated the alarm/event ceases to exist. If this property is active and the sound file is not specified, the runtime software uses the default sound (see the chapter 2.4.2.1 ALARMS/EVENTS).

On acknowledge

Run script — Determines whether to run the specified script when the alarm/event is acknowledged by the operator.

Information

Comment — Can be used to specify an optional comment about the alarm/event. The comment is intended especially for the author of the visualization project (not the operator).

Description — Can be used to specify an optional description of the alarm/event. In multi-language projects, the description can be localized (i.e. translated into all project languages). It is meant as an explanation for the operator. The operator can display it by choosing a special command.

2.6.4 Data Table Manager

The ***Data Table Manager*** allows you to define and configure data tables. In the context of *Reliance*, a data table is an object representing a physical table in a database. By defining a data table you enable the runtime software to access (read and/or write) historical data.

Data table properties

Basic

Name — Specifies the data table's name that must be unique within the project and cannot contain illegal characters.

Physical table name — Specifies the name used for the table in a database (for SQL-based data tables) or the first part of a filename (for file-based data tables, i.e. *Paradox*, *DBase*).

Data acquisition method — Specifies the way in which historical data for the data table will be acquired.

- *Not specified* — If this option is active, the runtime software does not log data to the data table using the built-in support. However, you can create custom scripts in order to access the data table's files directly.
- *Sample real-time data* — If this option and the *Log data* property of the data table connected to a computer are active, the runtime software running on the computer samples real-time data and logs the samples (i.e. historical data) to the data table using the built-in support (also see the chapter 2.6.2.5 CONNECTING DATA TABLES).

Sampling — Specifies when the sampling is performed.

- *Periodic* — If this option is active, the sampling is performed periodically using the specified time interval.

Tag-controlled — If this option is active, the sampling is performed on the leading edge (the off-to-on transition) of the specified digital-type tag. The *Reset tag* property determines whether the control tag should be set to 0 after detecting the leading edge of the tag by the runtime software. If this property is active, it is important that each data table use a different control tag.

Script controlled — If this option is active, the sampling is performed when the RDB.AppendRecord method is called in a script. For more information see the Script help.

Time stamp source — Specifies the way in which a sample's time stamp will be acquired.

Computer time — If this option is active, the time stamp for a new sample (record) is equal to the current time of the computer.

Tag — If this option is active, the time stamp for a new sample (record) is equal to the current value of the specified control tag.

Blocking tag — Determines whether to use the specified digital-type tag to block the sampling. If the value of the tag is equal to 0, sampling is enabled. Otherwise, sampling is disabled.

Block sampling if one or more tags have invalid value

If this property is active, the sampling is blocked while the value of one or more tags is not valid.

Advanced

Database type — Determines the data format used for the data table (*Paradox*, *DBase*, *SQL*).

Not indexed — Determines whether to create and maintain the primary index file for the data table. It only applies to Paradox-type data tables. If this property is not active, the data table is

indexed by date and time of the records. The index ensures that the records are always sorted by date and time when accessing the data table. By default, this property is not active.

Archive files — Determines whether and how to create archive files for the data table. An archive file is created by copying the data table's current file (the file to which the runtime software logs new records) to the data table's archive directory and renaming it. If the operation is successful, the current file is deleted. Archiving can be performed periodically (*Monthly* or *Daily*) or not at all (*None*). This property only makes sense for file-based data tables, i.e. *Paradox*, *DBase*.

Data delay — Is only used in some very specialized applications.

Data table field properties

Name — Specifies the field's name that must be unique within the data table and cannot contain illegal characters.

Tag — Specifies the link to the tag whose value is to be logged to the data table.

2.6.5 Trend Manager

The ***Trend Manager*** allows you to define and configure trends. In the context of *Reliance*, a trend is an object used for graphic presentation of the data stored in a data table(s) (i.e. historical data). Trends can be displayed at runtime via the trend viewer.

Trend properties

Name — Specifies the trend's name that must be unique within the project and cannot contain illegal characters.

Title — Specifies the text to be displayed as the title of the trend using the specified font. If the *Use trend alias or name* property is active, the trend's alias (or name, if alias is not specified) is used as the title.

Background

Color — Specifies the background color of the trend. The color should contrast with colors used for individual series.

Ruler — Determines the color and width of the ruler. The ruler is a vertical straight line drawn on the background. It is designed for accurate reading of values of individual series at the crossing point of the ruler and the series. The color of the ruler should contrast with the background color.

Vertical axis — Enables you to configure the behavior of the vertical axis common to all series. The *Hidden* property can be used to hide the axis in cases when each series uses its own vertical axis. The *Automatic* property determines whether the axis automatically adjusts its minimum and/or maximum to the series' values within the current time range. If the axis' minimum and/or maximum are not automatic they must be specified as the *Minimum* and/or *Maximum* properties.

Trend type — Determines how to graphically represent the series' values.

Display style — Determines the time range displayed on a single page of the trend. If *Point count* is active, the trend viewer always attempts to display the specified number of series points on a single page. In this case, it is required that all the series

be linked to the same data table so that each series displays points with the same time stamps. If *Time range* is active, the trend viewer always displays the specified time range on a single page regardless of the amount of series points.

Trend series properties

Name — Specifies the series' name that must be unique within the trend and cannot contain illegal characters.

Data table field — Specifies the link to the data table field whose values (time samples) are to be displayed by the series.

Series

- *Color* — Determines the series' color. It should contrast with the background color of the trend.

Vertical axis — Determines whether the series has its own vertical axis or uses the vertical axis common to all series. The other properties enable you to configure the behavior of the vertical axis private to the series. The *Hidden* property can be used to hide the axis. The *Automatic* property determines whether the axis automatically adjusts its minimum and/or maximum to the series' values within the current time range. If the axis' minimum and/or maximum are not automatic, they must be specified as the *Minimum* and/or *Maximum* properties.

2.6.6 Real-Time Trend Manager

The ***Real-Time Trend Manager*** allows you to define and configure real-time trends. In the context of *Reliance*, a real-time trend is an object used for graphic presentation of a sequence of the most recent values (time samples) of the selected tags. The values (time samples) are only stored in the

memory, not in a data table. A real-time trend can be displayed at runtime by a *Real-Time Trend* component placed into a visualization window.

Real-time trend properties

Name — Specifies the trend's name that must be unique within the project and cannot contain illegal characters.

Update — Determines when to update the trend (i.e. when to add the next time sample to the trend). Updating can be done periodically (*Periodic*) at the specified time interval or on the leading edge (the off-to-on transition) of the specified digital-type tag (*Tag-controlled*).

Visible point count — Specifies the number of points (time samples) to be displayed by the trend.

Blocking tag — Determines whether to use the specified digital-type tag to block updating the trend. If the value of the tag is equal to 0, updating is enabled. Otherwise, updating is disabled.

Real-time trend series properties

Name — Specifies the series' name that must be unique within the trend and cannot contain illegal characters.

Tag — Specifies the link to the tag whose values (time samples) are to be displayed by the series.

Series

Color — Determines the series' color. It should contrast with the background color of the trend (see the chapter 2.9.13 REAL-TIME TREND).

Line width Determines the width (in pixels) of the line used for drawing the series.

2.6.7 Report Manager

The ***Report Manager*** allows you to define and configure reports. In the context of *Reliance*, a report is an object used for graphic presentation of the data stored in a data table (i.e. historical data) in a tabular format. Reports can be displayed, printed, exported and sent by e-mail at runtime via the report viewer.

Report properties

Basic

Name Specifies the report's name that must be unique within the project and cannot contain illegal characters.

Data table Specifies the link to the data table whose contents are to be displayed by the report.

Report elements Determines which elements to include in the report.

Grid Determines whether to display grid lines on the background.

Row height Determines the height (in pixels) of a single row of the report. It may affect the number of rows displayed on a single page of the report.

Report title

Use report alias or name Determines whether to use the report's alias (or name, if alias is not specified) as the title.

Text

Font Determines the font of the title.

Background

Color — Specifies the background color of the title. The color should contrast with the text color.

Bar height — Specifies the height (in pixels) of the report header bar.

Frame — Determines whether to frame the report header with a line of the specified color and width (in pixels).

Alignment — Determines the alignment of the title within the report header.

Column header

Frame — Determines whether to frame the column header with a line of the specified color and width (in pixels).

Height — Specifies the height (in pixels) of the column header.

Page footer

Text on footer — Determines whether to display the specified text on the footer of each page of the report using the specified font and alignment.

Page numbers — Determines whether to display the page number on the footer of each page of the report using the specified font and alignment.

Page numbers and text on separate rows

Determines whether to print the page number and footer text on separate rows. If this property is not active, the footer is printed on a single line.

Background

Color — Specifies the background color of the footer.

Height — Specifies the height of the footer.

Report item properties

Basic

Name — Specifies the item's name that must be unique within the report and cannot contain illegal characters.

Data source — Specifies the link to a data source. The data source may be a data table field or date or time of a data table record. For the most part, the first two report items are linked to date and time, and the other report items are linked to data table fields.

Text

Font — Determines the font of the item's text.

Background

Color — Determines the background color of the item's text.

Value format — Specifies the format string to be used when converting the item's value to text for display purposes. It is only used for report items linked to numeric data table fields.

Column width — Specifies the width (in pixels) of the item's column.

Alignment — Specifies the alignment of the item's text within the item's column.

Item title

Use item alias or name Determines whether to use the item's alias (or name, if alias is not specified) as the column title.

Text

Font Determines the font of the item's column title.

Background

Color Determines the background color of the item's column title.

Alignment Specifies the alignment of the item's column title within the column.

2.6.8 Custom Report Manager

The ***Custom Report Manager*** allows you to define and configure custom reports. In the context of *Reliance*, a custom report is an object used for graphic presentation of real-time values of the selected tags in a user-defined format. Custom reports can be displayed, printed, exported and sent by e-mail at runtime via the custom report viewer.

When defining a new custom report, it is necessary to prepare a template in FastReport, HTML or text format and save it to a disk file. The template may also include graphic elements if using FastReport or HTML format. To mark the place where the value of a tag should occur in the template, use a special character string with the following format: **{$Item_name}**. **Item_name** is the name specified for the custom report item.

To prepare a template in the FastReport format, choose the ***Edit Report*** command which brings up a built-in designer. For more information on designing templates in the FastReport format, see the context sensitive help available in the designer.

Custom report properties

Name	Specifies the report's name that must be unique within the project and cannot contain illegal characters.
File name	Specifies the file that contains the template for the report.

Custom report item properties

A custom report item corresponds to the mark contained in the template for the report.

Name	Specifies the item's name that must be unique within the custom report and cannot contain illegal characters.
Tag	Specifies the link to the tag whose value is to be displayed on the place(s) marked in the template.
Show eng. units	Determines whether to display units of measurement after the value of the tag.
Test value	Specifies the value to be used instead of the real-time value of the tag when previewing the report.
Occurrence count	Indicates how many times the item occurs in the template. It is updated every time the list of items is updated by the *Load Report Items* command.

2.6.9 Recipe Manager

The ***Recipe Manager*** allows you to define and configure recipes. In the context of *Reliance*, a recipe is an object representing a group of tags. It is used to store real-time values of the tags to a disk file. A stored recipe can later be loaded from the file and transferred to the respective devices.

Recipe properties

Name Specifies the recipe's name that must be unique within the project and cannot contain illegal characters.

Secure

Using Specifies the access rights required for using the recipe (i.e. loading a stored recipe from a file into the respective devices).

Saving Specifies the access rights required for saving the recipe to a file (i.e. creating a stored recipe).

Deleting Specifies the access rights required for deleting a stored recipe (i.e. deleting the file containing the stored recipe).

Recipe transfer to devices

Ack. Specifies the link to the digital-type tag that should receive acknowledgment (by setting the value to 1 by the runtime software) of loading the stored recipe from a file and transferring it to the respective devices. This property is optional.

Automatic transfer

Control Specifies the link to the digital-type tag to be used to control the automatic transfer of a stored recipe to the respective devices (the transfer does not require the operator's command). The transfer is performed on the leading edge of the tag (the off-to-on transition).

Prefix Specifies the link to a string-type tag to be used to control the prefix of the name of the file containing a stored recipe to be transferred to the respective devices.

Recipe item properties

Name	Specifies the item's name that must be unique within the recipe and cannot contain illegal characters.
Tag	Specifies the link to the tag whose value is to be stored as part of a stored recipe.

2.6.10 Picture Manager

Graphic elements are usually used in visualization projects for better illustration and more precise appearance. They can be divided into raster and vector graphics formats, i.e. bitmaps and vectors. *Reliance* works with raster files in the **.bmp*, **.jpg*, **.gif* and **.png* format as well as **.wmf* and **.emf* vector formats. Bitmaps are raster pictures with fixed resolution (size in pixels). Bitmaps are in general more effective than vector graphics. However, if their size is changed, deformations and loss of picture information occur. All vector graphics formats are saved as mathematically defined curves and no quality loss occurs, if their size changes.

Graphic elements in *Reliance* are commonly called pictures. If you want to use a picture in a visualization project, it must be first imported to a picture database of the project via the ***Picture Manager***. To bring up the ***Picture Manager***, choose the ***Managers*** ▸ ***Picture Manager*** command.

The ***Picture Manager*** enables you to create a single-level folder structure (displayed in the top left pane of the window), import pictures to the database from various locations (preferably from the Picture library) using the appropriate target folder, view the list of pictures (displayed in the right pane of the window) included in the selected folder and view and/or edit the selected picture.

Description of toolbar commands

Add Pictures	Is used to import pictures from graphic files (**.bmp*, **.gif*, **.jpg*, **.png*, **.wmf*, **.emf*) to the picture database.
Edit Picture	Is used to run an external program (defined via the ***Environment Options*** dialog) in order to edit the picture. After editing the picture and closing

the program, you have to choose the *Refresh Pictures* command in order to see the changes immediately.

2.6.11 Script Manager

In *Reliance*, it is possible to configure most of the behavior and properties of the runtime environment by setting visualization project options and the properties of individual components. For special actions that cannot be realized this way, *Reliance* is equipped with a possibility to write pieces of program code – so-called scripts.

In the context of *Reliance*, a script is a piece of program code written in the *Visual Basic Script* scripting language (hereinafter referred to as *VBScript*). The script may perform calculations, operations on tags, databases and files, send e-mail or GSM SMS messages and perform many other operations. An important feature is communication with external programs through a COM/DCOM-based interface. Thus, it is possible to connect to an external program (e.g. MS Word or MS Excel) from a script and call its procedures and functions.

Each script in a *Reliance* project can be of one of the following types:

- Time scripts
- Key scripts
- Data change scripts
- Periodic scripts
- Condition scripts
- Event scripts

Common script properties

These properties are common to scripts of all types. Each script can also have additional specific properties depending on its type.

Name	Specifies the script's name that must be unique within the project and cannot contain illegal characters.
Type	Specifies the script's type mentioned above.
Disable execution	Determines whether the script is disabled by default. If this property is active, the script cannot

be executed. However, the script can get enabled or disabled from another script's code (see the Script help).

Thread

Specifies the thread of execution in which the script be processed. By dividing scripts into different threads, it is possible to achieve a parallel processing of multiple scripts. This is convenient e.g. when there are 2 script groups: scripts for fast operations (e.g. calculations of internal tags for display purposes) and lengthy operations (e.g. with databases). As a result, lengthy operations don't block fast operations.

Run on thread initialization

Must be active for scripts containing definitions of global identifiers (constants, variables, procedures, functions, etc.) intended to be accessible from other scripts. The thread initialization is performed at project startup and upon a possible forcible termination of the script (see below). Global identifiers are not shared among scripts executed in different threads.

Terminate after exceeding max. execution time

Determines whether to forcibly terminate the script execution when exceeding a time period after which a normal completion of the script is improbable (e.g. 60s). This situation can be caused by the wrong algorithm (e.g. continuous loop) or a lengthy response of a function called from the script (e.g. function of an external program that is not currently responding). It is recommended that you only use this option in the most extreme cases since terminating the script execution is a dangerous operation which may cause the program to hang or crash.

Max. execution time (ms)

Specifies the maximum execution time for the script. If the time is exceeded, the script is forcibly terminated. It must be greater than

0. If it is equal to 0, the execution time is not limited.

Priority Specifies the script's priority of processing. The priority increases with the value of this property.

2.6.11.1 Time scripts

Time scripts get executed every day at the specified time including the possibility of periodical repetition at the specified time interval.

First execution time (hh:mm:ss)
Determines the time of the first execution of the script.

Execute repeatedly Determines whether to execute the script periodically at the specified time interval until the time specified by the *Repeat until* property. If this property has a value of 0 hours, 0 minutes, 0 seconds, the time is only limited by midnight.

2.6.11.2 Key scripts

Key scripts get executed when the specified key shortcut is pressed.

Execute on pressing keys
Specifies the key shortcut to be pressed in order for the script to be executed.

2.6.11.3 Data change scripts

Value change scripts get executed when the specified tag changes its data value or quality.

Execute on change in data value of tag
Specifies the link to the tag related to the script.

Execute even when data becomes valid

Determines whether to execute the script even when the tag becomes valid (although the value remains unchanged).

2.6.11.4 Periodic scripts

Periodic scripts get executed periodically at the specified time interval.

Repeat interval (hh:mm:ss:fff)

Specifies the time interval to be used for periodical execution of the script.

First time execute only after specified interval

Determines when to execute the script for the first time after it gets enabled. If this property is active, the script gets executed only after expiration of the specified time interval. Otherwise, the script gets executed right after it gets enabled.

2.6.11.5 Condition scripts

Condition scripts get executed when the specified logic condition is met including the possibility of periodical repetition at the specified time interval.

Compare tag value with

Constant	Determines that the value of the tag (specified by the *Tag* property) should be compared with the value of a constant (specified by the *Compare with* property).
Another tag	Determines that the value of the tag (specified by the *Tag* property) should be compared with the value of another tag (specified by the *Compare with* property).

Tag	Specifies the link to the tag whose value is to be compared with the value of a constant or another tag (specified by the *Compare with* property).
Condition	Determines the logic condition required for the script to get executed.
Compare with	Specifies the constant or link to the other tag whose value is to be used for comparison.
Execute repeatedly	Determines whether to execute the script only once after the specified logic condition gets met, or periodically at the specified time interval while the specified logic condition is still met.

2.6.11.6 Event scripts

Event scripts get executed when a certain event occurs (e.g. when a *Button* component is clicked).

2.6.12 User Manager

The ***User Manager*** allows you to define and configure users. In the context of *Reliance*, a user is an object representing an end user (usually operator) allowed to log on to the program.

User properties

Basic

Name	Specifies the user's name that must be unique within the project and cannot contain illegal characters. The name is entered by the user when logging on to the program.
Password	Specifies the password that is entered by the user when logging on to the program.

Confirm password — Must have the same value as the *Password* property.

HW code — Specifies the code assigned to the user if logging on by a hardware code sensor is used. Otherwise, the value is ignored.

Active user — Determines whether the user's account is active. It allows you to temporarily enable or disable the user to log on to the program.

Log user log-on — Determines whether to log the information about logging the user on to the program to the alarm/event database.

Log user log-off — Determines whether to log the information about logging the user off of the program to the alarm/event database.

User administrator — Determines whether to allow the user to administrate users at runtime. If the property is active, the user can add new users and modify or delete existing users via the ***User Manager***.

Access rights

These properties specify the user's access rights to the program (see the chapter 2.4.2.3 ACCESS RIGHTS). For example, if the *Control process* access right is required for switching on o pump by a *Button* component, it can only be done by a user(s) who has been assigned this access right.

Servicing right — Determines whether the user has a special right that ensures the user cannot be modified or deleted at runtime via the ***User Manager*** by a user administrator that has not been assigned the *Servicing right*. This right can only be activated through this property in the development environment. This feature enables *Reliance* system integrators to secure certain parts of the application from other users.

Check All — Is used to select all the rights in the list.

Uncheck All	Is used to unselect all the rights in the list.
Edit	Is used to bring up the ***Project Options*** dialog to rename the rights (see the chapter 2.4.2.3 ACCESS RIGHTS).

Restrictions

These properties determine the access restrictions applied when the user logs on to the program. After logging the user off of the program, the default restrictions configured for the computer are restored (see the chapter 2.6.2.2 DEFINING COMPUTERS).

Disable 'Start' menu Determines whether to disable using the Windows ***Start*** menu.

Hide task bar
Determines whether to hide the Windows task bar.

Hide icons on desktop Determines whether to hide icons on the Windows desktop.

Disable minimizing runtime software's main window
Determines whether to disable minimizing the main window of the runtime software.

Disable moving runtime software's main window
Determines whether to disable moving the main window of the runtime software.

Disable printing alarms/events, trends, reports...
Determines whether to disable printing alarms/events, trends and reports. However, this property only applies to printing invoked by the user. It does not affect online printing alarms/events.

Disable editing trends Determines whether to disable editing trends via the ***Trend Manager***.

Disable editing reports Determines whether to disable editing reports via the ***Report Manager***.

2.6.13 Component Manager

The ***Component Manager*** is a tool window that enables you to view and edit properties of components (graphical objects) and visualization windows and select components in the active window.

On the ***Properties*** page, there is a list of component properties. The properties may be sorted alphabetically or divided into groups. If no components are selected in the active window, the page displays the properties of the window itself. If a single component is selected in the active window, the page displays the properties of the component. If multiple components are selected in the active window, the page only displays the properties common to all the selected components. All the displayed properties can be edited.

On the ***Components*** page, there is a list of components contained in the active window. The components in the list can be sorted by one of the columns (*Name*, *Type*, *Order*, *Group* and *Layer*). The list allows you to select or unselect a component in the window by toggling the checkbox next to the component's name. It is especially useful when you need to select or unselect a component hidden by another overlying component.

Groups	Determines whether to arrange the properties into groups. If this option is not active, the properties are sorted alphabetically.
Expand	Is used to expand the groups of properties.
Collapse	Is used to collapse the groups of properties.
Aliases	Determines whether to display the localized names (aliases dependent on the language version of the development environment) of properties instead of the programmatic names.
Help	Determines whether to display a brief description of the selected property in the bottom section of the ***Properties*** page.

Collection editor

Some components have properties of a special type called a collection. A collection is a list of items, each of which has its own properties. To view or edit a collection-type property, use the *Collection editor*.

An example of a collection-type property is the *States* property of the *Active Text* component. The *States* property represents a list of texts that can be displayed by the component depending on the value of a control tag. To bring up the *Collection editor* for this property, select the component and double-click the corresponding cell in the ***Component Manager*** or click the ellipsis button in this cell.

The *Collection editor* allows you to add and delete the items and change their order in the list. If you select one or more items in the list, you can view and edit their properties via the ***Component Manager***.

2.6.14 Window Manager

The ***Window Manager*** is a tool window that enables you to manage visualization windows (i.e. open, activate and close the windows, remove the windows from a project, add new or existing windows to a project, etc.). The ***Window Manager*** contains a toolbar, a status bar and a list of visualization windows. The toolbar contains several buttons representing commands described later in this chapter. The status bar displays the ID and name of the window currently selected in the list. The list of visualization windows displays information in two columns (*Name* and *Title*) and can be sorted by one of the columns. Each row in the list represents a visualization window. A visualization window can be in one of two states: opened (i.e. loaded into memory) and closed (i.e. not loaded into memory). At design-time, these states are indicated by the font style used to draw the row's text (opened: bold font, closed: regular font). The icon displayed at the beginning of each row indicates the window's type (standard, dialog, tray, template) and the way of loading into memory (dynamic or not).

New Folder	Is used to create a new window folder.
New Window	Is used to create a new window.
New Window Template	Is used to create a new window template. A window template is a special kind of window that can be embedded into normal windows through the *Container* component.
Open	Is used to open the window (i.e. load the window into memory and activate it).
Delete	Is used to remove the window from the project. By default, *Reliance* also deletes the window's file and picture database after removing the window.

2.6.15 Layer Manager

At design-time, each component in a visualization window is located on a certain layer. Each visualization window has 16 layers named *Layer0* to

Layer15 by default. The system of layers gives one more dimension to a visualization window.

The ***Layer Manager*** is a tool window located usually in the bottom right corner of the screen. The ***Layer Manager*** contains a toolbar, a status bar and a list of layers of the active visualization window. The toolbar contains several buttons representing options and commands described later in this chapter. The status bar displays the name of the layer currently selected in the list.

Visible	Determines whether the layer selected in the list is visible. If this option is active (button pressed), all the components located on the layer are visible at design-time.
Locked	Determines whether the layer selected in the list is locked. If this option is active (button pressed), the position and size of all the components located on the layer are locked.
Rename	Is used to rename the layer selected in the list. It is useful to rename the layer so that the new name describes the components located on the layer.

2.7 INFORMATION WINDOW

The ***Information Window*** is a tool window located at the bottom of the screen. When opening and closing a visualization project, the ***Information Window*** displays the current status of the operation. If the project is open and the mouse cursor is placed above a visualization window, the ***Information Window*** displays the current cursor position (in pixels) and basic properties of the component under the cursor (name, layer, order, position, size, angle, link to the main tag, scripts, actions, etc.).

2.8 STANDARD DIALOG BOXES

2.8.1 Color selection dialog box

This is a *Reliance*-defined dialog box for selecting colors. You can choose a basic color or you can define your own custom colors. The custom colors get stored with the visualization project.

2.8.2 Font selection dialog box

This is the standard Windows dialog box for selecting fonts.

2.8.3 Selection dialog box

This is a *Reliance*-defined multiple-purpose selection dialog box. It is used for selecting one or more objects of a certain type. It contains a toolbar, a status bar, an edit box with a drop-down list, and a list of objects. The toolbar contains several buttons representing options and commands described later in this chapter. The status bar displays the name of the object currently selected in the list.

Up	Is used to display the objects on the immediate superior level.
Set Filter	Is used to apply the filter specified in the edit box to restrict the objects displayed in the list. You can either type the filter directly in the edit box or select it from the drop-down list. The filter supports wild card characters "?" (representing any single character) and "*" (representing any character string).
Cancel Filter	Is used to cancel the applied filter.

Folders	Determines whether to display folders in the list. If this option is active and the objects (e.g. tags) are arranged in folders, the list also displays the folders.
View	Is used to switch between different view styles.
Manager	Is used to bring up the respective manager (e.g. the ***Picture Manager*** from the ***Select Picture*** dialog box).
Show or Hide Tags	This command displays a drop-down menu that contains a list of data types. It enables you to filter the list of tags according to data type by deactivating particular types. This command is only available when selecting a tag.

2.8.4 Access rights selection dialog box

This is a *Reliance*-defined dialog box used for selecting a set of access rights required for a certain operation (e.g. alarm/event acknowledgment).

Servicing right	Determines whether to include the *Servicing right* in the specified set of rights.
Check All	Is used to select all the rights in the list.
Uncheck All	Is used to unselect all the rights in the list.
Edit...	Is used to bring up the ***Project Options*** dialog to rename the rights (see the chapter 2.4.2.3 ACCESS RIGHTS).

2.9 COMPONENTS

Components are the building blocks of every *Reliance* visualization project and are used to design visualization windows. After adding the appropriate components to a visualization window, you need to configure their properties. To configure the properties of multiple components at the same time, you need to use the ***Component Manager***. To configure the properties of a single component, you can also use the component's property editor (double-click on the component or choose the ***Component Properties...*** command from the component's popup menu). The properties of individual components differ depending on their type. Therefore, each component type has a corresponding property editor. However, some properties are common to all component types (e.g. position, size, etc.). In each property editor, the properties (i.e. the controls meant for editing the properties) are well arranged in pages according to their function. The following chapter describes the pages common to property editors of several or all component types.

2.9.1 Common component properties

Basic

Name	Specifies the component's name that must be unique within the window (i.e. the visualization window in which the component is placed) and cannot contain illegal characters.
Alias	Specifies an alternative name for the component. In multi-language projects, alias can be localized (i.e. translated into all project languages).
Position	Specifies the co-ordinates (in pixels) of the top left corner of the component. The X co-ordinate (the *X* property) increases from left to right and the Y co-ordinate (the *Y* property) from top to bottom.
Size	Specifies the width and height (in pixels) of the component.
Angle	Specifies the rotation angle (in degrees) of the component.

Visible Determines whether to display the component during runtime.

Enabled Determines whether to enable the user to work with the component during runtime.

Show hint Determines whether to display the specified help hint when the mouse cursor rests momentarily on the component.

Comment Can be used to specify an optional comment about the component. The comment is intended especially for the author of the visualization project (not the operator).

Description Can be used to specify an optional description of the component. In multi-language projects, the description can be localized (i.e. translated into all project languages).

Dynamic

Visible Determines whether to control the visibility of the component at runtime by the specified tag. If this property is active, the component is only visible, when the value of the control tag is not equal to 0.

Enabled Determines whether to control enabling the component at runtime by the specified tag. If this property is active, the component is only enabled, when the value of the control tag is not equal to 0. Otherwise, the component is disabled, i.e. the user cannot work with it.

X Determines whether to control the horizontal position (the X co-ordinate) of the component at runtime by the specified tag. The control tag determines the relative position (an increment/decrement in the position relative to the

design-time position) or the absolute position. This depends on the state of the *Coordinates* option which can be configured via the ***Project Options*** dialog (*Project* > *Options* > *Components* > *Dynamic properties* > *Position*).

Y — Determines whether to control the vertical position (the Y co-ordinate) of the component at runtime by the specified tag. The control tag determines the relative position (an increment/decrement in the position relative to the design-time position) or the absolute position. This depends on the state of the *Coordinates* option which can be configured via the ***Project Options*** dialog (*Project* > *Options* > *Components* > *Dynamic properties* > *Position*).

Width — Determines whether to control the width of the component at runtime by the specified tag. The control tag determines the relative position (an increment/decrement in the position relative to the design-time position) or the absolute position. This depends on the state of the *Coordinates* option which can be configured via the ***Project Options*** dialog (*Project* > *Options* > *Components* > *Dynamic properties* > *Size*).

Height — Determines whether to control the height of the component at runtime by the specified tag. The control tag determines the relative position (an increment/decrement in the position relative to the design-time position) or the absolute position. This depends on the state of the *Coordinates* option which can be configured via the ***Project Options*** dialog (*Project* > *Options* > *Components* > *Dynamic properties* > *Size*).

Angle — Determines whether to control the rotation angle (in degrees) of the component at runtime by the specified tag. The control tag determines the relative angle (an increment/decrement in the angle relative to the design-time angle) or the

absolute angle. This depends on the state of the *Angle* option which can be configured via the ***Project Options*** dialog (*Project* > *Options* > *Components* > *Dynamic properties* > *Rotation*).

Menu

These properties enable you to specify the popup menus to be shown when clicking individual mouse buttons on the component at runtime.

Scripts/Actions

These properties enable you to specify the scripts and/or actions to be executed when clicking or double-clicking individual mouse buttons on the component. You can also specify the parameters to pass to the scripts.

2.10 TIPS AND TRICKS

This chapter provides several tips and tricks to help you develop your application more easily and efficiently.

2.10.1 Adding multiple components of the same type to a window

To add multiple components of the same type to a visualization window, press the *Shift* key while selecting the component on the component palette. Then click the left mouse button on the window area to place the component as many times as needed. To exit this mode, click the arrow icon on the component palette.

2.10.2 Fine moving and sizing components

To slightly move the components currently selected in a visualization window, press and hold down the *Ctrl* key while pressing the arrow key representing the direction in which you want to move the components.

To slightly resize the components currently selected in a visualization window, press and hold down the *Shift* key while pressing the arrow key representing the direction in which you want to resize the components.

2.10.3 Selecting multiple components

To invert the selection of a component in a visualization window, press and hold down the *Shift* key while clicking on the component.

2.10.4 Defining a link to an object

When invoking a selection dialog to define a link to an object, such as a tag or item, you can influence which object in the list is initially selected (by default, it is the object defined by the existing link). To preselect the object most recently selected via the selection dialog, press and hold down the *Shift* key, while invoking the dialog. To preselect the first object on the top level of the object hierarchy, press and hold down the *Ctrl* key, while invoking the dialog.

2.10.5 Starting a project automatically after turning on a computer

Sometimes it is desired to start the visualization project automatically after turning on a computer. First, configure the operating system's profile of the user on whose behalf the project is to be running, so that it starts the project automatically after user log-on. Then, configure the operating system, so that it automatically logs on the user.

Starting a project automatically after user log-on

1. Create a shortcut to the project on the Windows desktop via the ***Create Shortcut to Project*** dialog (see the chapter 2.4.4 CREATING A SHORTCUT TO A PROJECT). Specify the runtime software (*Reliance View*, *Reliance Control*, *Reliance Control Server*) as the value of the *Software* option.
2. Move the shortcut to the Windows »Startup« folder.

Automatic user log-on in Windows NT/2000

Windows NT/2000 operating systems can be configured to automatically log on the specified user. The user must have administrator access rights and a password longer than 4 characters.

1. Start the *regedt32.exe* (located in the »System32« subdirectory) or *regedit.exe* program.
2. Find the *DefaultDomainName*, *DefaultUserName*, *DefaultPassword* keys in *HKEY_LOCAL_MACHINE|SOFTWARE|Microsoft|Windows NT|CurrentVersion|Winlogon* (if the *DefaultPassword* key does not exist, it must be created with the data type *REG_SZ*), and supply information on the user as values of those keys (domain, name, password).
3. Enter "1" as the value of the *AutoAdminLogon* key (if the key does not exist, it must be created with the data type *REG_SZ*).
4. Save changes and restart the system. If everything is specified correctly, the operating system logs on the specified user automatically after restarting.

WARNING If the user's password is blank or shorter than 4 characters, the operating system logs on the user automatically after restarting only for the first time. Then the system changes the *AutoAdminLogon* key's value to "0", thus deactivating the automatic user log-on feature.

2.10.6 Safe termination of a project during power failure

If the computer running the visualization project is powered by a UPS (Uninterruptible Power Supply), it is recommended to configure the UPS driver to safely terminate the project before shutting down the operating system in case of a long-term power failure. First, it is necessary to copy the program *R_Termin.exe* (located in the »Utils« directory on the installation CD) to the hard disk. Then configure the UPS driver to run the program before shutting down the operating system. This ensures safe termination of the visualization project.

2.10.7 Optimization of graphic efficiency

This chapter contains information that can help you develop applications that are more graphically efficient and do not uselessly overburden the computer. The main goal is to minimize the computer's load caused by drawing graphics in a *Reliance* application.

Each picture used in a project should have its color depth and total size optimized. Ideally, you should edit the pictures in a graphic editor that enables you to save the pictures only with colors used. The graphics format of the picture (e.g. **.bmp*, **.gif*, **.jpg*, etc.) does not directly affect the application's performance. However, it affects the disk space occupied by the picture.

Windows raster (**.bmp*) This is an uncompressed format suitable for small pictures only. The main disadvantage is a direct dependency of file size on picture size.

CompuServe raster (**.gif*) This is a compressed format suitable for pictures with small number of colors (the format supports 256 colors only) and for pictures used as transparent.

JPEG raster (**.jpg*) This is a compressed format suitable, for example, for photographs. The format is not suitable for pictures used as transparent.

Windows metafile (**.wmf*), *extended Windows metafile* (**.emf*)
These are vector formats suitable, for example, for technical schemes.

These are several factors that affect the computer's load caused by drawing graphics in a *Reliance* application:

- Animation speed (applies to *Animation* components): It is controlled by a component's *Interval* property, which specifies the time interval of switching between the pictures. The shorter the interval, the higher the animation speed, the heavier graphical load.
- Picture transparency (applies to all the components that draw pictures): It is controlled by a picture's *Transparent* property which determines whether the picture's background should be considered transparent. Drawing pictures as transparent increases graphical load. It is recommended to avoid using transparent pictures whenever possible (especially with *Animation* components). Instead, use the same color for the picture's background as used for the window's background.
- Picture display mode (applies to all the components that draw pictures): It is controlled by a component's *Layout* property which affects whether the picture is drawn in its original size or not. Drawing pictures as stretched or compressed increases graphical load. It is strongly recommended to use pictures in their original size whenever possible (especially with *Animation* components).
- Relative position of components: If multiple components overlap with one another, graphical load increases. If one of the components needs to be redrawn, it causes all the overlapping components to be redrawn, too. It is recommended to avoid overlapping components that are redrawn frequently (especially *Animation* components) with components whose drawing is time-consuming (e.g. a *Picture* component displaying a large picture).

2.10.8 Optimization of communications with I/O devices

This chapter contains information that can help you develop applications with efficient communications with I/O devices (e.g. PLCs). The main goal is to minimize the computer's load caused by the communications and reduce the time necessary to transfer the latest data from the devices to the application.

When assigning an address to tags in a device (e.g. PLC), try to concentrate the tags with identical update interval (time interval used by the communication driver to periodically update its image of the device's memory) into a continuous memory area.

When defining communication zones for a device, try to cover as many tags (with identical update interval) as possible with a single zone. As for the communications, it is much more efficient to define a small number of long zones than a large number of short zones. Sometimes, it is better to define a long zone, even if it covers unused memory areas, than define multiple shorter zones.

3. APPENDIXES

3.1 ILLEGAL CHARACTERS

The names of non-visual objects (e.g. devices, tags, data tables, etc.) and components (graphical objects) cannot contain the character to be used to separate name parts when building the complete name of an object (see the chapter 2.4.2.10 OBJECTS).

When naming files and objects of any kind in *Reliance* visualization projects, it is recommended to avoid using the following characters:

Special characters:

! ? @ # $ % ^ & * ( ) + - ˇ % { } / \ , „ < > = ; : ’ | § .

Letters with diacritics:

ě š č ř ž ý á í é ů ú ď ť ň Ě Š Č Ř Ž Ý Á Í É Ů Ú Ď Ť Ň

3.2 FILE AND DIRECTORY STRUCTURE

3.2.1 Reliance main program files

Install directory

(by default, »c:\Program Files\GEOVAP\Reliance4«)

R_Design.exe – *Reliance Design*
R_View.exe – *Reliance View*
R_Ctl.exe – *Reliance Control*
R_CtlSrv.exe – *Reliance Control Server*
R_Srv.exe – *Reliance Server*

3.3 POINT RATING OF RELIANCE VISUALIZATION PROJECTS

The size of a visualization project is one of the main factors affecting the price of the license needed to develop the project and run it at the end user site. The size is determined by the number of data points used in the project. The number of data points depends on the number and data type of tags defined in the project.

The following rules hold true:

- Tags defined within the *System* device do not affect the number of data points.
- Each tag of a simple data type (e.g. *Bool*, *Byte*, *Word*, *String*, etc.) uses one data point.
- Each array-type tag uses one or more data points. If the element count is less than or equal to five, the tag uses one data point. Otherwise, the number of data points is equal to the element count divided by five (the result is rounded in the direction of zero to the nearest integer).
- The number of data points at design-time and runtime can differ. At design-time, all the devices defined via the ***Device Manager*** are considered. At runtime, only the devices connected to a computer via the ***Project Structure Manager*** are considered.

To view the number of data points used in a visualization project, open the project in the development environment and choose the ***Project*** ▸ ***Information...*** command. This will bring up the ***Project Information*** dialog.