TABLE OF CONTENTS

3. Technological data distribution over the Internet/intranet

4. Appendixes

VERSION 3.3 – NOVEMBER 2002

1. INTRODUCTION

1.1 ABOUT RELIANCE

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

Reliance 3 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 before.

1.2 RELIANCE 3 STRUCTURE

Reliance 3 consists of several SW modules described briefly in the following chapters.

1.2.1 Development environment: Reliance design

The development environment is a program designed for creating visualizing projects.

- *Reliance design DESKTOP* is a basic version of the program meant for developing applications without the possibility to distribute the data across a network.
- *Reliance design ENTERPRISE* is an enhanced version of the program that includes the ability to develop network applications.

1.2.2 Runtime modules: Reliance runtime, Reliance server, Reliance runtime server

The runtime module is a program designed for running a visualizing project on the end user's computer. The runtime module is a common term for the following programs: *Reliance runtime*, *Reliance server* and *Reliance runtime server*. These programs have an identical core and some basic functions – in particular communication to drivers of the connected stations, network communication to other runtime modules, logging data and alarms, executing scripts and recipes.

Reliance runtime

In addition to the basic functions of a runtime module, *Reliance runtime* allows the graphic display of technological data and alarms (current and historical), creation of printed outputs (reports or trends), user administration and logging on and off the program.

- *Reliance runtime VIEW* is a "view only" version of the program. It does not allow users to send commands to the technology.
- *Reliance runtime CONTROL* is a fully functional version of the program that allows users to send commands to the technology.

Reliance server

In addition to the basic functions of a runtime module, *Reliance server* is able to distribute technological data and alarms (current and historical) to *Reliance J* Web clients and execute the commands received from these clients. Therefore, *Reliance server* can be used as a data server for the Web clients.

Reliance runtime server

In addition to the basic functions of a runtime module, *Reliance runtime server* provides all the functions of the *Reliance runtime* and *Reliance server* programs.

1.2.3 Web client: Reliance J

The *Reliance J* Web client is a Java applet designed for running a visualizing project using a standard Web browser that supports the Java programming language (e.g. Microsoft Internet Explorer 4 and higher, Netscape Communicator 4 and higher). The Web client provides a powerful and easy way to access the technology from remote locations over the Internet/intranet. For more information on this topic, see the chapter 3. TECHNOLOGICAL DATA DISTRIBUTION OVER THE INTERNET/INTRANET.

1.2.4 Communication drivers

Communication drivers ensure the communication to technological stations (e.g. PLCs) and other I/O devices. *Reliance* includes native communication drivers for many types of I/O devices, e.g. *TECO*, *Allen Bradley*, *Modicon* and others. *Reliance* is also an *OPC client*, i.e. it is able to communicate to any I/O device for which an *OPC server* is available. *Reliance* also supports the GSM technology (sending and receiving SMS messages).

1.3 BASIC PROGRAM FUNCTIONS

- monitoring and controlling processes in real time
- archiving selected variables to databases
- creating and displaying trends of selected variables
- creating and displaying reports and print reports
- defining, archiving and displaying alarms
- defining users, access rights and restrictions
- multimedia support
- technological data distribution over the Internet/intranet
- Postmort, i.e. replaying the operation of the monitored process in a retrospective way
- ActiveX controls support
- multiple monitor system support
- verification of users by a fingerprint sensor
- recipe control support

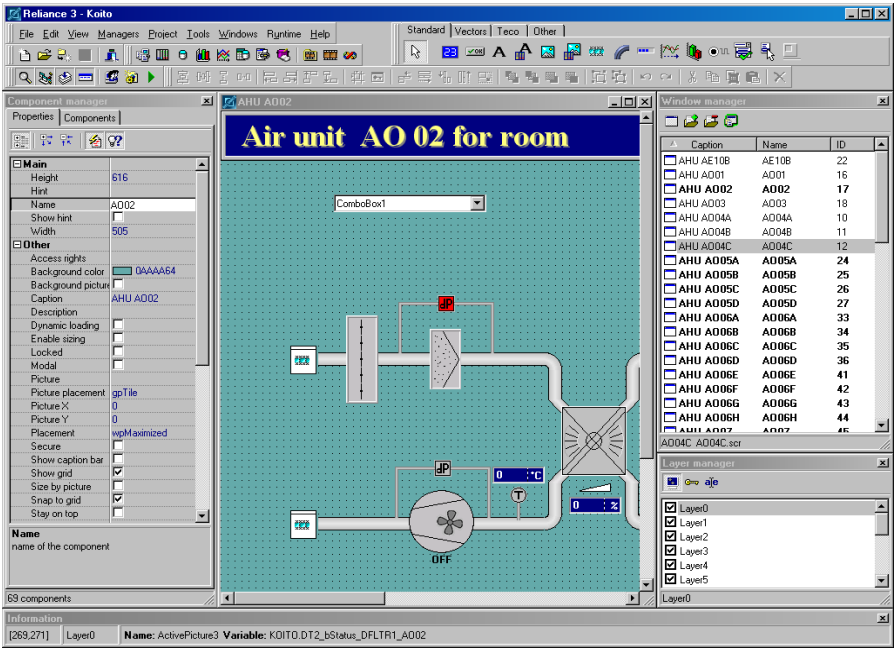
1.4 HW AND SW REQUIREMENTS

Reliance 3 can be operated on computers running MS Windows XP/2000/NT/9x/ME operating systems. However, MS Windows 9x/ME operating systems are not recommended. To develop medium-sized applications (c. 2000 variables), we recommend the following minimum HW configuration: PC Pentium II, 350 MHz, Windows 2000, 64 MB RAM, HD 2 GB, 8bit graphic adapter, resolution 800 × 600, parallel or USB port. The complete installation of *Reliance 3* requires c. 30 MB of free hard disk space.

2. DESCRIPTION OF THE DEVELOPMENT ENVIRONMENT

The *Reliance design* development environment is a program designed for creating visualizing 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.

When starting the program for the first time, you will be asked whether you want to associate the files with the *.prj* extension (*Reliance* visualizing project main files) with the program. This operation is called the registration of a file type in the operating system and allows opening files of that type by the specified program without needing to run the program in advance.

Registration of the files with the *.prj* extension can also be done later via the ***Environment options*** dialog (see the chapter 2.3.3.5 FILE TYPES).

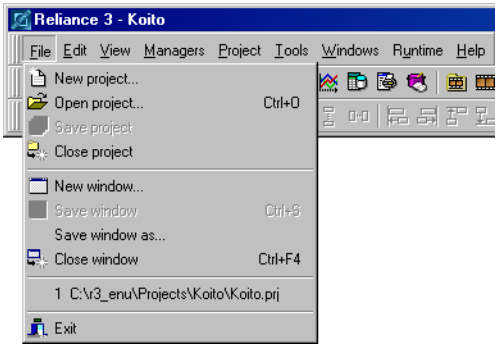
Upon registration, the files with the *.prj* extension are marked as *Reliance Project* and they are assigned a specific icon. Thus, for example, a visualizing project can be opened by double-clicking on the shortcut to the main file on the Windows desktop.

2.2 DESCRIPTION OF MENU FUNCTIONS

This chapter describes commands that are available in the main menu. The detailed description of a command may include a picture and 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 visualizing projects and project windows. The menu also contains a list of most recently open projects.



New project... This command is used to create a new visualizing project (see the chapter 2.4.1 CREATING A NEW PROJECT).

Open project... (*Ctrl+O*) This command is used to open an existing visualizing project.

Close project This command is used to close the open visualizing project.

New window... This command is used to create a new window.

Save window (*Ctrl+S*) This command is used to save the active window.

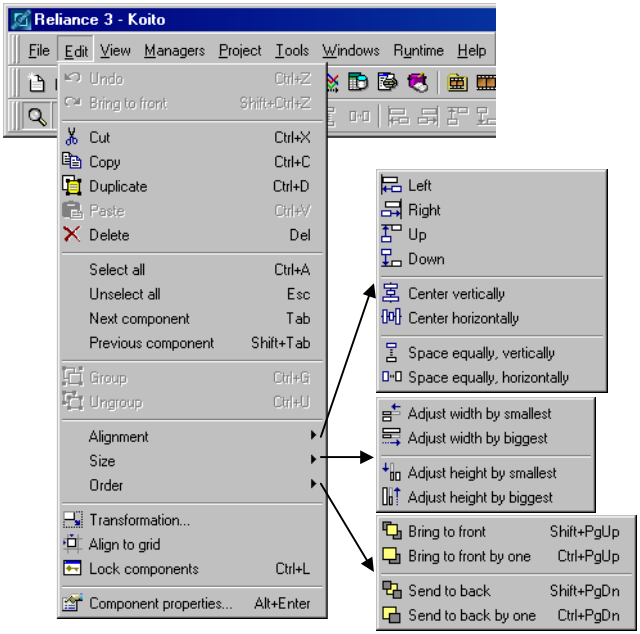
Save window as... This command is used to save the active window using a new name.

Close window (*Ctrl+F4*) This command is used to close the active window.

Exit This command 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 the command is to be applied and then select the command from the menu.



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

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

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

Copy (*Ctrl+C*) — This command is used to copy the currently selected component(s) to the clipboard.

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

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

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

Select all (*Ctrl+A*) This command is used to select all the components in the active window.

Unselect all (*Esc*) This command is used to unselect all the currently selected components in the active window.

Next component (*Tab*) This command is used to select the next component in the Z-order.

Previous component (*Shift+Tab*) This command is used to select the previous component in the Z-order.

Group (*Ctrl+G*) This command 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*) This command is used to break the currently selected group into individual components.

Alignment

Left This command is used to align the left edge of all selected components with the left edge of the leftmost selected component.

Right This command is used to align the right edge of all selected components with the right edge of the rightmost selected component.

Top This command is used to align the top edge of all selected components with the top edge of the topmost selected component.

Bottom This command is used to align the bottom edge of all selected components with the bottom edge of the bottommost selected component.

Center vertically This command is used to align the vertical centerline of all selected components with the centerline of the group of components selected.

Center horizontally This command is used to align the horizontal centerline of all selected components with the centerline of the group of components selected.

Space equally, vertically

This command is used to equally space all selected components vertically between the topmost selected component and the bottom-most selected component.

Space equally, horizontally

This command is used to equally space all selected components horizontally between the leftmost selected component and the right-most selected component.

Size

Adjust width by smallest

This command is used to change the width of all selected components by the smallest selected component.

Adjust width by largest

This command is used to change the width of all selected components by the largest selected component.

Adjust height by smallest

This command is used to change the height of all selected components by the smallest selected component.

Adjust height by largest

This command is used to change the height of all selected components by the largest selected component.

Order

 ***Bring to front*** (*Shift+PgUp*)

This command 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 the component(s).

 ***Bring to front by one*** (*Ctrl+PgUp*)

This command is used to move the currently selected component to the front by one level. This is called changing the Z-order of the component.

 ***Send to back*** (*Shift+PgDn*)

This command 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 the component(s).

 ***Send to back by one*** (*Ctrl+PgDn*)

This command is used to move the currently selected component to the back by one level. This is called changing the Z-order of the component.

 ***Transformation...*** This command is used to proportionally resize or (and) move the currently selected component(s).

 ***Align to grid*** This command is used to align the currently selected component(s) to the closest grid point.

 ***Lock components*** (*Ctrl+L*)

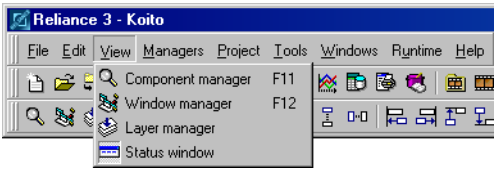
This command is used to lock the position and size of all the components in the window.

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

This command 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*)
This command is used to show/hide the ***Component manager*** (see the chapter 2.6.12).

Window manager (*F12*)
This command is used to show/hide the ***Window manager*** (see the chapter 2.6.13).

Layer manager This command is used to show/hide the ***Layer manager*** (see the chapter 2.6.14).

Information window This command is used to show/hide the ***Information window*** (see the chapter 2.7).

2.2.4 Managers menu

Project structure manager
This command is used to bring up the ***Project structure manager*** (see the chapter 2.6.2).

Station manager This command is used to bring up the ***Station manager*** (see the chapter 2.6.3).

Database manager This command is used to bring up the ***Database manager*** (see the chapter 2.6.4).

Trend manager This command is used to bring up the ***Trend manager*** (see the chapter 2.6.5).

Real-time trend manager This command is used to bring up the ***Real-time trend manager*** (see the chapter 2.6.6).

Report manager This command is used to bring up the ***Report manager*** (see the chapter 2.6.7).

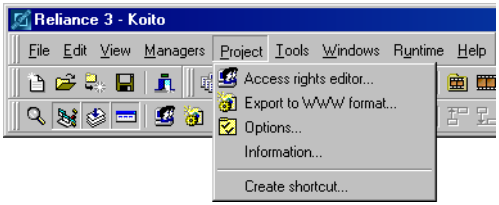
Print report manager This command is used to bring up the ***Print report manager*** (see the chapter 2.6.8).

Recipe manager This command is used to bring up the ***Recipe manager*** (see the chapter 2.6.9).

Shared picture manager This command is used to bring up the ***Shared picture manager*** (see the chapter 2.6.10 PICTURE MANAGER).

Script manager This command is used to bring up the ***Script manager*** (see the chapter 2.6.11).

2.2.5 Project menu



Access rights editor... This command is used to bring up the ***Access rights editor*** (see the chapter 2.4.3).

Export to WWW format...

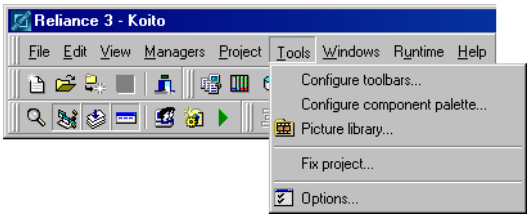
This command is used to bring up the ***WWW format export wizard***.

Options... This command is used to bring up the ***Project options*** dialog.

Information... This command is used to bring up the ***Project information*** dialog. The dialog contains information on the number of variables and data points within the project (see the chapter 4.3 POINT RATING OF RELIANCE VISUALIZING PROJECTS).

Create shortcut... This command is used to bring up the ***Create shortcut to the project*** dialog (see the chapter 2.4.4).

2.2.6 Tools menu



Configure toolbars... This command is used to bring up the ***Configure toolbars*** dialog (see the chapter 2.3.1).

Configure component palette...

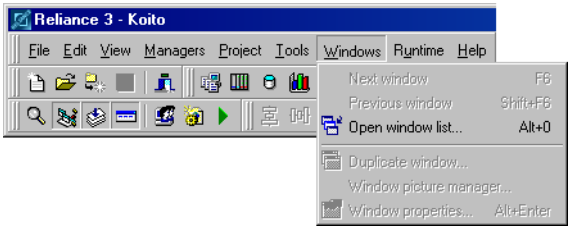
This command is used to bring up the ***Configure component palette*** dialog (see the chapter 2.3.2).

Picture library This command is used bring up the ***Picture manager*** to view or edit the ***Picture library*** (see the chapter 2.6.10 PICTURE MANAGER).

Fix project... This command is used to bring up the ***Fix project*** dialog (see the chapter 2.4.5 FIX PROJECT).

Options... This command is used to bring up the ***Environment options*** dialog (see the chapter 2.3.3 OPTIONS).

2.2.7 Windows menu



Next window (*F6*) This command is used to change the active window to the next open window.

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

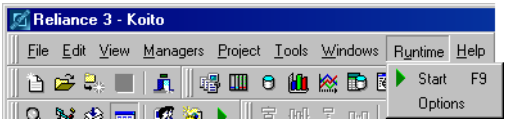
Open window list... (*Alt+0*)
This command is used to bring up the ***Open window list***. Any window in the list can then be selected and activated.

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

Window picture manager...
This command is used to bring up the ***Picture manager*** of the active window (see the chapter 2.6.10 PICTURE MANAGER).

Window properties... (*Alt+Enter*)
This command is used to bring up the property editor of the active window (see the chapter 2.5.3 WINDOW PROPERTIES).

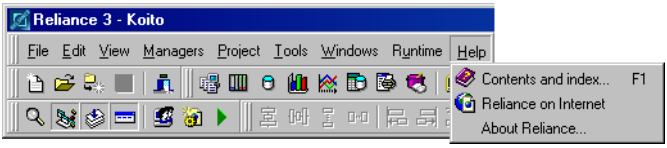
2.2.8 Runtime menu



Start (*F9*) This command is used to start the visualizing project in runtime mode. The development environment runs the runtime module of the specified type and passes the project name and location as parameters.

Options... This command is used to bring up the ***Project options*** dialog (see the chapter 2.4.2.2 RUNTIME MODULE). Here, it is possible to select the type of runtime module to run the visualizing project in runtime mode and configure the parameters affecting the appearance and behavior of the runtime module.

2.2.9 Help menu



Contents and index... (*F1*)
This command is used to access the help system.

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

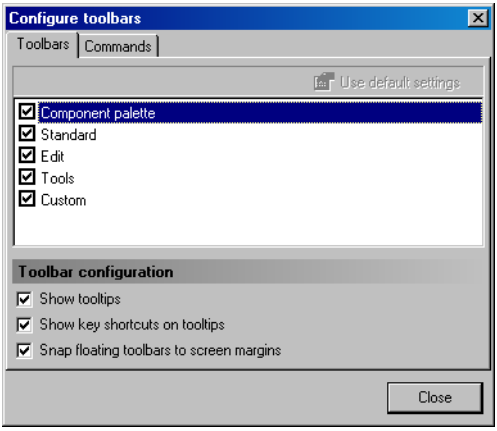
About Reliance... This command is used to view the version and license information for the *Reliance design* program.

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	This toolbar 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	This toolbar is designed for project operation commands.
Edit	This toolbar is designed for commands used when working with components.
Tools	This toolbar is designed for commands for opening individual managers.
Custom	This is a user-defined toolbar.

Toolbar configuration

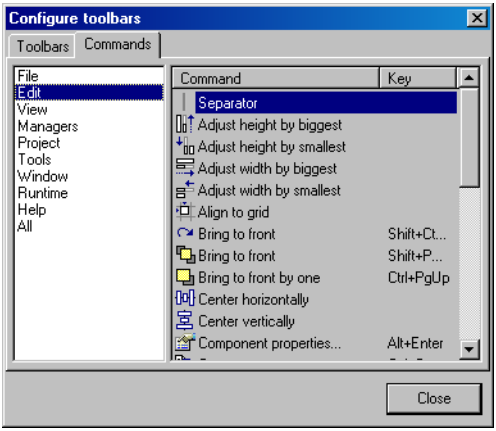
Show tooltips This option determines whether to show a brief description of the command after positioning the mouse cursor over the toolbar button.

Show key shortcuts on tooltips
This option determines whether to show key shortcuts on tooltips.

Snap floating toolbars to screen margins
This option determines whether to attach floating toolbars to screen margins.

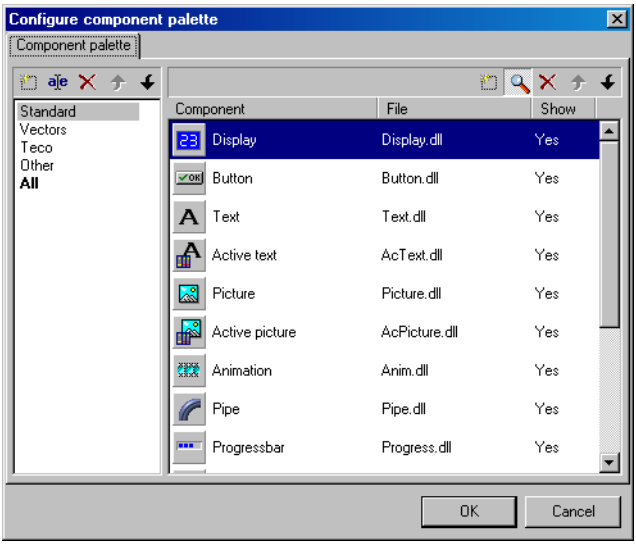
Commands

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 up to the user.



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*, *Teco*, *Sauter* 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 This command is used to add a new page to the component palette, or install a new component from the »Components« directory.

Edit This command is used to rename the page of the component palette.

Show This command is used to show or hide the selected component.

Remove This command is used to remove the selected page from the component palette, or remove the selected component from the page.

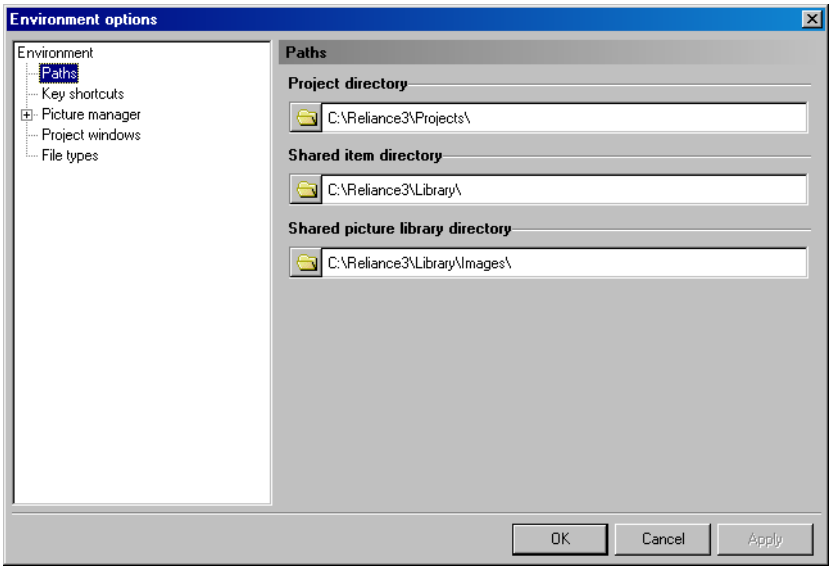
Move up This command is used to move the selected page or selected component up by one position.

Move down This command is used to move the selected page or selected component down by one position.

2.3.3 Options

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

2.3.3.1 Paths

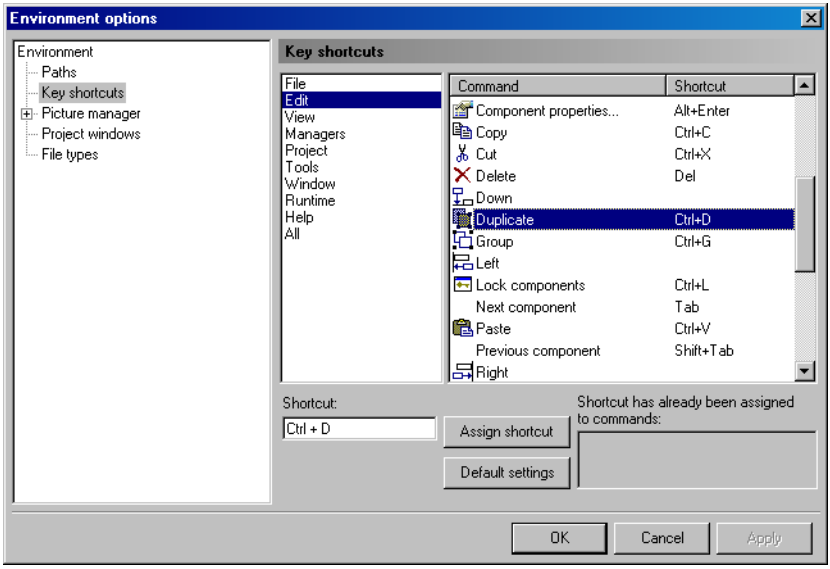


Project directory This parameter specifies the path for storing visualizing projects.

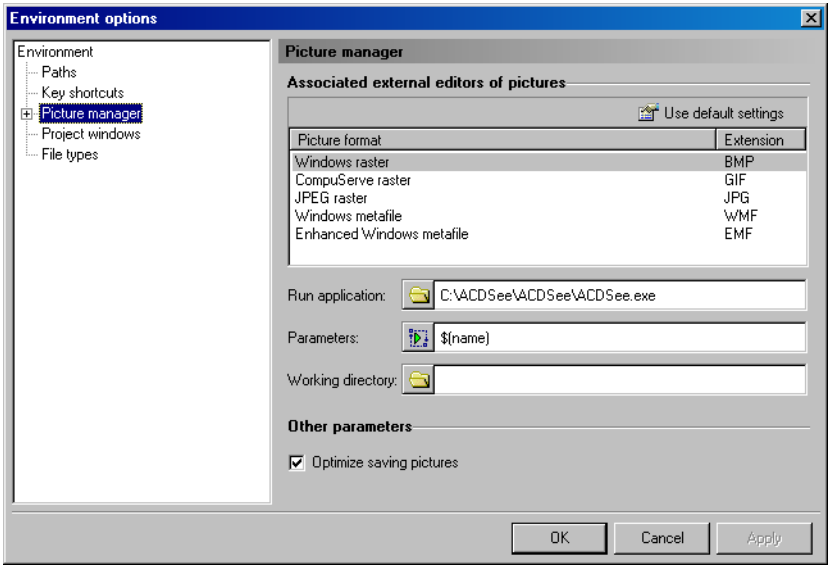
Shared item directory This parameter specifies the path for storing shared items.

Shared picture library directory
This parameter specifies the path for storing the shared picture library.

2.3.3.2 Key shortcuts



2.3.3.3 Picture manager



Associated external editors of pictures

Here it is possible to associate external picture editors with the picture formats supported by *Reliance*. 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, associations are adopted from the Windows registry.

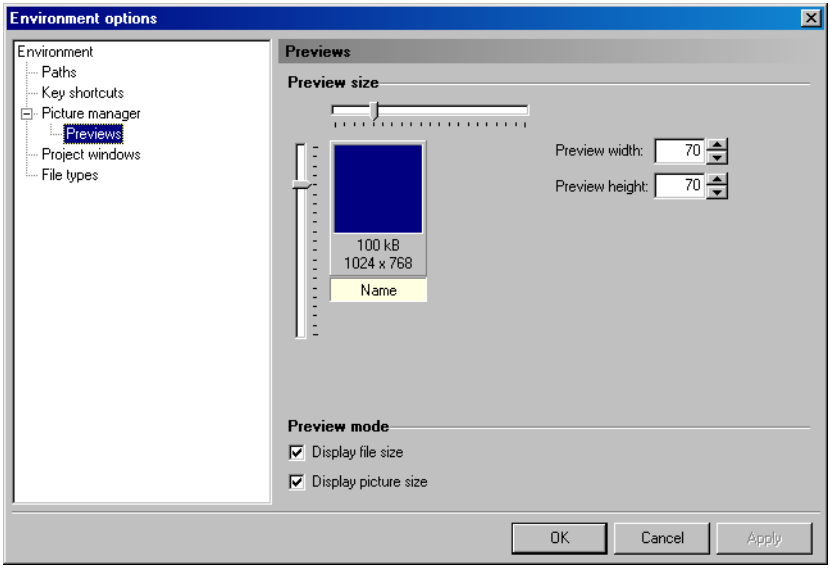
Run application	This parameter specifies the path and name of the picture editor's executable file.
Parameters	This parameter specifies the parameters to pass to the picture editor at startup. By default, *Reliance* passes the picture name as a parameter to the editor.
Working directory	This parameter specifies the path for storing temporary files.

Other parameters

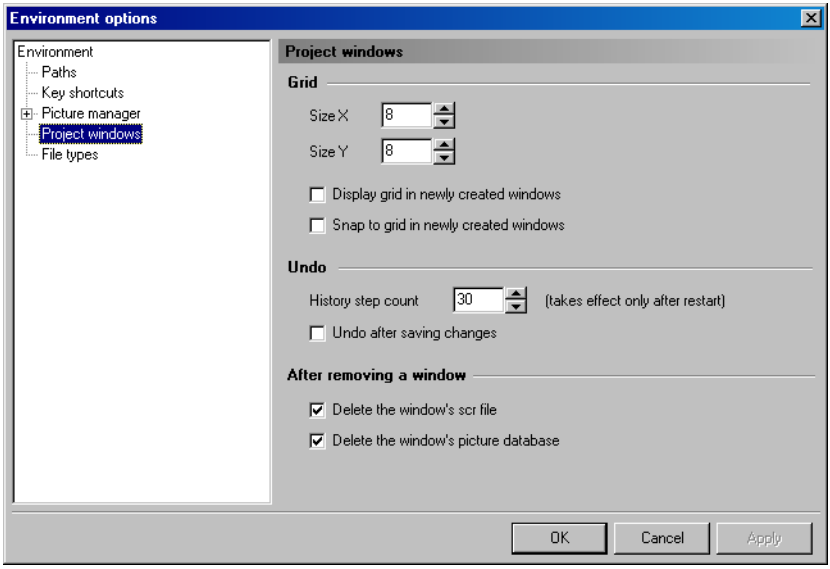
Optimize saving pictures	This option determines whether to optimize the pictures for size when saving to the database.

Previews

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



2.3.3.4 Project windows



Grid

Size X This parameter specifies the grid width.

Size Y This parameter specifies the grid height.

Display grid in newly created windows
This option determines whether to activate the *Display grid* parameter in newly created windows.

Snap to grid in newly created windows
This option determines whether to activate the *Snap to grid* parameter in newly created windows.

Undo

History step count This parameter 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 saving changes
This option determines whether to enable the user to use the *Undo* command after saving the window.

After removing a window

Delete the window's scr file
This option determines whether to delete the window's *scr* file after removing the window from the project.

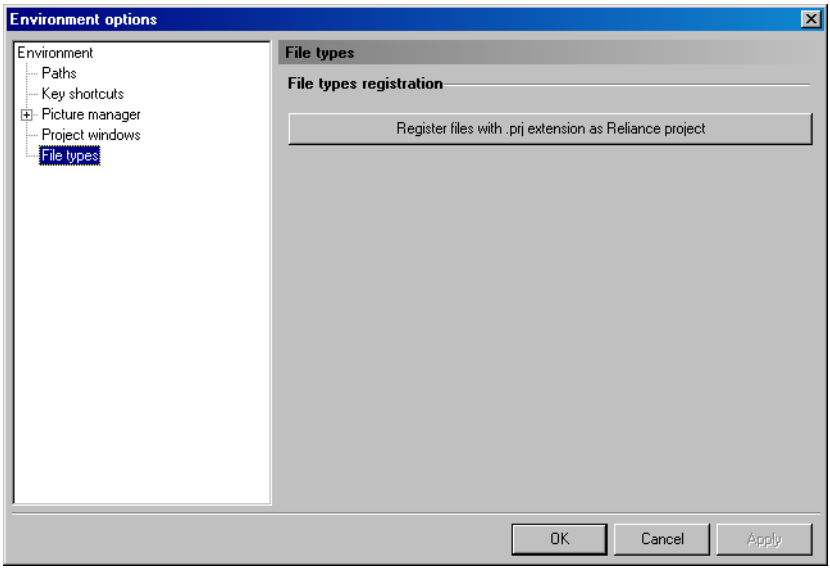
Delete the window's picture database
This option determines whether to delete the window's picture database after removing the window from the project.

2.3.3.5 File types

File type registration

Here, it is possible to associate files with the *.prj* extension with the *Reliance design* development environment.

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

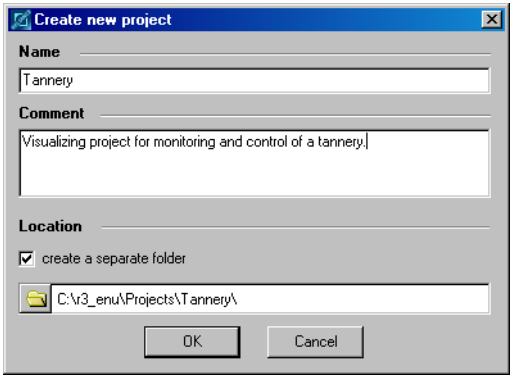


2.4 VISUALIZING PROJECT

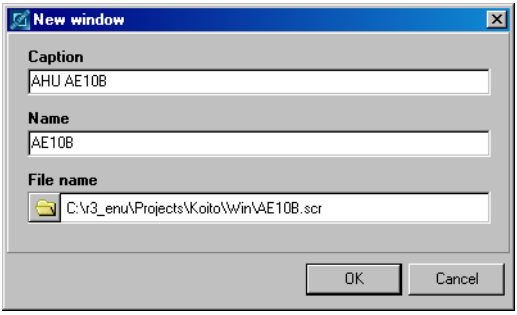
A *Reliance* visualizing project is a group of files stored in a specific directory structure (see the chapter 4.2 FILE AND DIRECTORY STRUCTURE).

2.4.1 Creating a new project

To create a new visualizing project, choose the ▸ ***File*** ▸ ***New project*** command. First, you will be prompted to enter the name, comment and location for the new project.



After the new project is generated, you will be prompted to specify the parameters for the first project window.

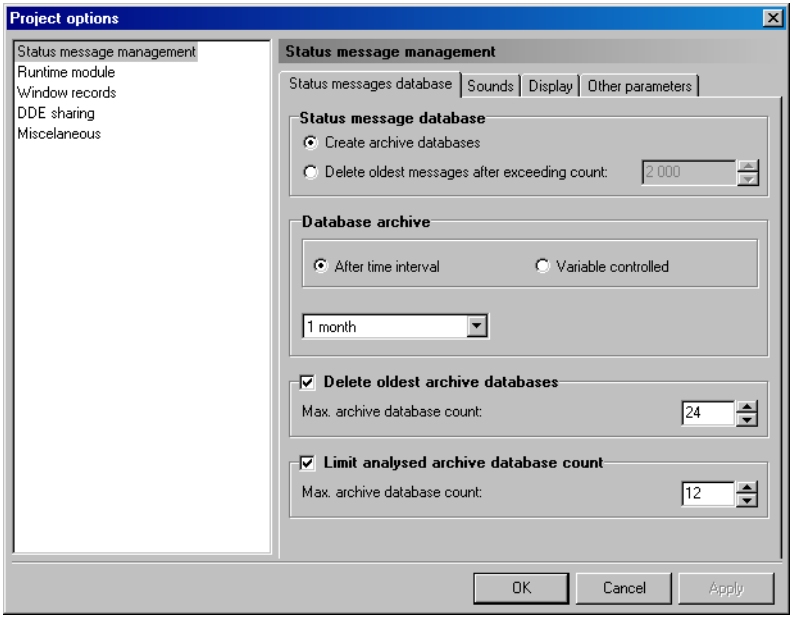


For detailed description of the ***New window*** dialog box see the chapter 2.5.1 CREATING A NEW WINDOW.

2.4.2 Project options

2.4.2.1 Status message management

Status message database



Status message database These options determine whether to create archive databases or maintain the latest messages in the current database.

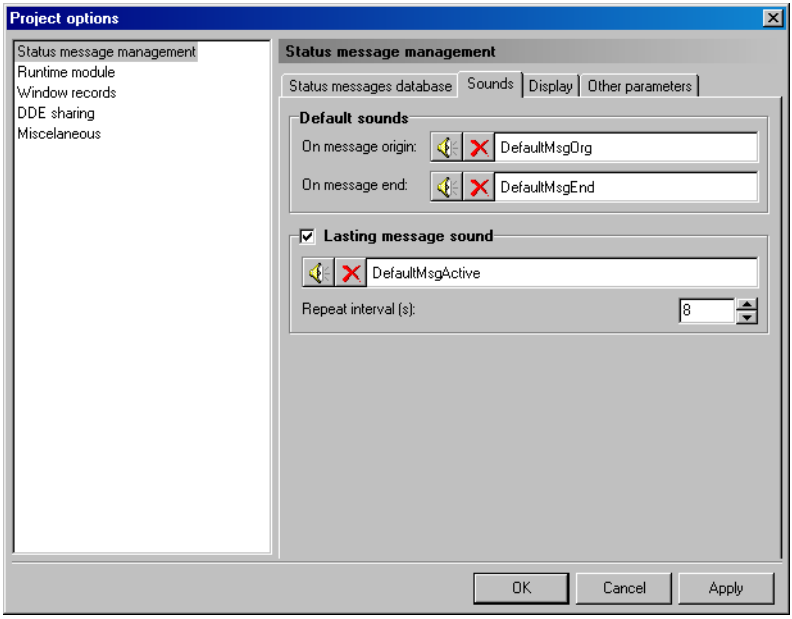
Database archive These options determine the type of archive databases. Creating the archive databases can be either time-controlled or variable-controlled.

Delete oldest archive databases
These options determine whether to delete the oldest databases after exceeding the specified count.

Limit analyzed archive database count
These options determine whether and how to limit the number of archive databases analyzed during starting the project in runtime mode.

Sounds

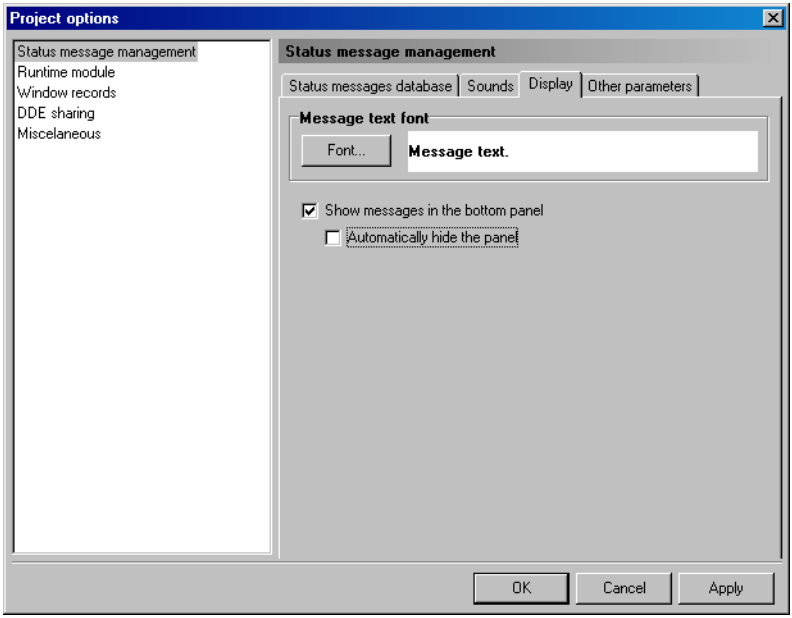
These parameters enable you to configure sounds that can be played when a status message originates, ends or lasts. The sound files must be located in the »MMedia« directory.



Default sounds — These parameters specify the default sounds in the **.wav* format. The default sounds will be used for status messages that do not specify a particular sound.

Lasting message sound — These options determine whether and how often a runtime module should play the specified sound if the current message list contains at least one lasting unacknowledged status message.

Display

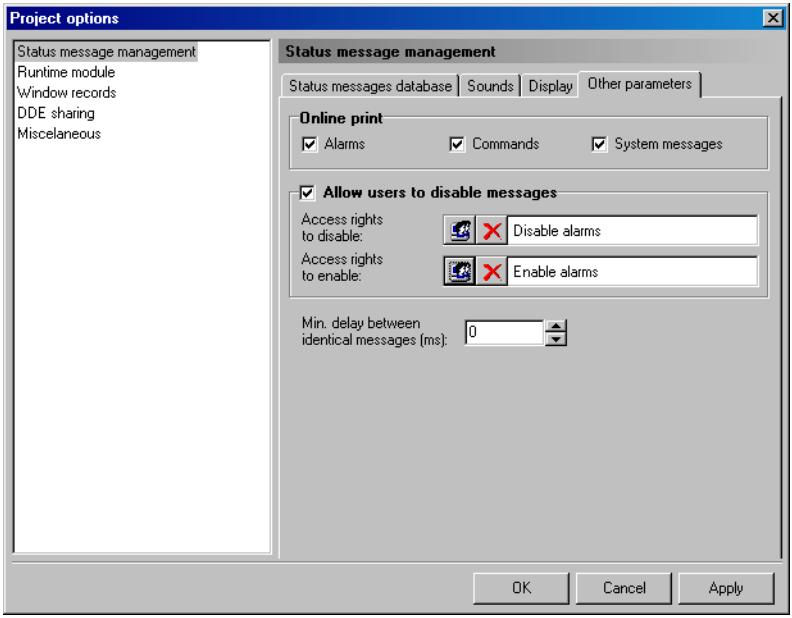


Message text font This parameter determines the font of message texts in runtime mode.

Show messages in the bottom panel
This option determines whether to show a special panel in the bottom part of the runtime module's main window and display status messages in the panel.

Automatically hide the panel
This option determines whether to automatically hide the panel after acknowledging all the messages in the panel.

Other parameters



Online print These options determine whether to enable the online printing of status messages of a certain type in runtime mode.

WARNING Laser printers cannot be used for this purpose!

Allow users to disable messages

This option determines whether to allow users to temporarily disable and enable status messages in runtime mode. 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 from being triggered.

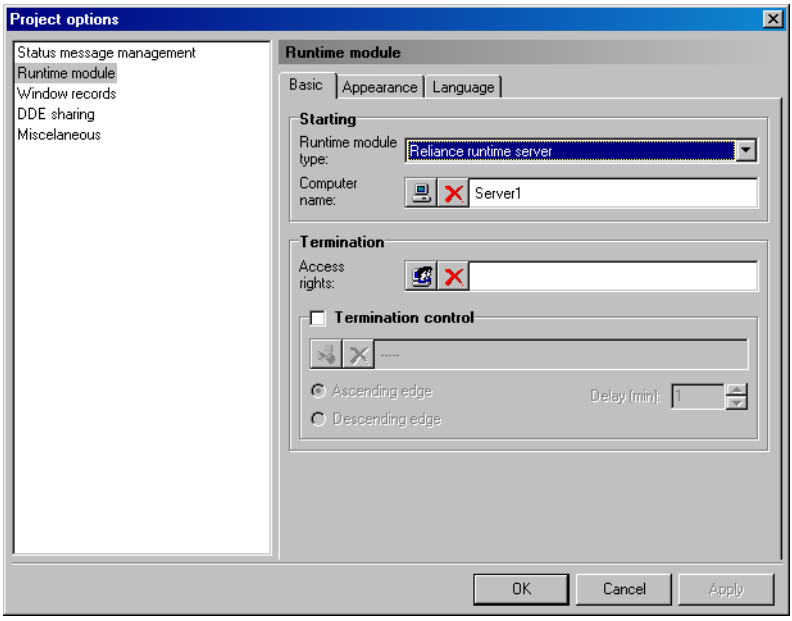
Min. delay between identical messages (ms)

This parameter determines the minimum time period between the moment that the condition triggering a status message ends and the moment

that the condition occurs again to trigger a new instance of the message.

2.4.2.2 Runtime module

Basic



Starting

Runtime module type — This parameter determines the type of runtime module (*Reliance runtime*, *Reliance server*, *Reliance runtime server*) to run the visualizing project in runtime mode when started from the development environment.

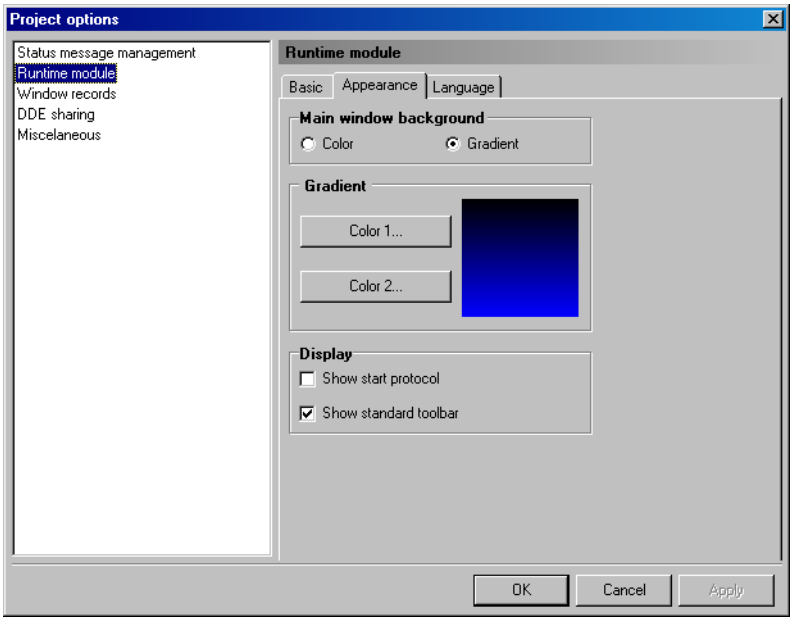
Computer name — This parameter determines the logical computer (defined through the ***Project structure manager***) whose settings (e.g. IP address, system name, connected stations, etc.) will be used by the runtime module when starting the visualizing project from the development environment.

Termination

Access rights This parameter determines the access rights required for terminating the visualizing project running in runtime mode.

Termination control These parameters determine whether the visualizing project running in runtime mode can be terminated owing to changing the value of the specified variable. If the control variable's value changes to the specified value, the runtime module displays an information message regarding the termination. After expiration of the specified delay, the visualizing project is terminated.

Appearance



Main window background

These parameters determine the appearance of the runtime module's main window.

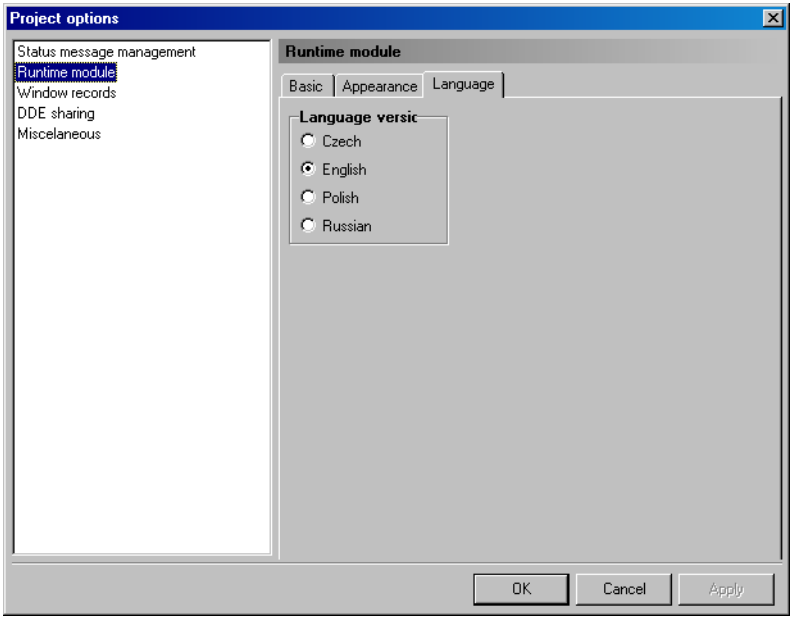
Display

Show start protocol — This option determines whether to show the start protocol while starting and terminating the visualizing project in runtime mode. The protocol contains a description of the operations being performed while starting and terminating the project.

Show standard toolbar — This option determines whether to display the standard toolbar in the runtime module's main window. You can hide the standard toolbar and create your own one.

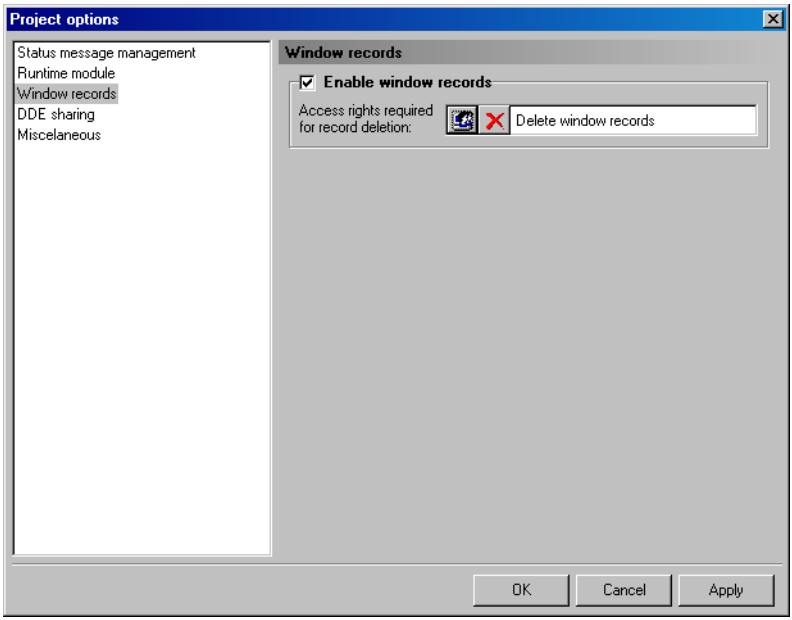
Language

This parameter determines the language version of the visualizing project. The selected language is used for all texts (i.e. menus, toolbars, etc.) in the runtime module. When developing a foreign-language visualizing project, set this parameter appropriately before designing project windows as it also affects the initial setting of components' font.



2.4.2.3 Window records

In runtime mode, it is possible to make operating records for each visualizing project 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 in the window related to the technology.

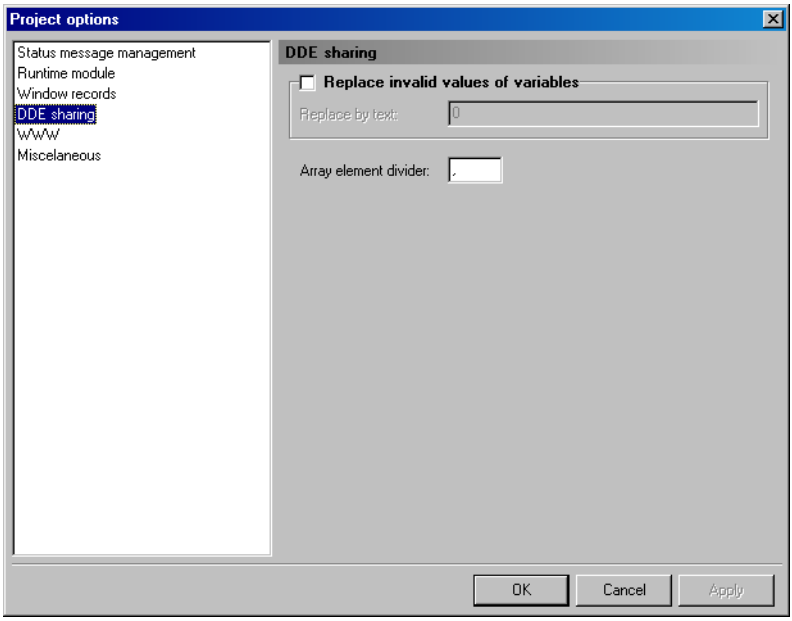


Enable window records This parameter determines whether to allow the users to access window records in runtime mode.

Access rights required for record deletion
This parameter 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.

2.4.2.4 DDE sharing

The runtime module acts as a *DDE server* (see the chapter 2.6.3.3 DEFINING A VARIABLE).



Replace invalid values of variables

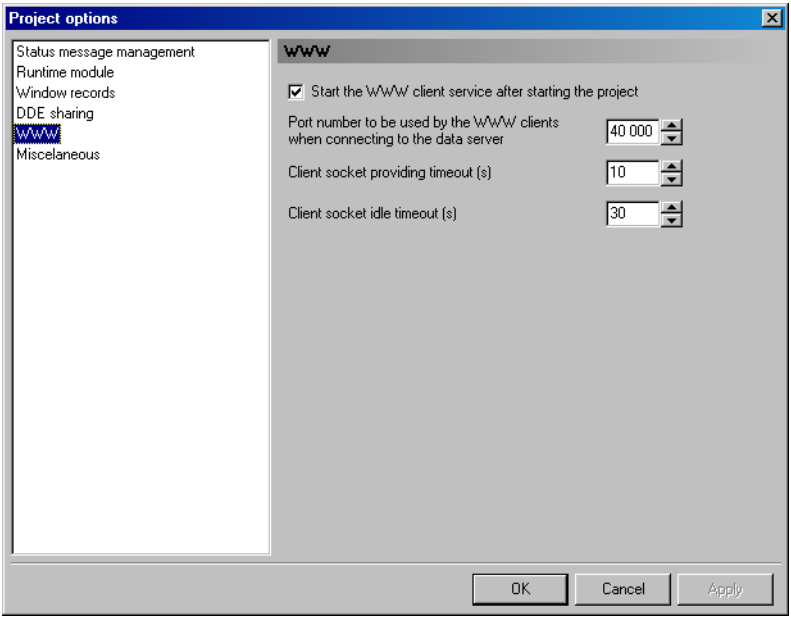
These options determine whether and how to replace the value of a DDE-shared variable when providing it to *DDE client* applications in case that the value is not valid (e.g. because of a failure in communication to the station).

Array element divider

This parameter specifies the text to be used as element divider when sharing an array-type variable.

2.4.2.5 WWW

These parameters enable you to configure the communication between the data server and the *Reliance J* Web clients. Since only *Reliance server* and *Reliance runtime server* can be used as a data server for the Web clients, the parameters have no effect if *Reliance runtime* is used as a runtime module.



Start the WWW client service after starting the project

This option determines whether to start the Web client service after starting the project in runtime mode. If active, the runtime module starts to act as a data server for the Web clients and enables them to connect to it via the TCP/IP protocol.

Port number to be used by the WWW clients when connecting to the data server

This parameter specifies the TCP/IP port number to be used by the Web clients when connecting to the data server. At the same time, the data server uses the specified port number to open a TCP/IP socket in order for the Web clients to connect to

it. When a Web client connects to the data server, it is assigned a new port number. After that, the Web client closes the current TCP/IP socket and opens a new one using the new port number. It is recommended to use the default value.

Client socket providing timeout (s)

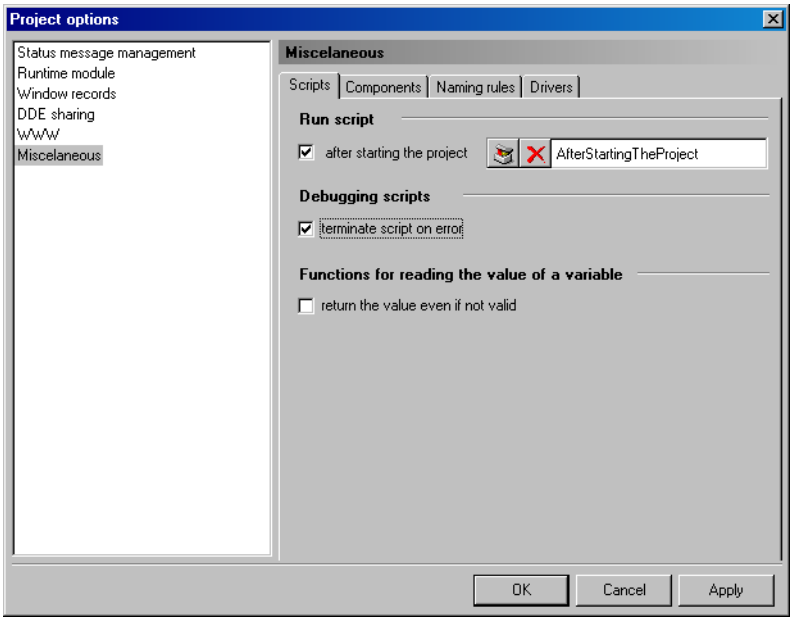
This parameter specifies the maximum time period for providing a Web client with a new port number. If the operation fails to complete within the specified time, the connection with the client is terminated and the service is ready for another client to connect.

Client socket idle timeout (s)

This parameter specifies the maximum time period for starting communication between a Web client and the data server using a new port number. If the Web client does not connect to the data server using the new port number within the specified time or no communication takes place between the client and the data server during this time, the port number is made available for another client.

2.4.2.6 Miscellaneous

Scripts



Run script

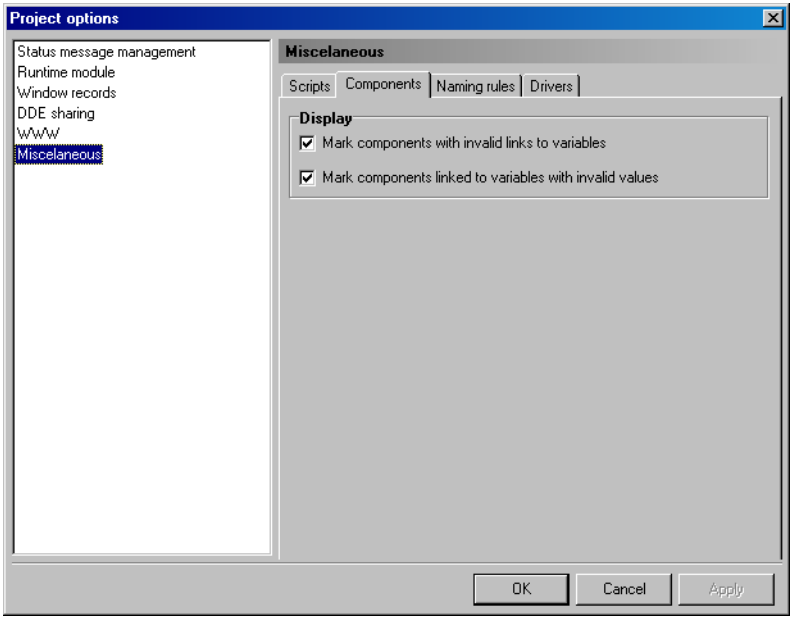
after starting the project This parameter specifies the script to be run as the first script after starting the visualizing project in runtime mode.

Debugging scripts

terminate script on error This option 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. *R*, *RSys*, etc.). It does not apply to other objects or syntax errors of the *VBScript* language.

Components



Display

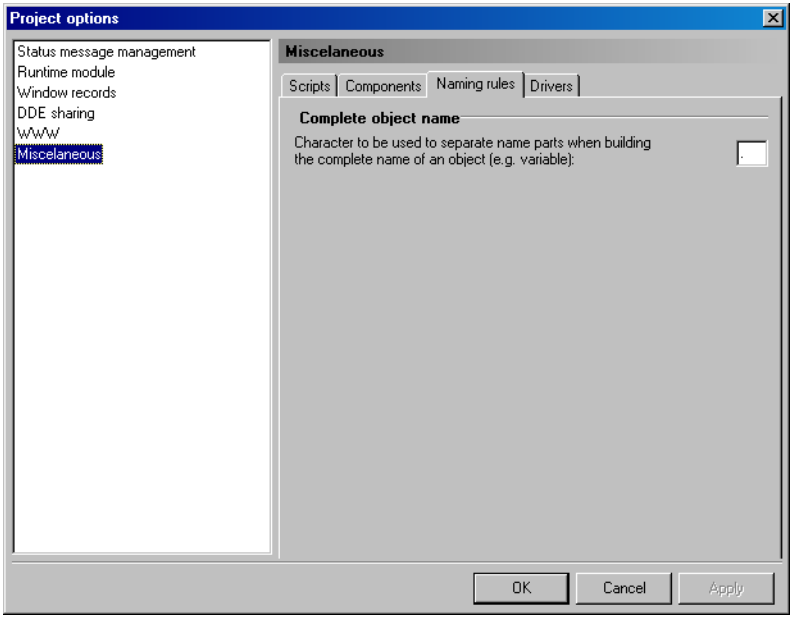
Mark components with invalid links to variables

This option determines whether to graphically mark the components with invalid links to variables (e.g. link to a variable that was later deleted through the ***Station manager***). If the option is active, such components will be marked with a red border.

Mark components linked to variables with invalid values

This option determines whether to graphically mark the components linked to variables with invalid values (e.g. link to a variable 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.

Naming rules



Complete object name

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

This parameter determines the character used when building the complete name of an object (e.g. variable) 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 variable *Temperature1* belonging to the station *Tecomat1* is displayed as *Tecomat1.Temperature1*.

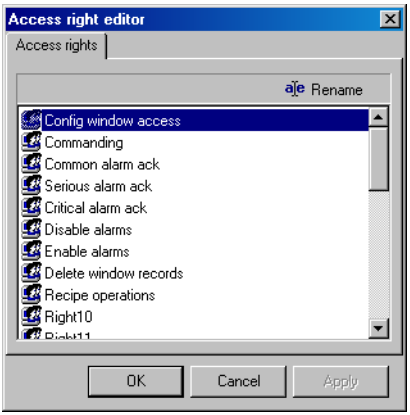
👁 **WARNING** This character must not be included in the name of any object within a *Reliance* visualizing project!

2.4.3 Access right editor

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

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 variable using the Display component, the user with the *Right3* access right will not be allowed to perform the operation).

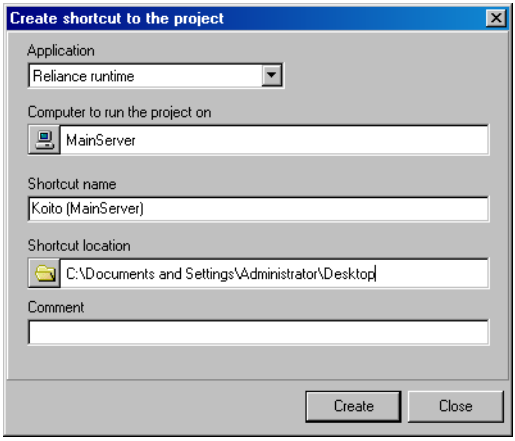
It is strongly recommended to rename the access rights on the functional basis.



Rename — This command is used to rename the currently selected access right.

2.4.4 Creating a shortcut to the project

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



Application — This parameter determines the program to open the project.

Computer to run the project on — This parameter, which is only used when the *Application* parameter specifies a runtime module, determines the logical computer (defined through the *Project structure manager*) whose settings (e.g. IP address, system name, connected stations, etc.) will be used by the runtime module when starting the visualizing project through the shortcut.

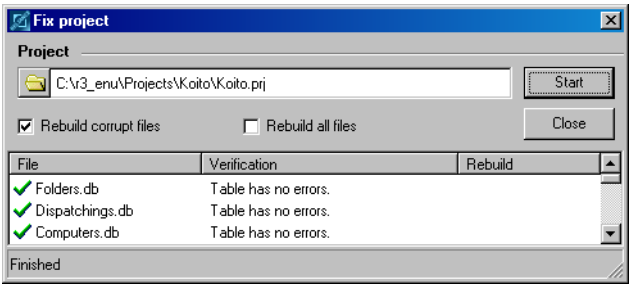
Shortcut name — This parameter determines the name of the shortcut.

Shortcut location — This parameter 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 — This parameter can be used to specify an optional description of the shortcut.

2.4.5 Fixing a project

To fix a visualizing project, choose the ▸ ***Tools*** ▸ ***Fix project...*** command. This command brings up the ***Fix project*** dialog box.



Rebuild corrupt files This option determines whether to rebuild only the files that appear to be corrupt.

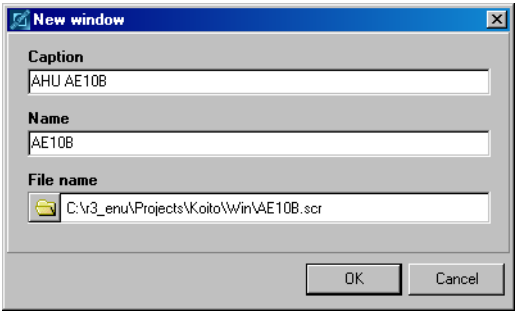
Rebuild all files This option determines whether to rebuild all the files.

2.5 VISUALIZING PROJECT WINDOW

A visualizing project usually contains one or more project windows. A project window is designed to contain components from the component palette. Together with these components, the window becomes the user's tool for monitoring and/or controlling technological processes.

2.5.1 Creating a new window

To create a new window, choose the ▸ ***File*** ▸ ***New window*** command. This command brings up the ***New window*** dialog box.



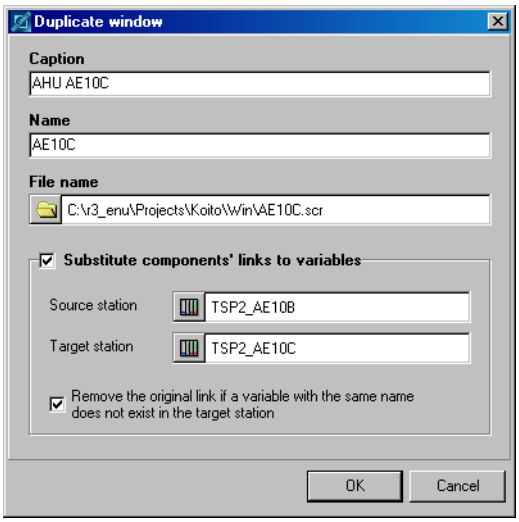
Caption — This parameter specifies the text to be displayed in the window's caption bar (see the chapter 2.5.3 WINDOW PROPERTIES).

Name — This parameter 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.

File name — This parameter specifies the complete name of the file for storing the window. It is automatically generated from the *Name* parameter.

2.5.2 Duplicating a window

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



Caption This parameter specifies the text to be displayed in the window's caption bar (see the chapter 2.5.3 WINDOW PROPERTIES).

Name This parameter 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.

File name This parameter specifies the complete name of the file for storing the window. It is automatically generated from the *Name* parameter.

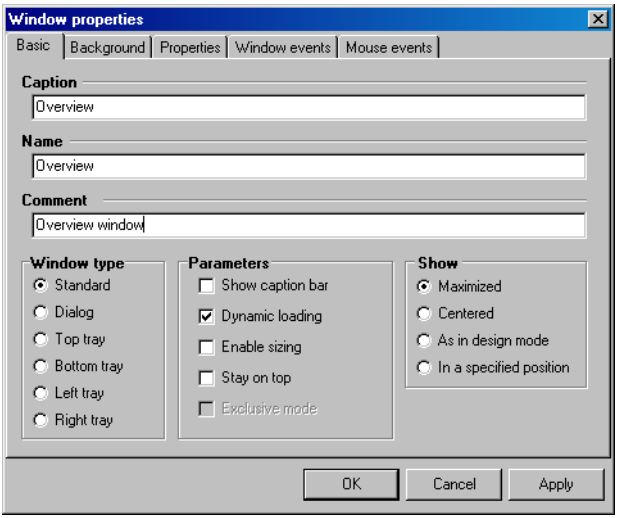
Substitute components' links to variables
This option determines whether to substitute components' links to variables while duplicating the window. If the option is active, all links to variables from the station specified by the *Source station* parameter will be replaced by links to same-named variables from the station specified by the *Target station* parameter.

2.5.3 Window properties

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

ⓘ **NOTE** Specification of certain properties will not take effect in the development environment, but in runtime mode only.

Basic



Caption This parameter specifies the text to be displayed in the window's caption bar.

Name This parameter 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.

Comment This parameter can be used to specify an optional description of the window.

Window type

Unlike standard windows, dialog windows have a fixed size and always display a caption bar. These properties are optional for standard windows. Also, dialog windows can be closed by the user in runtime mode. 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 module's main window.

Parameters

Show caption bar	This option determines whether to display the window's caption bar in runtime mode.
Dynamic loading	This option 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 option is not active, the window is loaded into memory right after starting the project in runtime mode and it stays in memory until the project is terminated. It is recommended to leave this option active for most windows.
Enable sizing	This option determines whether to enable the users to resize the window in runtime mode.
Stay on top	This option determines whether the window should stay on top of the other open windows in runtime mode – except for windows with the identical feature and trays.
Exclusive mode	This option determines whether the window should be displayed in the exclusive mode (modally). When the window is displayed in the exclusive mode, the user is not allowed to switch to another project window or to perform other operations within the runtime module until the window is closed.

Show

This parameter specifies how the window is to be displayed in runtime mode.

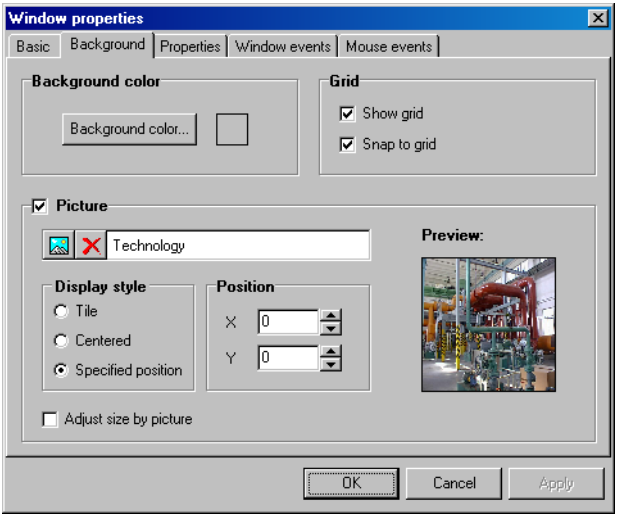
Maximized The window fills the area of the runtime module's main window.

Centered The window has the same size as in design mode and is positioned in the center of the runtime module's main window.

As in design mode The window has the same size and position as in design mode.

In a specified position The window's size and position is defined by the specified co-ordinates of the window's upper left corner.

Background



Background color This parameter specifies the window's background color.

Grid

Show grid This option determines whether to display dots on the window to make the grid visible.

Snap to grid This option determines whether to align components in the window to the closest grid point.

Picture This parameter specifies the picture to be displayed on the window's background.

Display style This parameter specifies how the background picture is to be displayed.

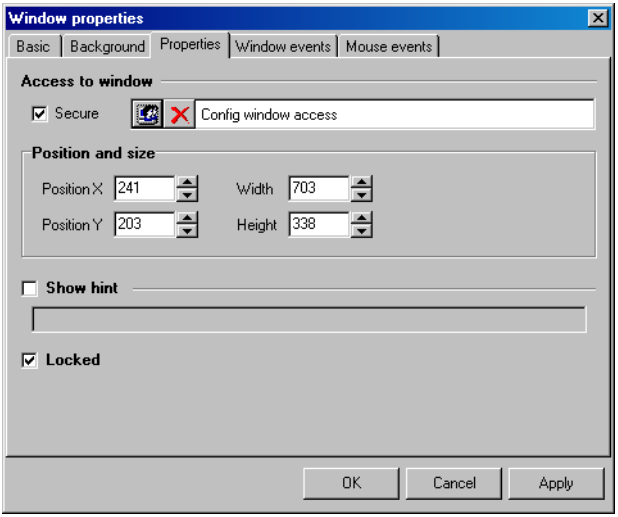
Tile The picture is drawn as tiles to fill up the entire window area.

Centered The picture is positioned in the center of the window.

Specified position The picture's position is defined by the specified coordinates (the *Position* parameter) of the picture's upper left corner.

Adjust size by picture This option determines whether to adjust the size of the window to the picture.

Properties



Access to window This parameter specifies the access rights required for accessing the window (i.e. activating the window).

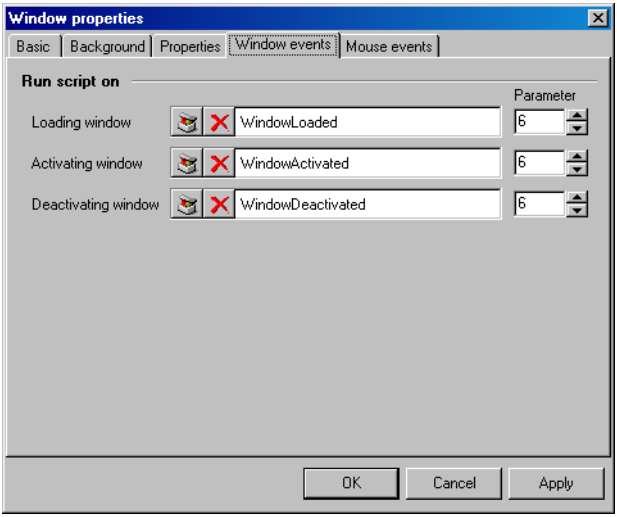
Position and size These parameters determine the window's position and size in pixels.

Show hint This option determines whether to display the specified help hint when the mouse cursor rests momentarily on the window area.

Locked This option determines whether the position and size of all the components in the window are locked.

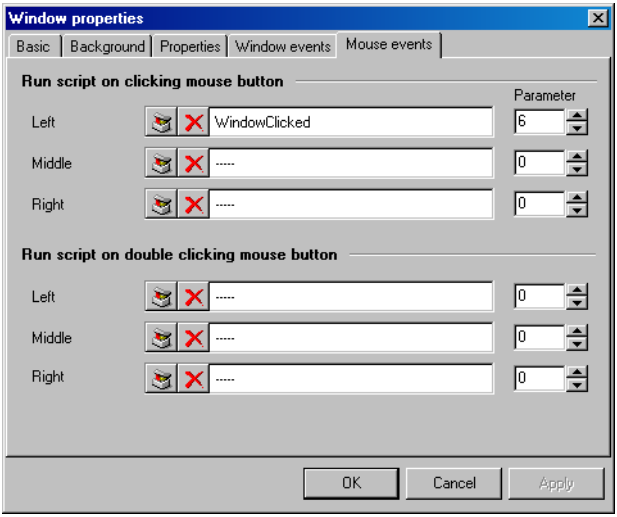
Window events

These parameters enable you to specify the scripts to be executed when the window is loaded into memory, activated or deactivated. You can also specify the parameters to pass to the scripts.



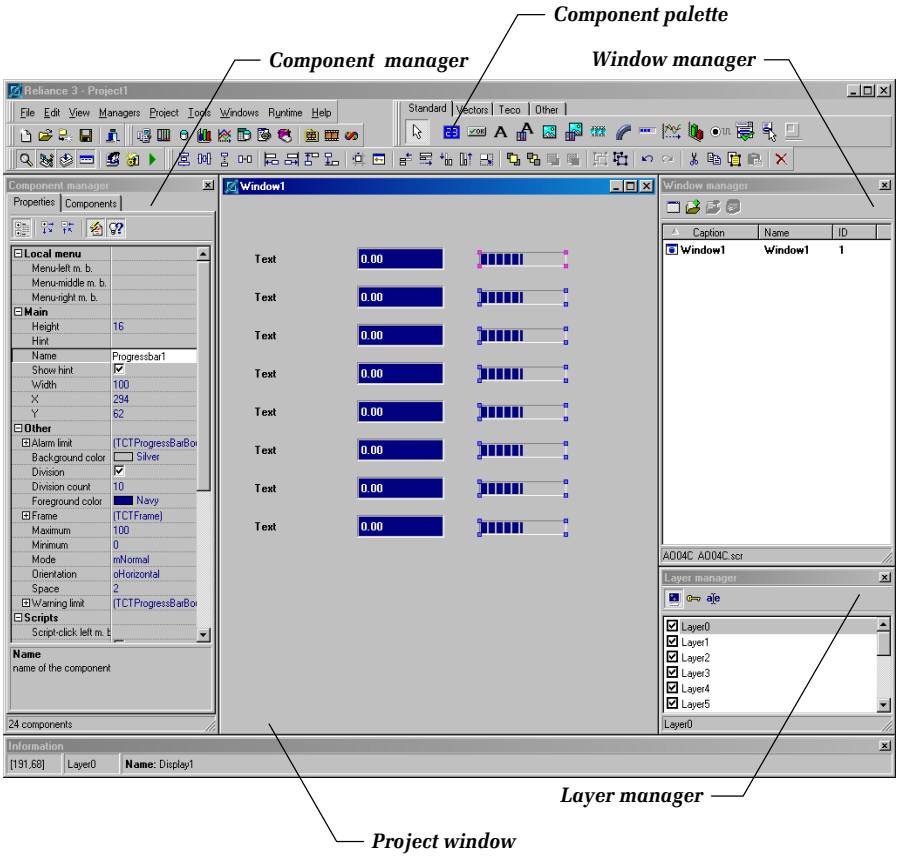
Mouse events

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



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 the other open windows.



When designing a window, add individual components from the component palette to the window. If the 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. The layer may be hidden (all components located on this layer will be hidden in design mode) or locked (all components located on this layer will be locked in design mode) as needed.

The component selected on the palette by clicking the left mouse button may be placed into the 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 several 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.

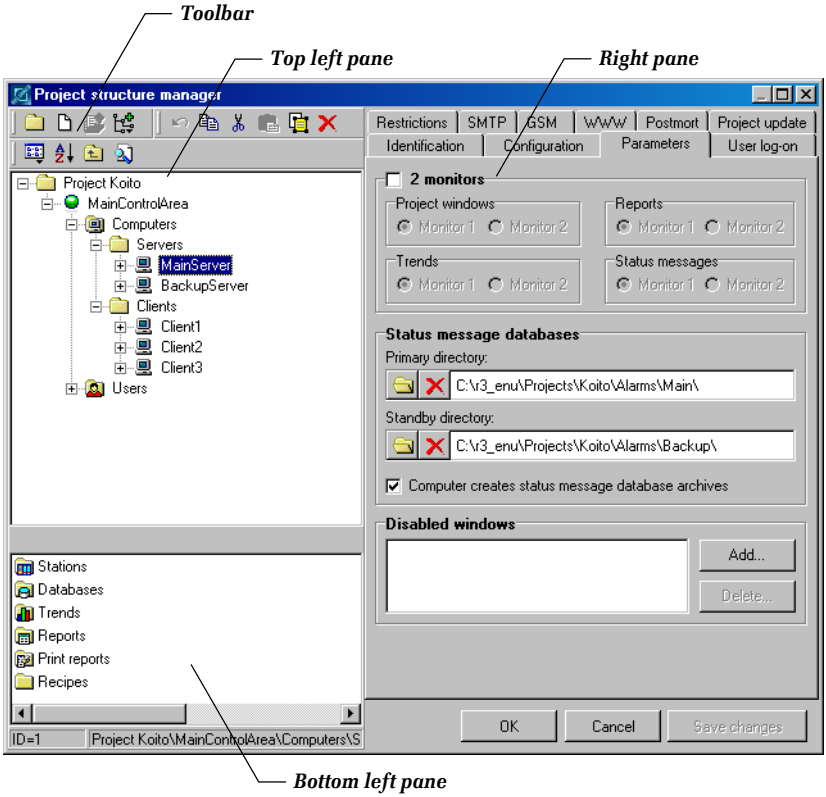
The last step when designing a window is configuring the properties of the components, which affects their appearance and behavior. To configure the properties of a single component, use the component's property editor (double-click the component or choose the ***Component properties...*** command from the component's local 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 several components at the same time, select the components and edit their properties through the ***Component manager***.

2.6 MANAGERS

The managers are windows designed to define and configure the objects that are the building blocks of every *Reliance* visualizing project. These are computers, users, stations, databases, trends, reports, print reports, recipes, scripts, windows, components and others.

Most of the managers have a uniform appearance and behavior. The following figure illustrates the appearance of most of the managers.



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 only allows you to select and edit one object at a time. When you select an object in the tree diagram, the manager displays the subordinated objects in the bottom left pane and the object's parameters in the right pane.

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 several objects at a time. When you select an object in the list, the manager displays the object's parameters in the right pane.

Right pane

The right pane displays parameters of the object(s) selected in the top left or bottom left pane. The parameters can be edited as needed. When you edit a parameter, 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 parameters to see if they are correct. If so, the manager assigns the edited parameters 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 databases by the *Save changes* command or canceled by the *Cancel* command. Naturally, you can also edit a parameter and use the *Save changes* command immediately. The manager checks all the edited parameters to see if they are correct. If so, the manager assigns the edited parameters to the selected object(s) and saves the object(s) to the databases.

2.6.1 Description of common commands

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

 ***Cancel last changes*** (*Ctrl+Z*)

This command is used to cancel the edits that have not yet been assigned to the selected object(s).

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

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

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

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

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

 ***Display*** This command is used to change the display style of the objects listed in the bottom left pane of the manager.

 ***Alphabetically sort objects***

This command is used to alphabetically sort the objects directly subordinated to the object selected in the tree diagram.

 ***One level up*** This command is used to move the selection one level up in the tree diagram.

 ***Find object*** (*Ctrl+F*) This command is used to bring up the ***Find object*** dialog to search for an object by its name.

 ***Options*** This command is used to bring up the ***Options*** dialog to view or configure the settings related to the manager.

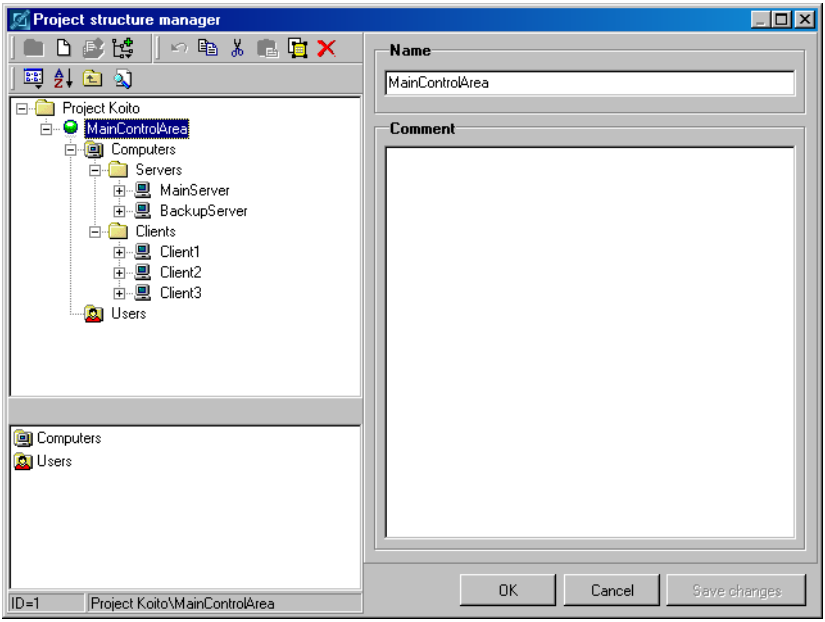
2.6.2 Project structure manager

The ***Project structure manager*** is a tool designed for defining the structure of the entire visualizing project. It enables you to define the structure consisting of control areas, computers, users, network connections, stations, databases and other objects, so that it corresponds to a real plant site.

A visualizing project always comprises at least one control area. The control area comprises at least one computer and any number of users. The objects defined through other managers, such as stations, databases, 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 visualizing project, see the examples at the end of this chapter.

2.6.2.1 Defining control areas

The control area is an independent unit representing a separate locality, where one or more computers designed for running the visualizing project are located. Computers and users can be defined within the control area.



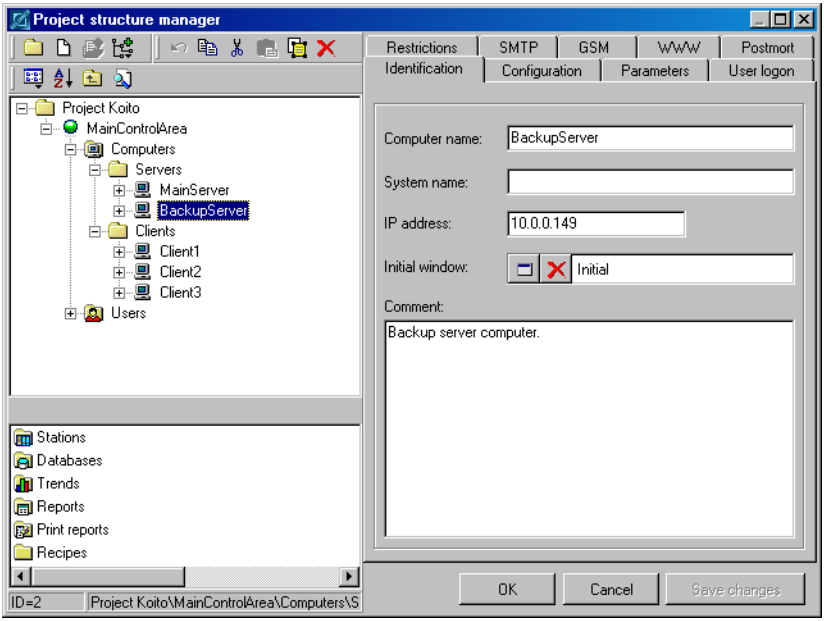
Name This parameter specifies the control area's name that must be unique within the project and cannot contain illegal characters.

Comment This parameter can be used to specify an optional description of the control area.

2.6.2.2 Defining computers

The computer, also referred to as logical computer, is an object representing the actual computer on which the visualizing project will be running at the end user site.

Identification



Computer name This parameter specifies the computer's name that must be unique within the project and cannot contain illegal characters.

System name This parameter specifies the name of the actual computer within the computer network.

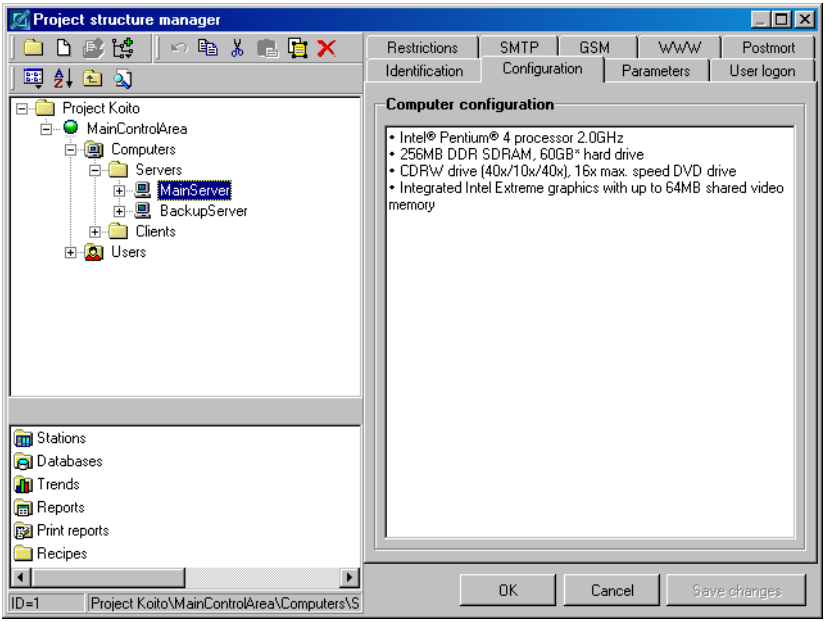
IP address This parameter specifies the IP address of the actual computer within the computer network. If the actual computer has a DHCP-assigned IP address, leave the parameter blank.

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

Initial window This parameter specifies the project window to appear as the first window after starting the visualizing project in runtime mode on the computer.

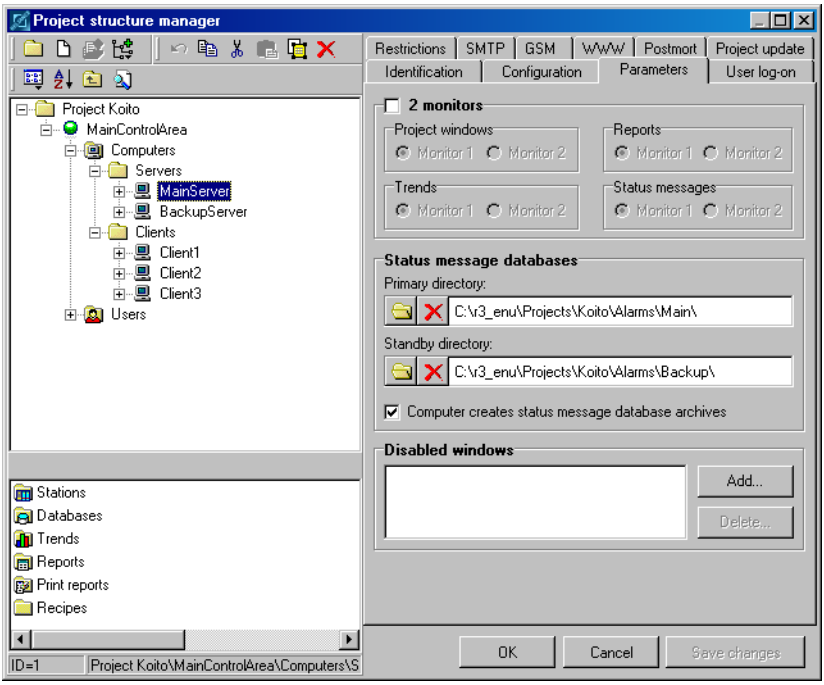
Comment This parameter can be used to specify an optional description of the computer.

Configuration



Computer configuration This parameter can be used to specify the computer's hardware configuration.

Parameters



2 Monitors These parameters allow you to configure the behavior of the runtime module when operating on a multi-monitor system.

Status message databases

Primary directory This parameter specifies the primary directory for storing the databases of status messages.

Standby directory This parameter specifies the standby directory for storing the databases of status messages. It will only be used when the primary directory is not available.

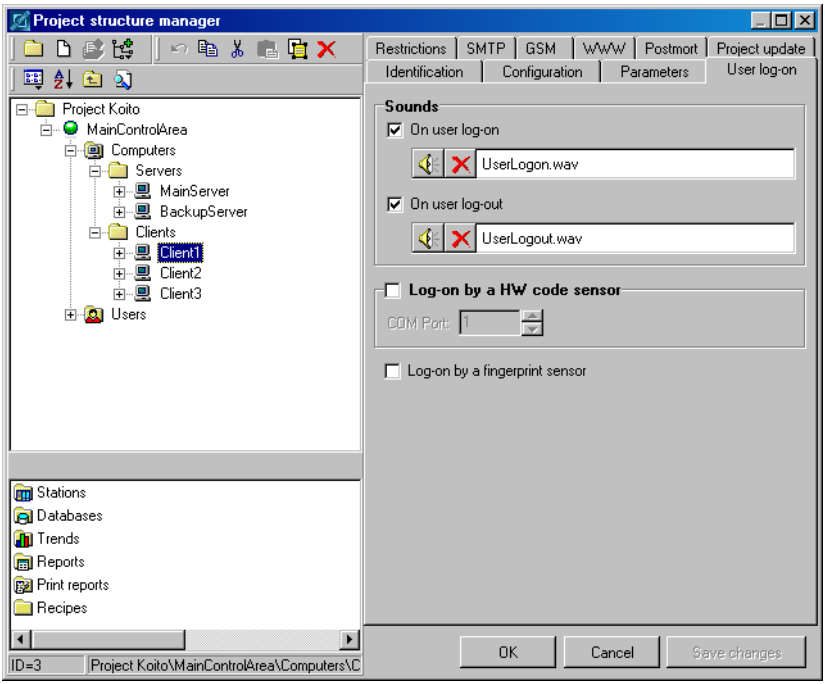
Computer creates status messages database archives

This option determines whether the runtime module running on the computer should create archive databases of status messages.

Disabled windows

This parameter specifies a list of project windows that should not be available to the user in runtime mode.

User log-on



Sounds

These parameters specify the sounds to be played when the user logs on and out. The sound files (files in the **.wav* format) must be located in the »MMedia« subdirectory of the visualizing project.

Log-on by a HW code sensor

This parameter determines whether a HW code sensor will be connected to the computer to enable

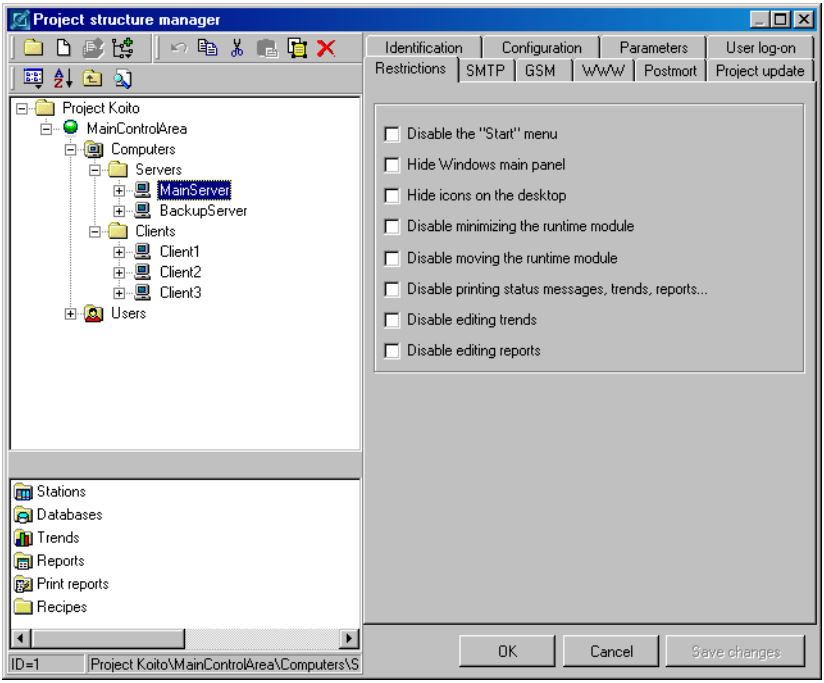
the users to log on to the program using a special card with a unique code.

Log-on by a fingerprint sensor

This parameter determines whether a fingerprint sensor will be connected to the computer to verify the users by their fingerprints when logging on to the program.

Restrictions

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



Disable the "Start" menu This option determines whether to disable using the Windows ***Start*** menu.

Hide Windows main panel

This option determines whether to hide the Windows main panel.

Hide icons on the desktop This option determines whether to hide icons on the Windows desktop.

Disable minimizing the runtime module
This option determines whether to disable minimizing the main window of the runtime module.

Disable moving the runtime module
This option determines whether to disable moving the main window of the runtime module.

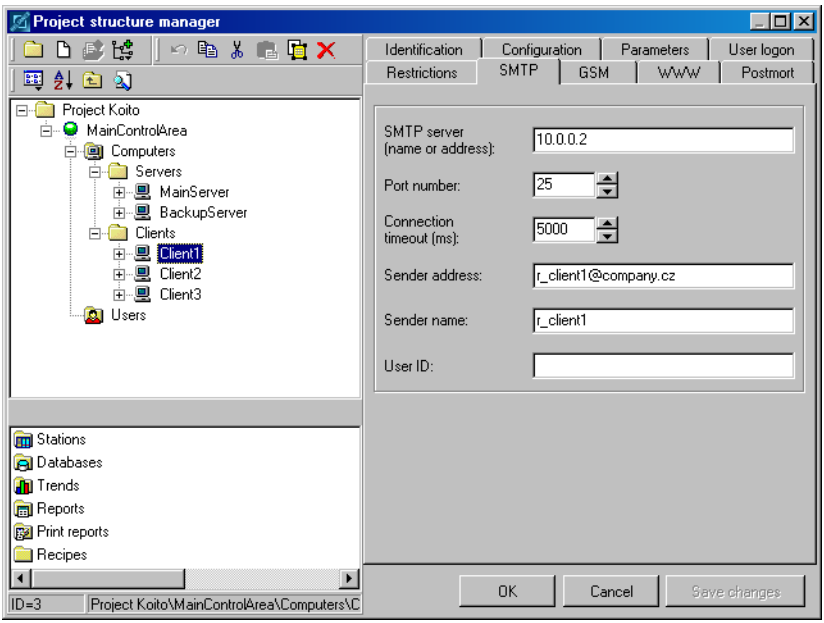
Disable printing status messages, trends, reports...
This option determines whether to disable printing status messages, trends and reports. However, this option only applies to printing invoked by the user. It does not affect online printing status messages.

Disable editing trends This option determines whether to disable editing trends via the ***Trend manager***.

Disable editing reports This option determines whether to disable editing reports via the ***Report manager***.

SMTP

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



SMTP server (name or address)

This parameter specifies the name or IP address of the SMTP server to be used for sending e-mail messages.

Port number

This parameter specifies the port used by the SMTP server for sending e-mail messages.

Connection timeout (ms)

This parameter specifies the maximum time period for the runtime module to connect to the SMTP server.

Sender address

This parameter specifies the e-mail address of the sender.

Sender name

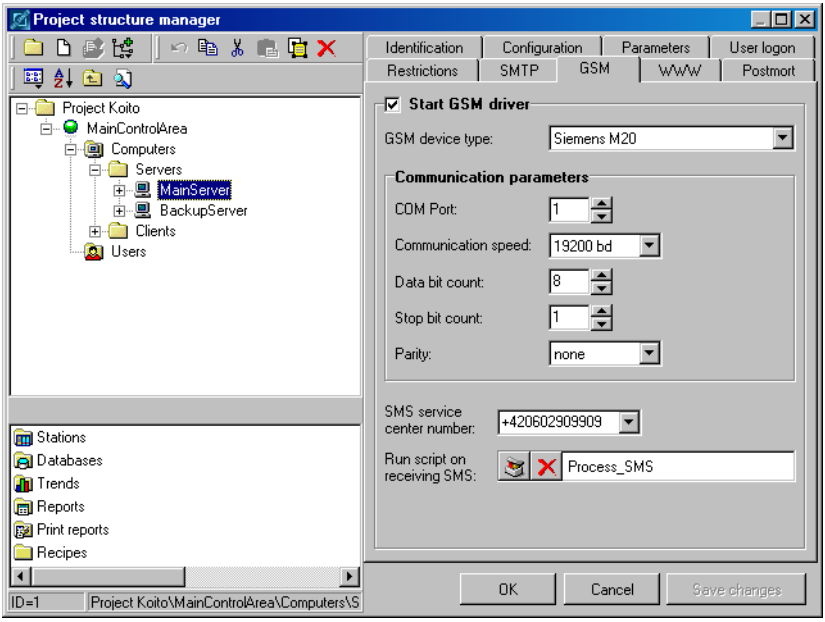
This parameter specifies the name of the sender.

User ID

This parameter specifies the identification string of the user. It must only be specified if the SMTP server requires a certain form of authentication.

GSM

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



Start GSM driver — This option determines whether to launch a GSM device driver when starting the visualizing project in runtime mode.

GSM device type — This parameter specifies the type of GSM device to be used for sending and receiving SMS messages.

Communication parameters

These parameters enable you to configure the connection to the computer using the RS232 interface (serial port number, communication speed in bauds, number of data and stop bits, parity).

SMS service center number

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

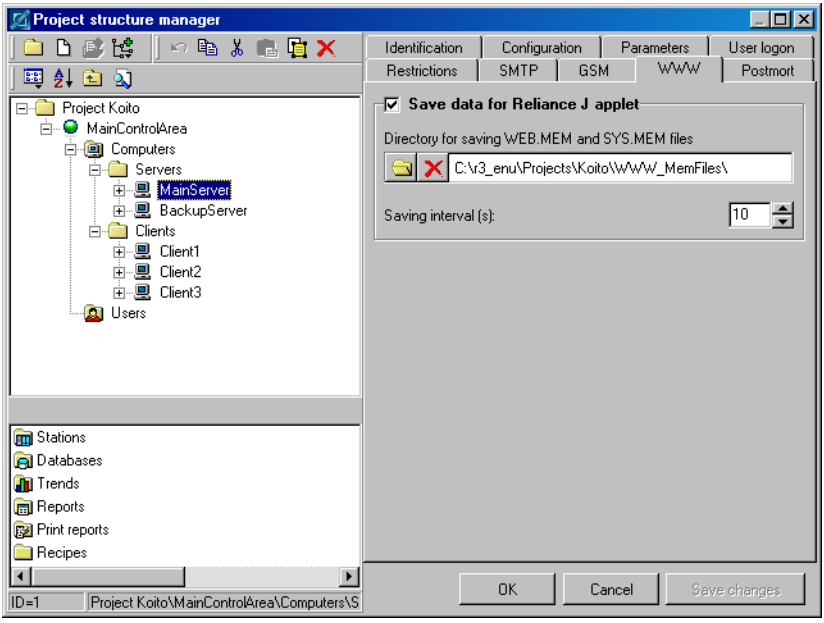
Run script on receiving SMS

This parameter 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 or on the installation CD.

WWW

These parameters allow you to configure whether and how the runtime module distributes current technological data to *Reliance J* Web clients using the MEM files. For more information on this topic, see the chapter 3. TECHNOLOGICAL DATA DISTRIBUTION OVER THE INTERNET/INTRANET).



Save data for Reliance J applet

This option determines whether to periodically save current technological data to the MEM files for the Web clients. The Web clients obtain the data by periodically reading the files.

Directory for saving WEB.MEM and SYS.MEM files

This parameter specifies the path for saving the MEM files. The MEM files must be located in the same directory on the Web server as the visualizing project in the WWW format.

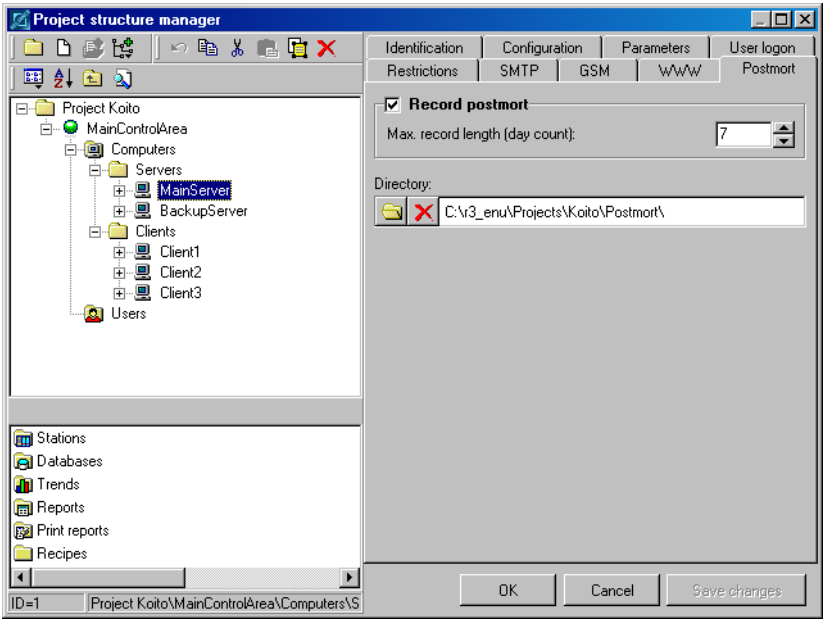
Saving interval (s)

This parameter specifies the time period used for saving the MEM files.

Postmort

These parameters allow you to configure the Postmort function. If the function is activated, the runtime module records changes in technological 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 view the process retrospectively. 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.



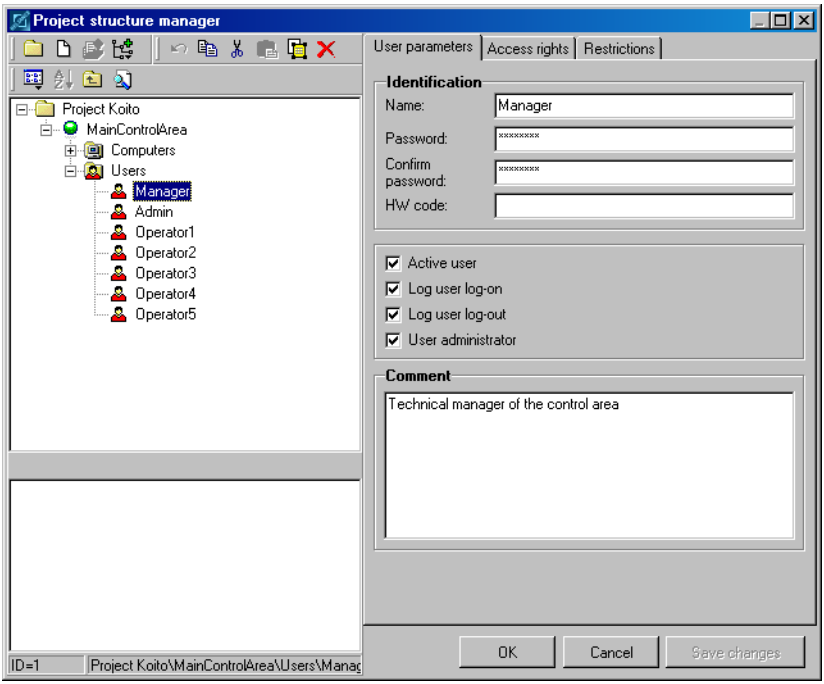
Record postmort This parameter determines whether to activate recording the process.

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

Directory This parameter specifies the directory for storing the files containing Postmort records.

2.6.2.3 Defining users

Parameters

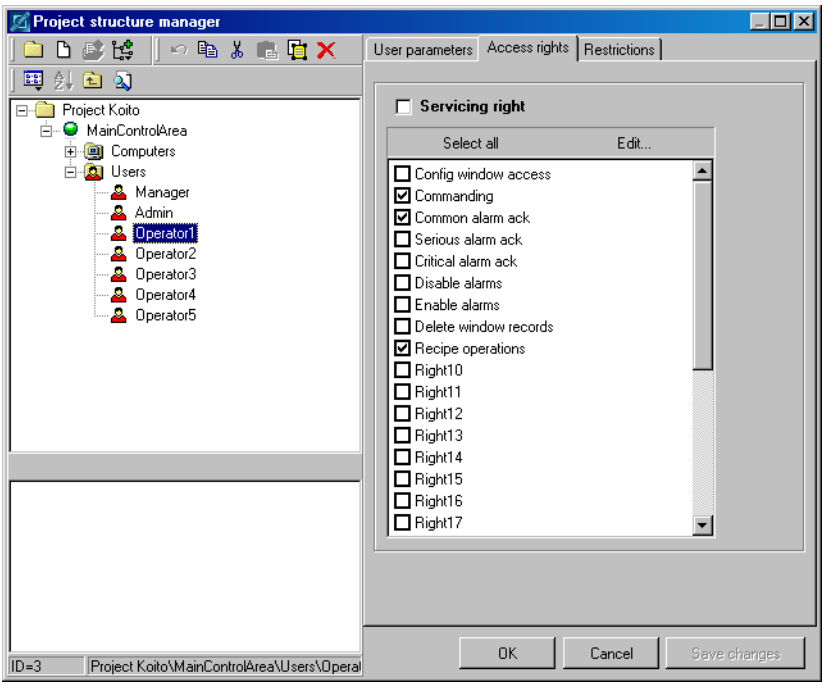


Name This parameter specifies the user's name that must be unique within the control area and cannot contain illegal characters. The name is entered by the user when logging on to the program.

Password	This parameter specifies the password that is entered by the user when logging on to the program.
Confirm password	This parameter must have the same value as the *Password* parameter.
HW code	This parameter specifies the code assigned to the user if logging on by HW code sensor is used. Otherwise, the value is ignored.
Active user	This option 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	This option determines whether to log the information about logging the user on to the program to the database of system messages.
Log user log-out	This option determines whether to log the information about logging the user out of the program to the database of system messages.
User administrator	This option determines whether to allow the user to administrate users in runtime mode. If the option is active, the user can add new users and modify or delete existing users.
Comment	This parameter can be used to specify an optional description of the user.

Access rights

These options specify the user's access rights to the program (see also the chapter 2.4.3 ACCESS RIGHTS EDITOR). For example, if the *Commanding* access right is required for switching on o pump by the *Button* component, it can only be done by the user(s) who has been assigned this access right.



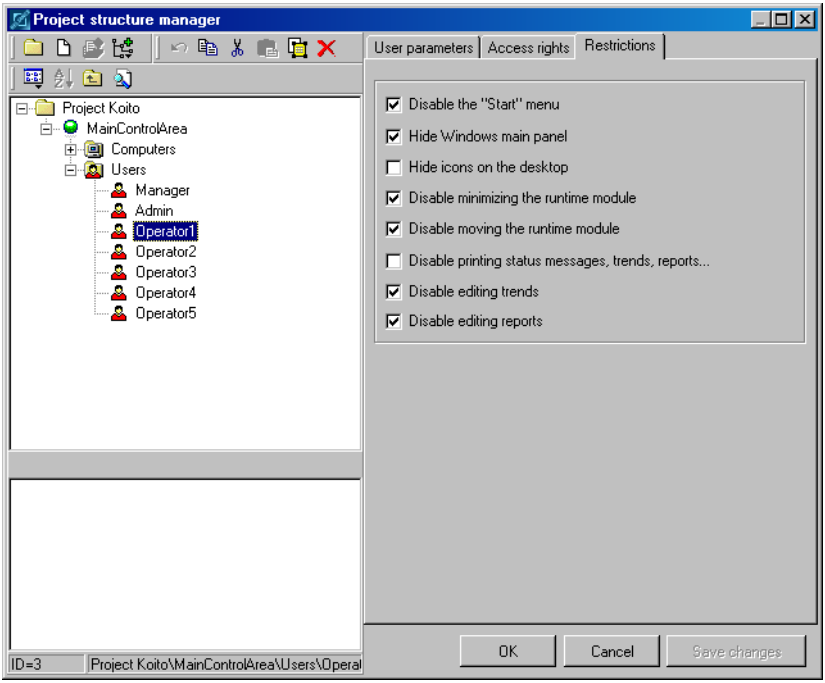
Servicing right — This option determines whether the user has a special right that ensures the user cannot be modified or deleted in runtime mode via the ***User manager*** by a user administrator that has not been assigned the *Servicing right*. This right can only be activated through this parameter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

Select all — This command is used to select all the rights in the list.

Edit — This command is used to bring up the ***Access rights editor*** to rename the rights (see the chapter 2.4.3).

Restrictions

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



Disable the 'Start' menu This option determines whether to disable using the Windows ***Start*** menu.

Hide Windows main panel
This option determines whether to hide the Windows main panel.

Hide icons on the desktop This option determines whether to hide icons on the Windows desktop.

Disable minimizing the runtime module
This option determines whether to disable minimizing the main window of the runtime module.

Disable moving the runtime module

This option determines whether to disable moving the main window of the runtime module.

Disable printing status messages, trends, reports...

This option determines whether to disable printing status messages, trends and reports. However, this option only applies to printing invoked by the user. It does not affect online printing status messages.

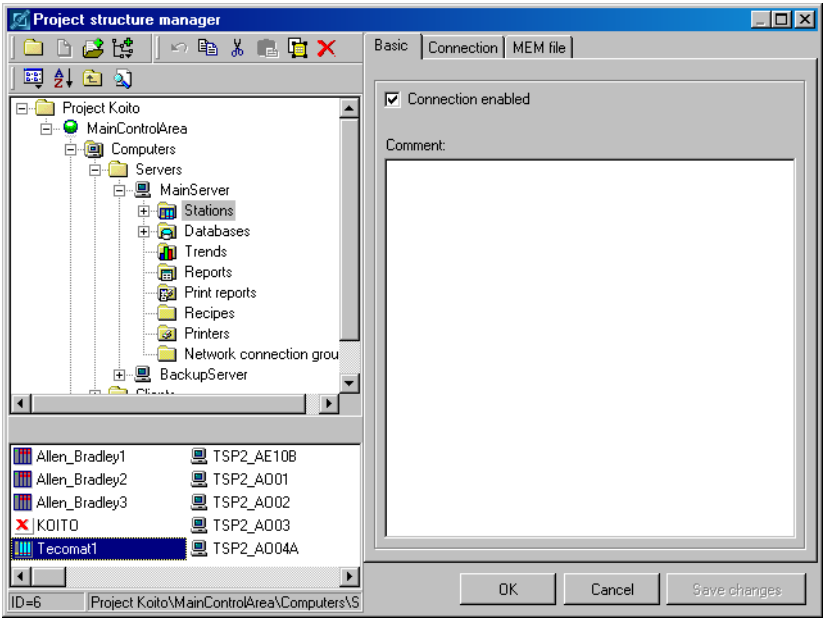
Disable editing trends This option determines whether to disable editing trends via the ***Trend manager***.

Disable editing reports This option determines whether to disable editing reports via the ***Report manager***.

2.6.2.4 Connecting stations

To make a station (e.g. PLC previously defined via the ***Station manager***) accessible to the runtime module running on the actual computer, you have to add a link to the station object to the logical computer's *Stations* folder. This is called connecting a station to a computer.

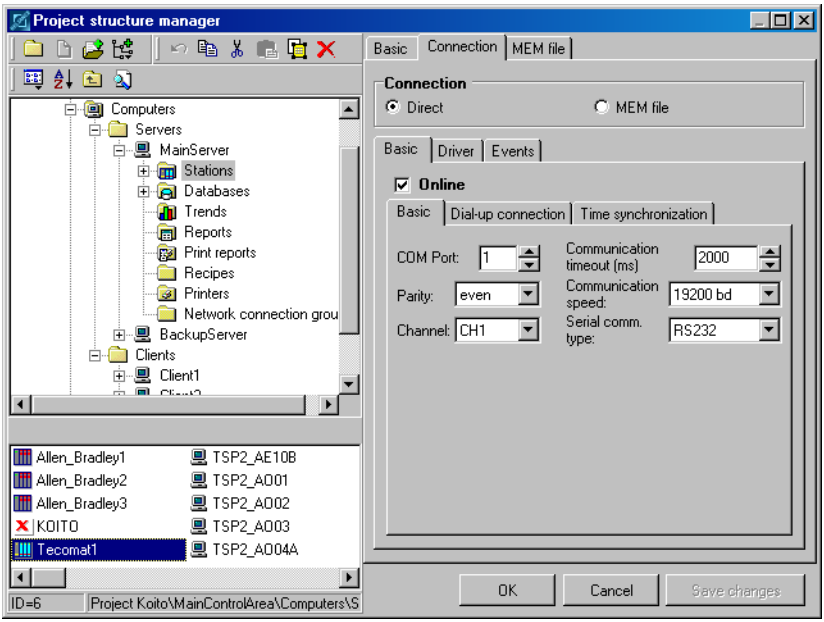
Basic



Connection enabled This option determines whether to make the station accessible to the computer. If the option is not active, the runtime module acts as if the station was not connected to the computer (i.e. as if the link to the station were not contained in the *Stations* folder) and ignores this station and its subordinated objects (variables, communication zones and status messages) when loading the visualizing project. It can be useful when debugging the application to speed up starting the project in runtime mode if concentrating only on certain stations.

Comment This parameter can be used to specify an optional description of the connected station.

Connection



Direct If this option is active, the runtime module is connected to the station through a communication driver.

MEM file If this option is active, the runtime module obtains current data of the station by periodically reading a MEM file updated by a runtime module running on another computer. The MEM file contains a binary image of the station's memory.

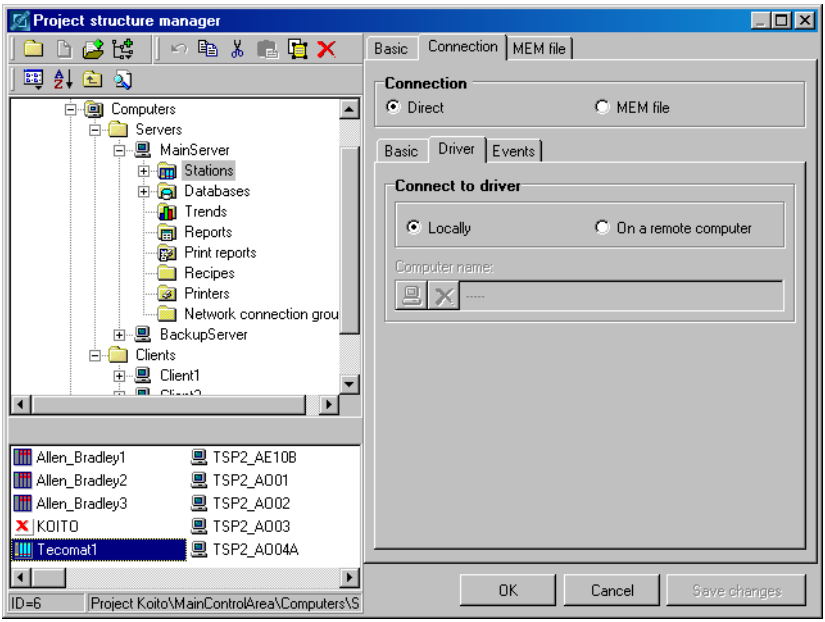
Basic

Most of the connection parameters are specific to particular station types and are described at the end of this chapter.

Online This option determines whether the communication driver attempts to establish a connec-

tion to the station. The option only applies to communication drivers that support offline mode (in offline mode, the driver does not attempt to establish a connection to the station).

Driver

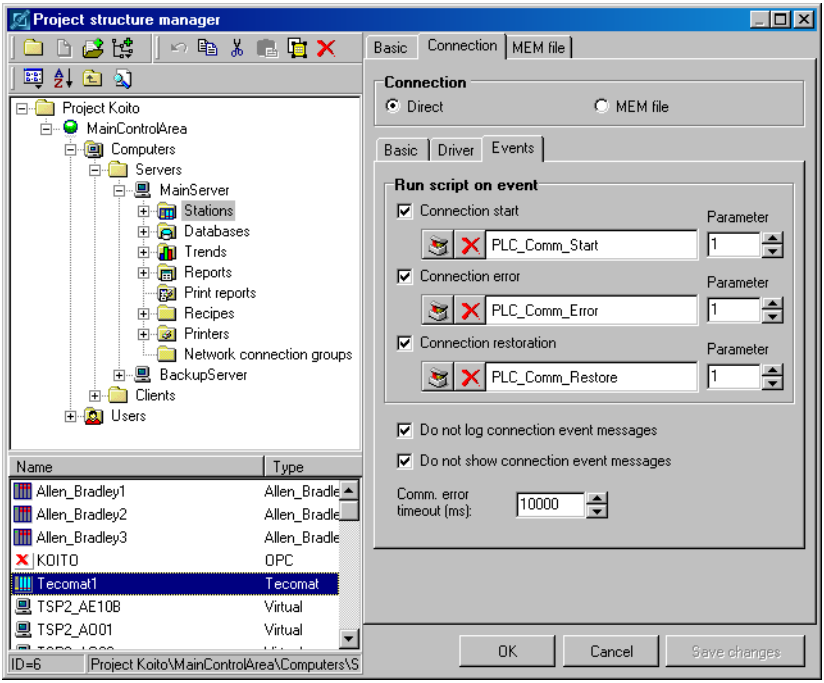


Connect to driver

Locally — If this option is active, the communication driver starts on the same computer as the runtime module.

On a remote computer — If this option is active, 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 the security settings for the driver in Windows.

Events



Run script on event

Connection start — These parameters determine whether to run the specified script when a connection to the station is established for the first time.

Connection error — These parameters determine whether to run the specified script when an error occurs in a connection to the station.

Connection restoration — These parameters determine whether to run the specified script when a connection to the station is restored.

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

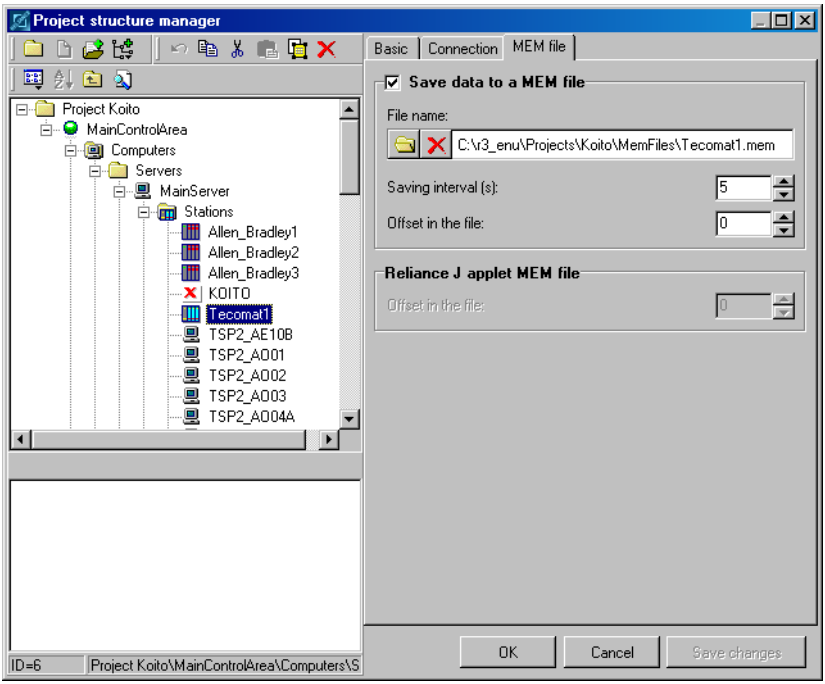
Do not log connection event messages

This option determines whether to log information about connection related events to the database of system messages.

Do not show connection event messages

This option determines whether to show information about connection related events in the list of current messages.

MEM file



Save data to a MEM file If this option is active, the runtime module periodically saves current data of the station to a MEM file. The MEM file contains a binary image of the station's memory. Thus, a runtime module running on another computer can obtain current data of the station by periodically reading the MEM file.

File name This parameter specifies the path and name of the MEM file.

Saving interval (s) This parameter specifies the time period used for saving the MEM file.

Offset in the file This parameter specifies the position in the MEM file at which to write the station's memory image. Thus, it is possible to use a single file to save data from more than one station.

WARNING Stations' memory images must not overlap.

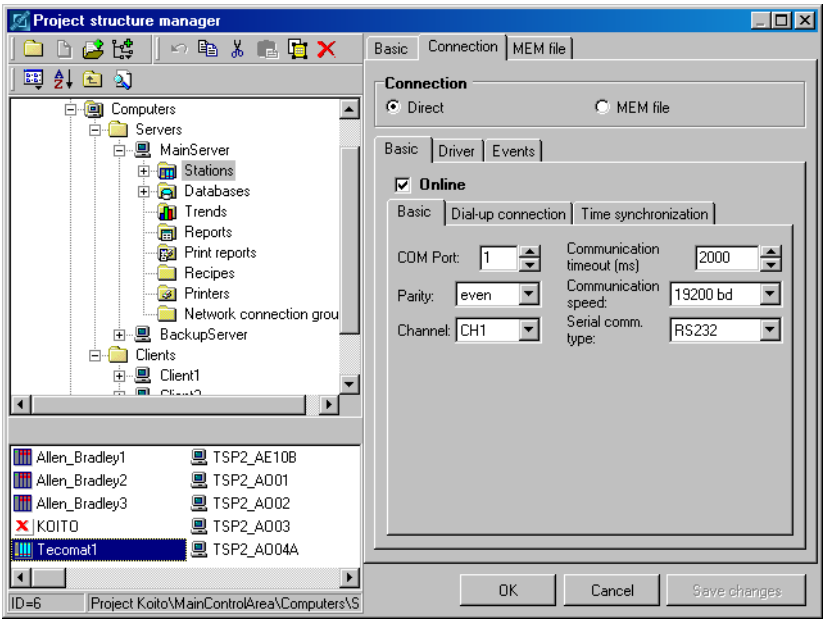
Reliance J applet MEM file

Offset in the file This parameter specifies the position in the *Reliance J* applet MEM file at which to write the station's memory image.

WARNING Stations' memory images must not overlap.

Connection parameters for Tecomat and Tecoreg stations

Basic



COM Port — This parameter specifies the number of COM port that is used for communication to the station via a serial cable.

Parity — This parameter specifies the parity used for communication to the station.

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

Channel — This parameter specifies the station's communication channel to be used for communication between the computer and the station.

Communication timeout (ms)

This parameter specifies the maximum time period between sending a request to and receiving

a response from the station by the communication driver. If the station 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 station again. If the station does not respond repeatedly, it is recognized as a failure in communication to the station.

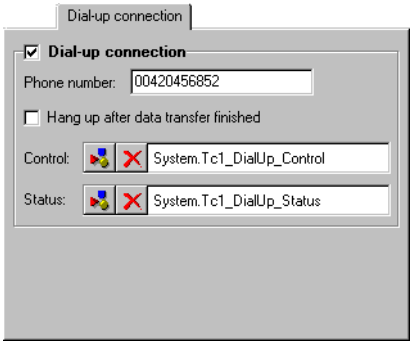
☞ **RECOMMENDATION** Failures in communication to the station slow down the communication to all the stations connected on the same serial cable. If the station is known to be out of service, it is recommended that you temporarily deactivate the *Online* parameter.

Communication speed This parameter specifies the communication speed in bauds.

WARNING The maximum possible communication speed depends on the data transmission line and connection quality – the better the connection, the higher the communication speed may be.

Serial comm. type This parameter specifies the type of serial communication (RS232 or RS485/422 interface).

Dial-up connection



Dial-up connection This parameter determines whether a connection to the station must be established through a modem.

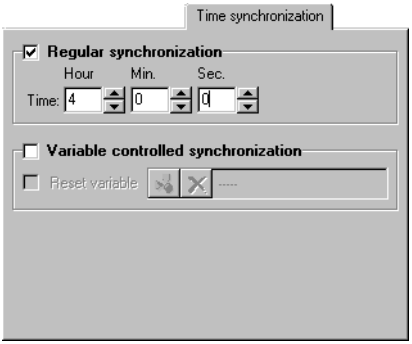
Phone number This parameter specifies the number of a telephone line used by the station's modem. Several stations can share the same modem.

Hang up after data transfer finished
This parameter determines whether to automatically terminate the dial-up connection after the completion of data transfer.

Control This parameter specifies the link to a binary-type variable to be used to control the dial-up connection.

Status This parameter specifies the link to an integer-type variable that should receive the dial-up connection status.

Time synchronization



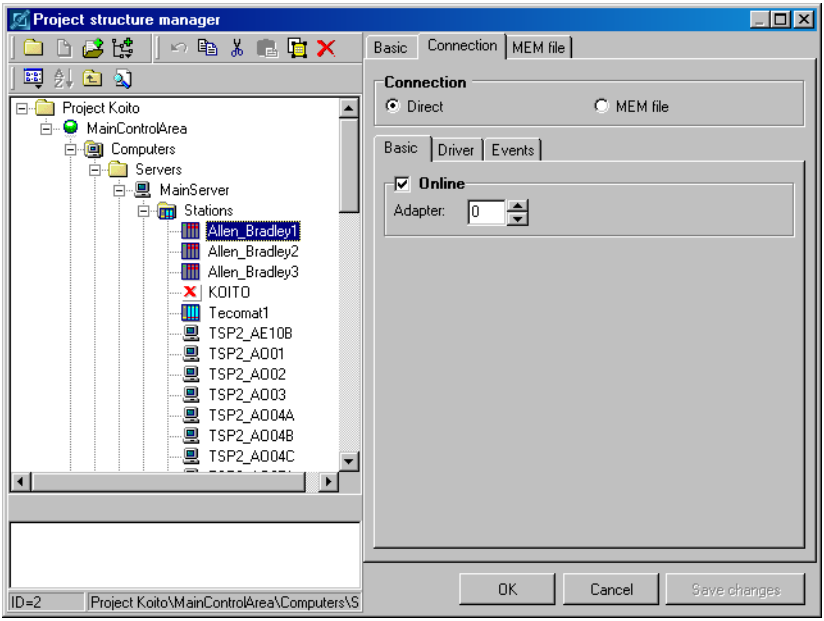
Regular time synchronization
These parameters determine whether and when to synchronize the system time of the station with the system time of the computer. The synchronization is performed every day at the specified time.

Variable controlled synchronization
These parameters determine whether and when to synchronize the system time of the station with the system time of the computer. The synchronization is performed on the leading edge of a binary-type variable (the off-to-on transition). The *Reset*

variable parameter determines whether to reset the control variable (i.e. set the variable to the off state) after detecting the leading edge.

Connection parameters for Allen Bradley stations

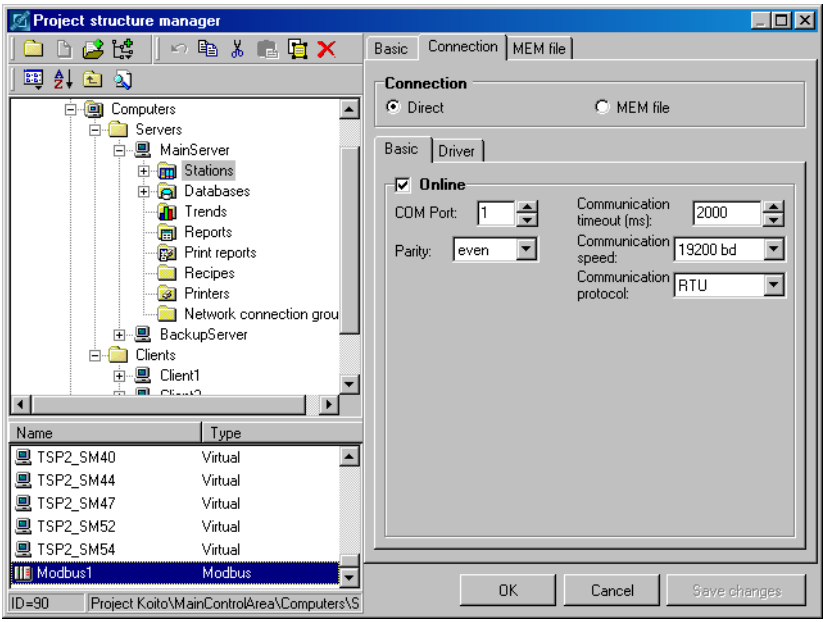
Basic



Adapter This parameter specifies the adapter number.

Connection parameters for Modbus stations

Basic



COM Port — This parameter specifies the number of COM port that is used for communication to the station via a serial cable.

Parity — This parameter specifies the parity used for communication to the station.

Communication timeout (ms)

This parameter specifies the maximum time period between sending a request to and receiving a response from the station by the communication driver. If the station 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 station again. If the station does not respond repeatedly, it is recognized as a failure in communication to the station.

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

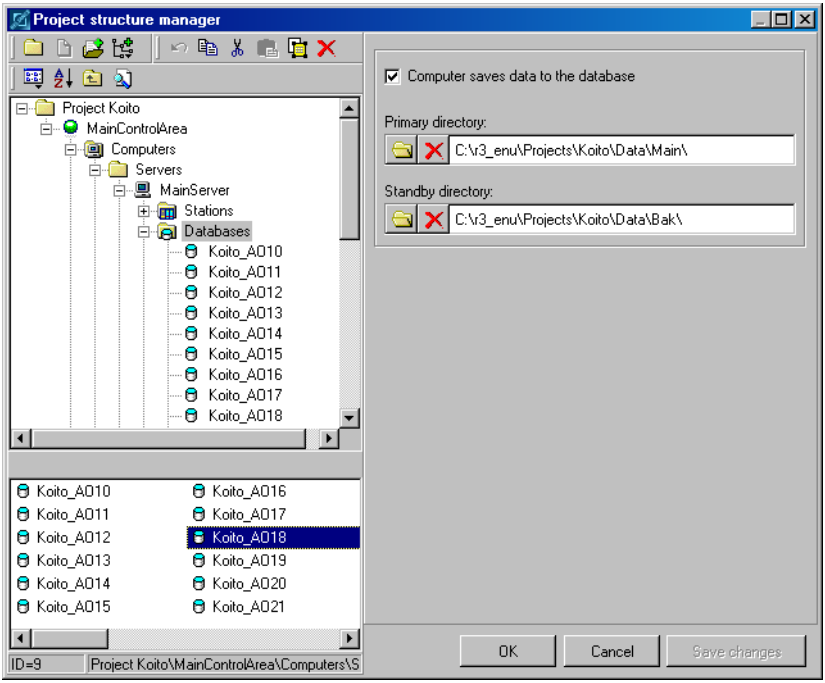
Communication speed This parameter specifies the communication speed in bauds.

👁 **WARNING** The maximum possible communication speed depends on the data transmission line and connection quality – the better the connection, the higher the communication speed may be.

Communication protocol This parameter specifies the protocol to be used for communication to the station.

2.6.2.5 Connecting databases

To make a database (previously defined via the ***Database manager***) accessible to the runtime module running on the actual computer, you have to add a link to the database object to the logical computer's *Databases* folder. This is called connecting a database to a computer.



Computer saves data to the database

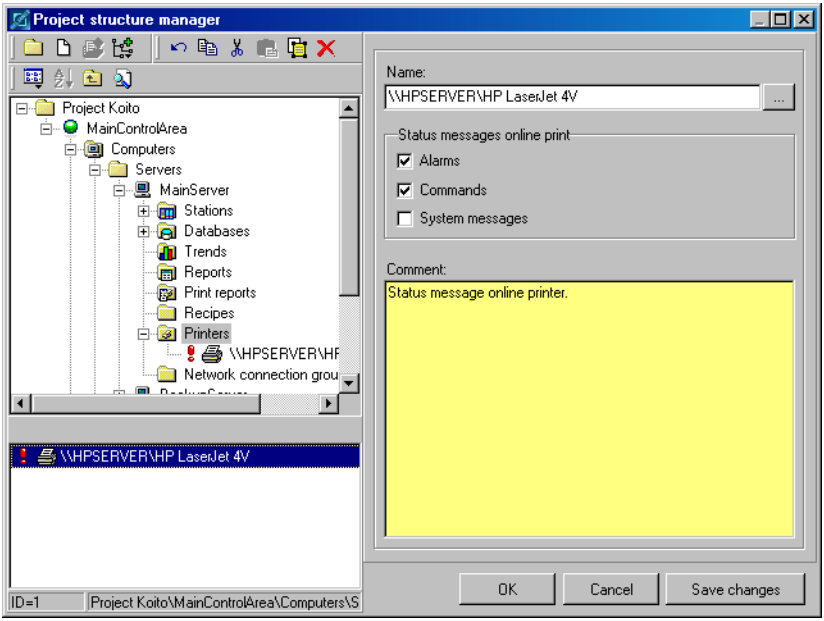
This option determines whether the runtime module running on the computer logs data to the database.

WARNING This option should only be active for one of the computers accessing the database. Usually, it is the one communicating to the station directly (i.e. via a communication driver). The other computers should only view the database via a computer network.

Primary directory This parameter specifies the primary directory for storing the database.

Standby directory This parameter specifies the standby directory for storing the database. It will only be used when the primary directory is not available.

2.6.2.6 Connecting printers for printing status messages



Name This parameter specifies the name of the printer in the operating system.

Status message online print

These options determine which types of status messages should be printed online (i.e. printed immediately when they occur). This feature is also affected by analogous global options (see the chapter 2.4.2 PROJECT OPTIONS). To make this feature operational, it is necessary that the printer supports printing a single line without ejecting the paper.

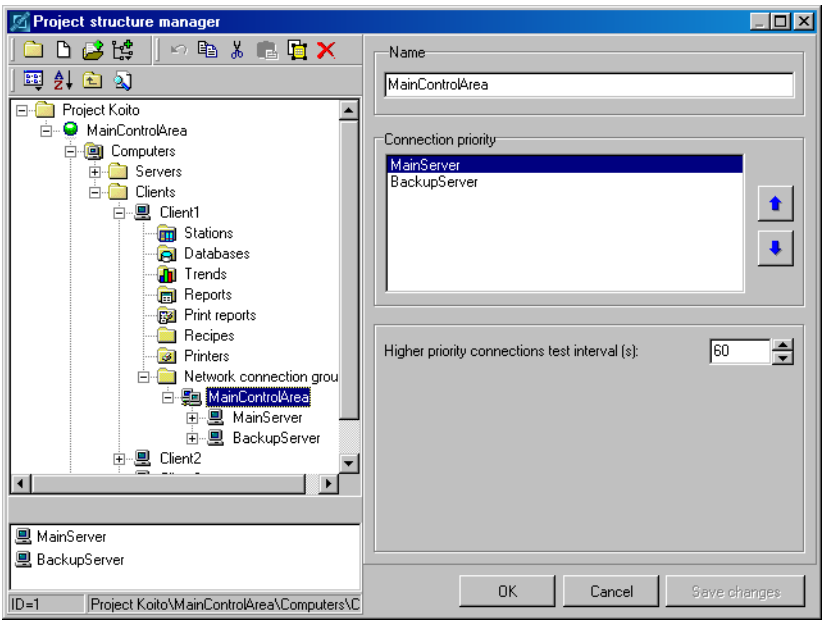
WARNING Laser printers and regular ink printers cannot be used for this purpose!

Comment This parameter can be used to specify an optional description of the printer.

2.6.2.7 Defining a network connection group

Network connections are the most common way to transfer data of a technological station between runtime modules running on different computers. A network connection always involves two computers: a client computer and server computer. The server computer is the one that has the data available (either through a communication driver or other network connection or a MEM file). The client computer is the one that needs to get the data from the server computer. The data is transferred through a so-called socket using the TCP/IP protocol. The socket is a communication channel identified by a unique number called a port.

To define a network connection between two computers, use the ***Project structure manager***. First, select the *Network connection groups* folder of the client computer and invoke the *New object* command from the toolbar. This will add a new network connection group to the folder and automatically select it. Now you can add one or more network 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 network connection to the group, you define a redundant configuration. This means that you define one primary and one or more standby (secondary) connections. The primary connection is the one with the highest priority. In the event of a communication failure on the current connection, the runtime module 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.



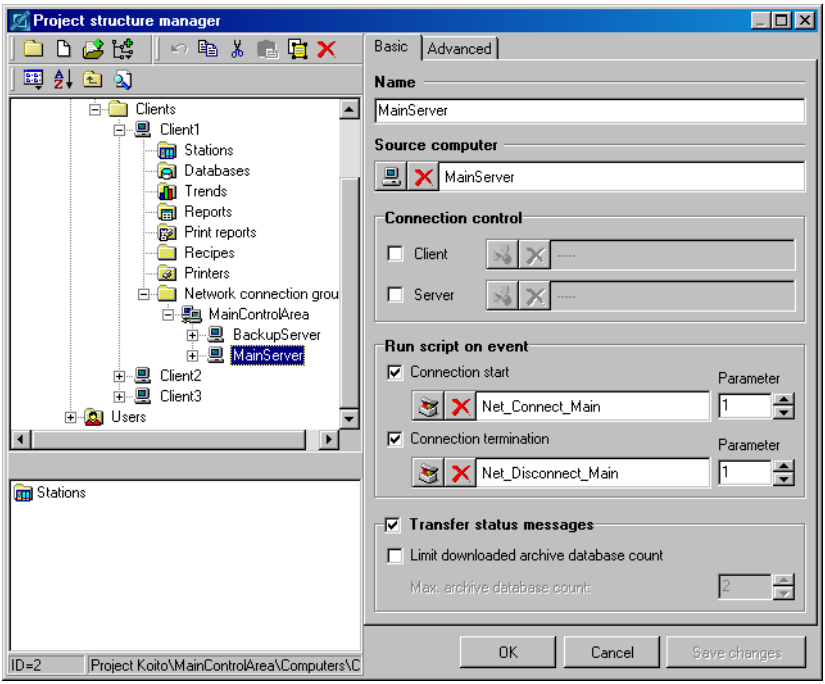
Name — This parameter specifies the network connection group's name that must be unique within the client computer and cannot contain illegal characters.

Connection priority — This parameter 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) — This parameter specifies the time interval at which the runtime module on the client computer checks for the availability of a higher-priority connection.

2.6.2.8 Defining a network connection

Basic



Name — This parameter specifies the connection's name that must be unique within the group and cannot contain illegal characters.

Source computer — This parameter specifies the server computer used as a data source for the client computer.

Connection control — These parameters specify whether to control communication on the connection by the specified binary-type variables on client and server runtime modules.

Run script on event

Connection start — These parameters determine whether to execute the specified script when communication on the connection is established.

Connection termination These parameters determine whether to execute the specified script when communication on the connection is terminated.

Transfer status messages This option determines whether to transfer status messages (alarms, commands and system messages) of the stations provided through this network connection. If this parameter is active, status messages of these stations are not generated on the client computer, but status messages generated on the server computer are accepted. Thus, both the client and server computers have the same status messages in the database (each computer has its own local database of status messages). The only difference is the receipt time of the messages.

Extended

Communication on a network connection between runtime modules is started at the client side. Unless communication at the client side is controlled by a variable, the client runtime module attempts to establish communication immediately after starting the visualizing project in runtime mode; otherwise only in case the connection control variable is equal to one.

The following situations may occur while establishing communication on a network 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 module is not running on it.

3) The server computer has been found within the network, the runtime module 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 variable).

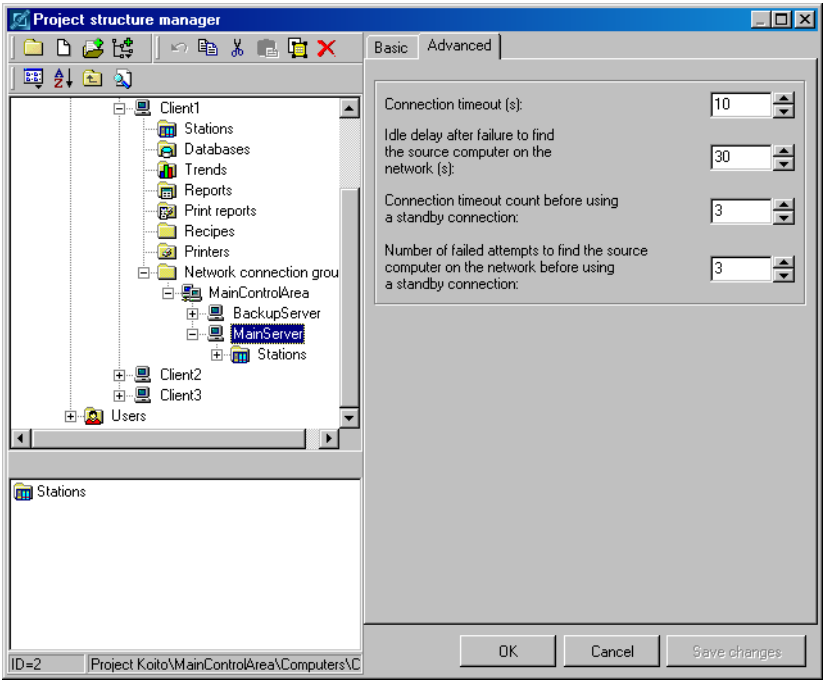
4) The server computer has been found within the network, the runtime module is running on it, and communication at the server side is enabled. In this case, communication is established successfully.

The first situation is usually caused by an incorrectly specified *IP address* (or system name, if a constant IP address cannot be specified) of the server computer in the visualizing project, or the fact, that the computer is not

running. Immediately after the attempt to establish communication, the client runtime module will qualify this situation as an unsuccessful attempt to find the server computer within the network.

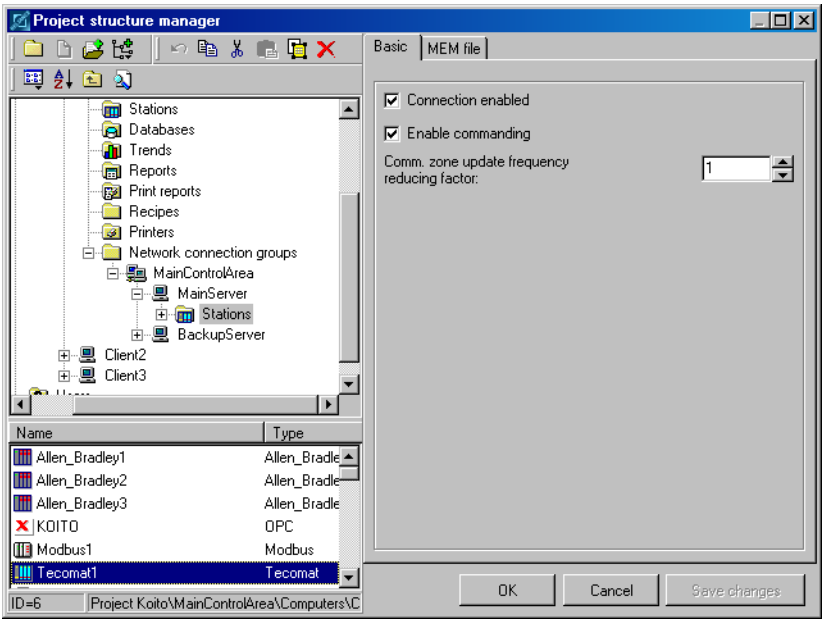
Next attempt to establish the connection will be made after expiration of the time period defined by the parameter *Idle delay after failure to find the source computer on the network (s)*. If this situation repeats more then *x*-times (the figure *x* is defined by the *Number of failed attempts to find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

The second and third situation will be qualified as an unsuccessful attempt to establish communication by the client runtime module after expiration of the time period defined by the *Connection timeout (s)* parameter, 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* parameter), a standby connection is activated (according to priorities), which is defined in the same network connection group.



2.6.2.9 Defining a station provided through a network connection

Basic



Connection enabled — This option determines whether to make the station accessible to the computer. If the option is not active, the runtime module acts as if the station was not connected to the computer (i.e. as if the link to the station were not contained in the *Stations* folder) and ignores this station and its subordinated objects when loading the visualizing project. It can be useful when debugging the application to speed up starting the project in runtime mode if concentrating only on certain stations.

Enable commanding — This option determines whether the client runtime module can send commands to the station through the network connection.

Data update interval (ms) — This parameter specifies the time interval used for updating the station's current data. It is intended for stations that do not support communication zones.

Comm. zone update frequency reducing factor

This parameter affects the time interval used for updating the station's current data by reading the station's communication zones via the network connection. It is intended for stations that support communication zones (*Tecomat*, *Tecoreg*, *Allen Bradley* and *Modbus*). If the value is equal to one, the update interval is the same as defined for the communication zone via the ***Station manager***. If the value is equal to *x*, where *x* is greater than one, the update interval is *x*-times greater than defined for the communication zone. Thus, the update frequency can be reduced.

MEM file

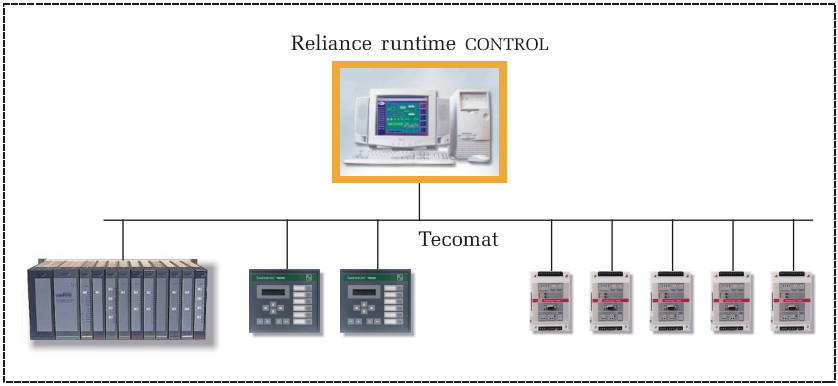
See the ***MEM file*** section in the chapter 2.6.2.4 CONNECTING STATIONS.

2.6.2.10 Examples

This chapter contains a few examples to demonstrate various structures of visualizing projects; the dashed line indicates individual control areas.

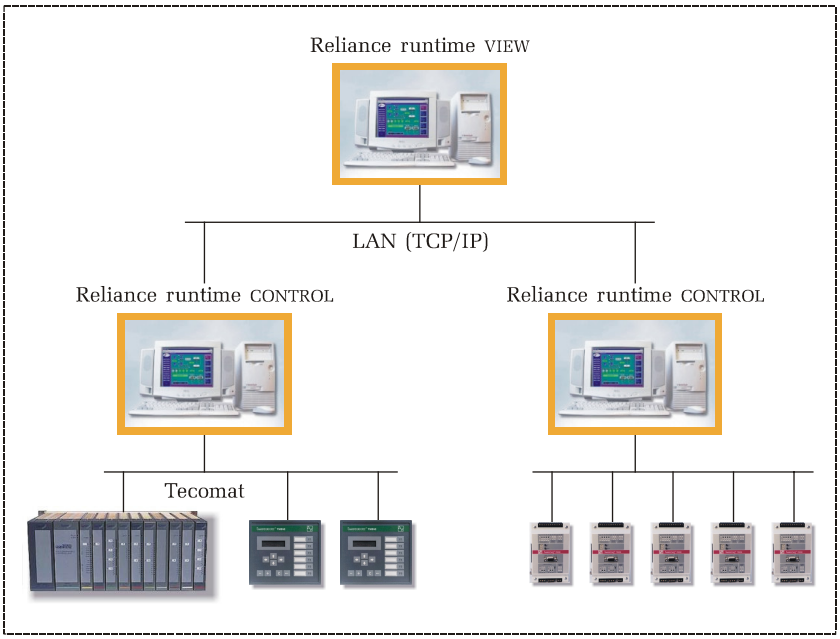
Example 1

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



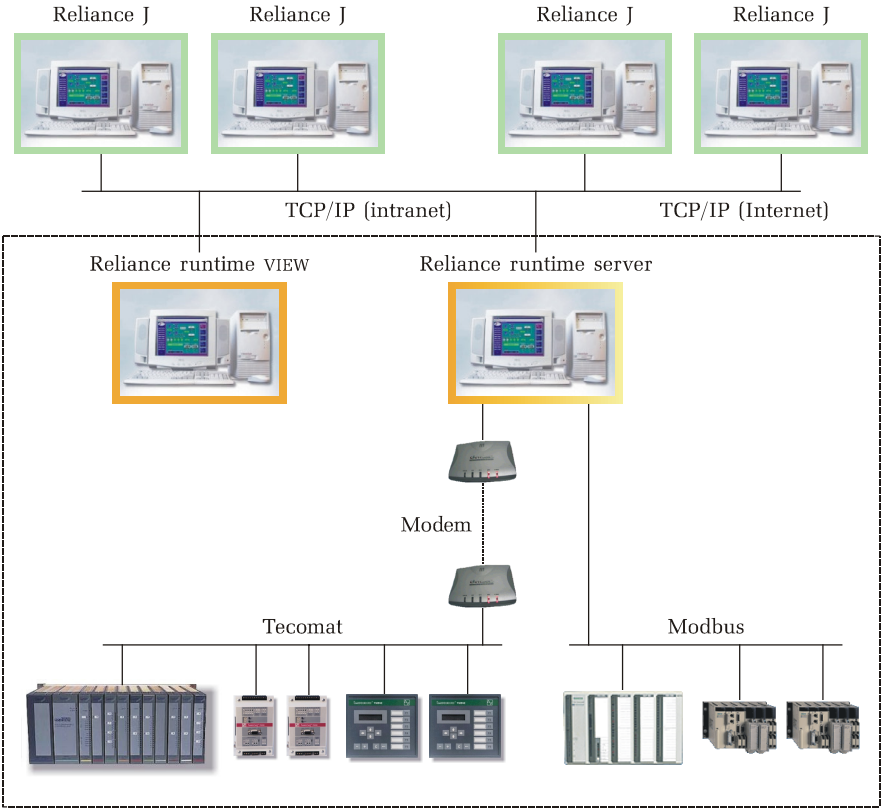
Example 2

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



Example 3

This example demonstrates an application that uses network communication between runtime modules and distribution of technological data to *Reliance J* Web clients. The visualizing project contains a single control area. The control area consists of two computers – a computer running the *Reliance runtime server* module and a computer running the *Reliance runtime VIEW* module. The computer running *Reliance runtime server* is connected to the network of *Tecomat* stations through a switched telephone line and to the network of *Modbus* stations through a serial cable. *Reliance runtime server* also enables the *Reliance J* Web clients to connect to it from a local intranet network or over the Internet. *Reliance runtime VIEW* obtains technological data from *Reliance runtime server* through a LAN network.



Example 4

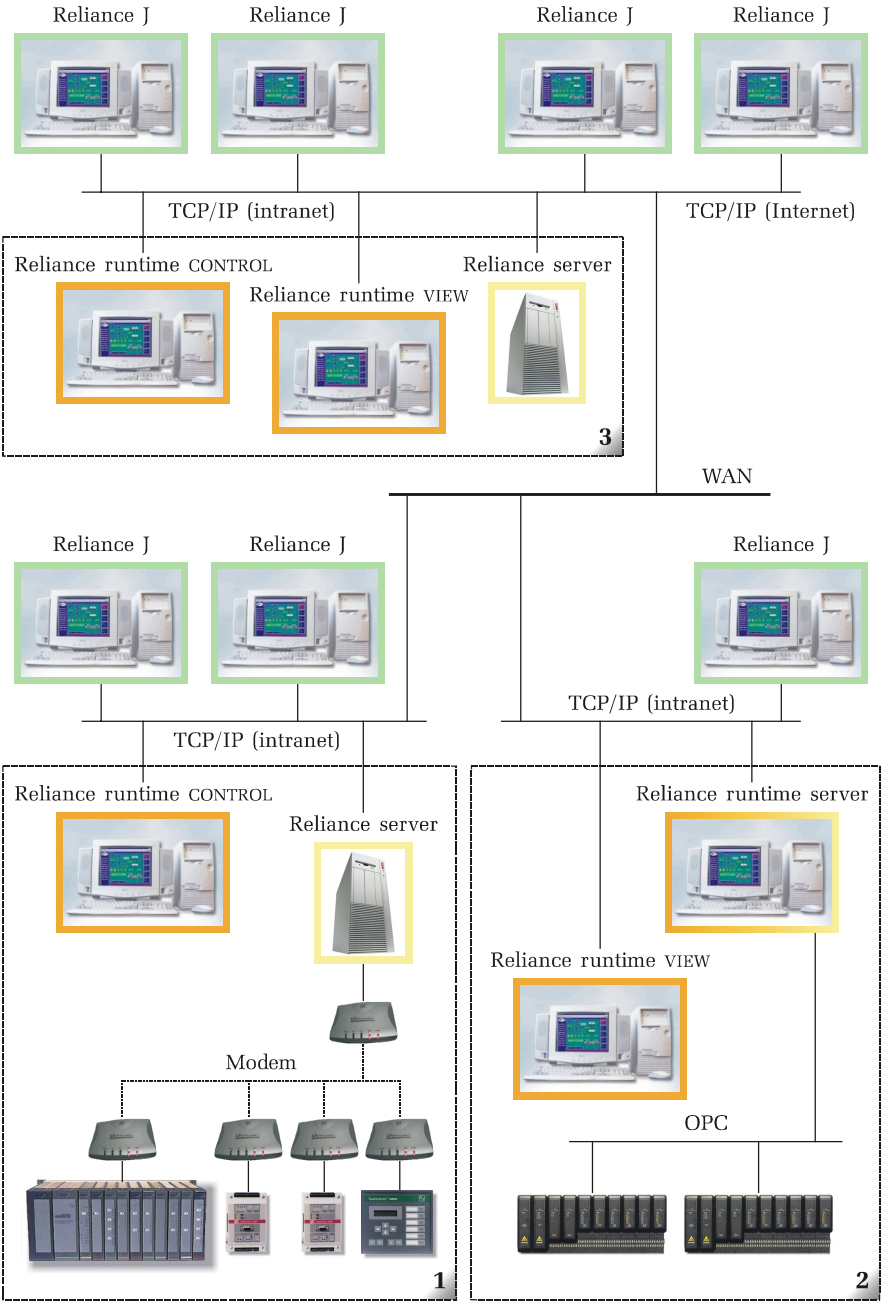
This example demonstrates a major application divided into three control areas.

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

The control area no. 2 consists of two computers. The computer running the *Reliance runtime server* module is connected to the stations through a serial cable. The *Reliance runtime VIEW* module running on the other computer obtains data of the stations from *Reliance runtime server* through a LAN network and allows monitoring the stations.

The control area no. 3 is a central control area collecting technological data from subordinated locations. It consists of three computers. The computer running the *Reliance server* module is connected to the other control areas through a WAN network. *Reliance server* obtains technological data from the runtime modules running on computers within the other control areas. The *Reliance runtime* modules running on the other computers within this control area obtain data of the stations from *Reliance server* through a LAN network and allow monitoring (both *VIEW* and *CONTROL* modules) and controlling (only the *CONTROL* module) the stations from the other control areas (1 and 2). When a command is sent to a technological station from *Reliance runtime CONTROL*, it is passed to *Reliance server* within this control area through the LAN network and then it is passed to a runtime module within the appropriate control area.

Reliance server and *Reliance runtime server* also enable the *Reliance J* Web clients to connect to them from a local intranet network or over the Internet.


Reliance J
Reliance J
Reliance J
Reliance J
TCP/IP (intranet)
TCP/IP (Internet)
Reliance runtime CONTROL
Reliance runtime VIEW
Reliance server
3
WAN
Reliance J
Reliance J
Reliance J
TCP/IP (intranet)
TCP/IP (intranet)
Reliance runtime CONTROL
Reliance server
Reliance runtime server
Reliance runtime VIEW
Modem
OPC
1
2

2.6.3 Station manager

The station is an object representing a physical device (e.g. PLC, programmable regulator, telemetric station or other I/O device) or a so-called virtual station. Within each station, you can define variables and status messages linked to the variables. The values of the variables are used to generate status messages (e.g. alarms), can be displayed via visual components, can be used to control the appearance and behavior of components, can be archived to databases, processed by scripts, etc. If the station represents a physical device, the runtime module obtains the values of the variables from the station's communication driver or another runtime module (e.g. through a network connection). Otherwise, the station's variables only exist in the memory of the computer on which the visualizing project is running and are referred to as internal variables.

Every visualizing project contains a predefined virtual station named *System*. This station is automatically accessible to each computer defined within the visualizing project (i.e. you do not need to connect this station to the logical computer via the ***Project structure manager***). It is intended for defining private internal variables. It cannot be shared among the computers in any way (i.e. cannot be provided through a network connection, etc.).

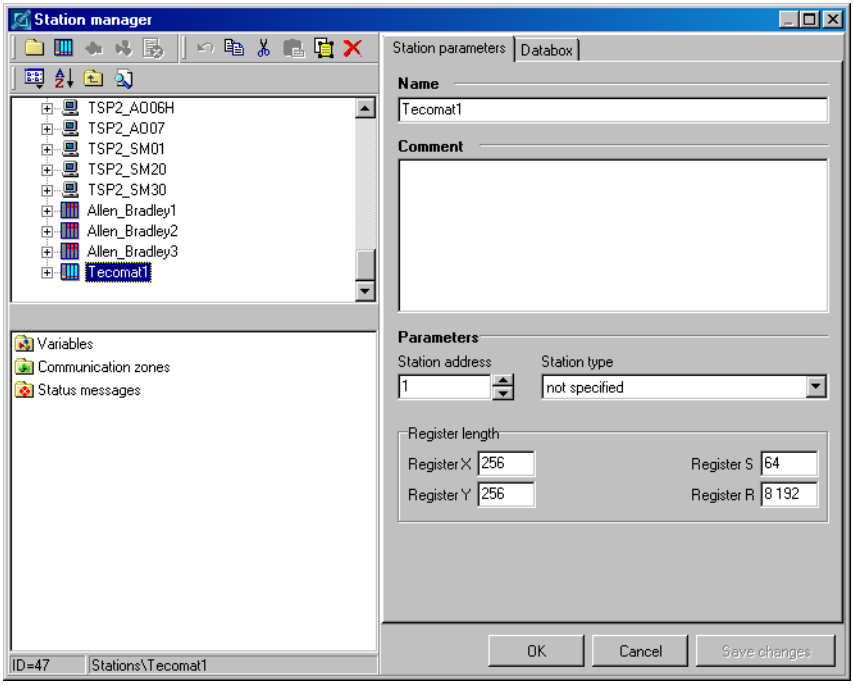
2.6.3.1 Defining a station

Name — This parameter specifies the station's name that must be unique within the project and cannot contain illegal characters.

Comment — This parameter can be used to specify an optional description of the station.

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

Defining a Tecomat and Tecoreg station



Station parameters

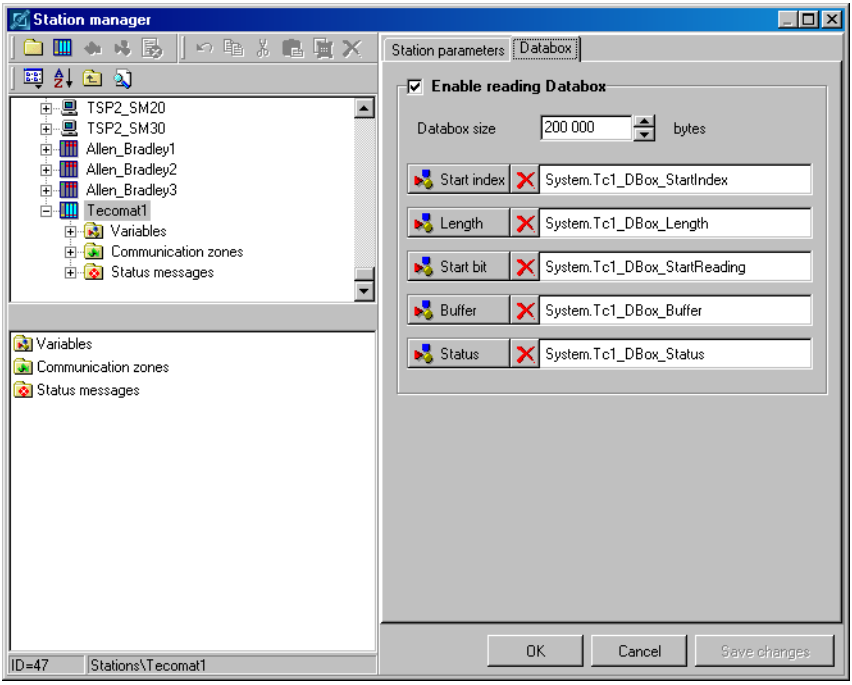
Parameters

Station address — This parameter specifies the station's address set on the communication channel used.

Station type — This parameter can be used to specify the station's type. In most cases, it is not required.

Register length — These parameters specify the sizes (in bytes) of the station's registers to cover the address space used for the variables defined within the station. These values do not need to correspond to the sizes of registers of the physical station.

Databox



Enable reading Databox This parameter determines whether to enable the communication driver to read data from the Databox (supplementary memory of *Tecomat* stations).

Databox size This parameter specifies the size of the Databox in bytes.

Start index This parameter specifies the integer-type variable to be used to control the initial address of the data block to be read from the station.

Length This parameter specifies the integer-type variable to be used to control the length of the data block to be read from the station.

Start bit This parameter specifies the binary-type variable to be used to control the reading. The rea-

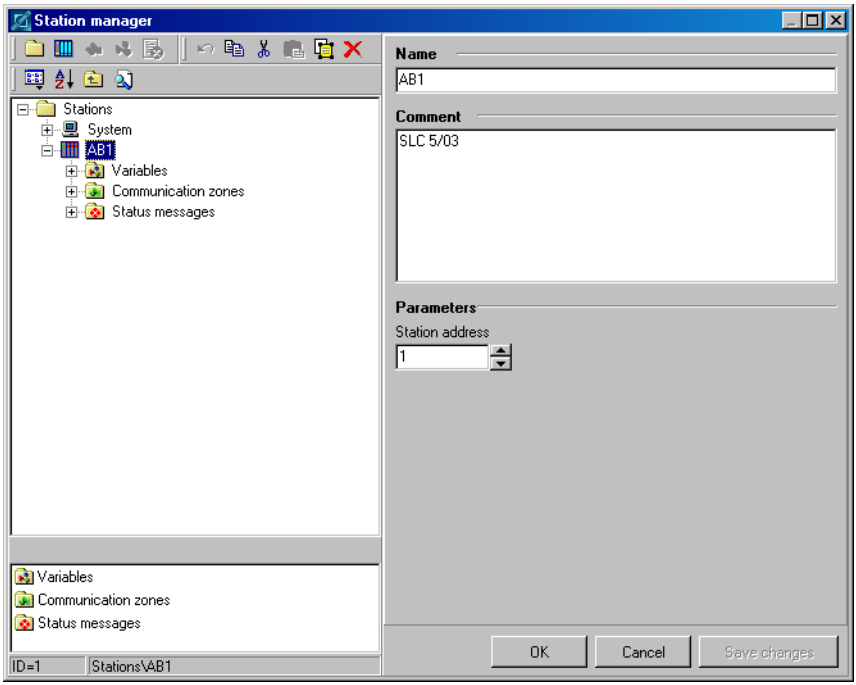
ding starts on the leading edge of the variable (the off-to-on transition).

Buffer This parameter specifies the array-type variable to be used to store the data block read from the Databox.

Status This parameter specifies the integer-type variable to be used to store the current status of the reading operation. It may contain the following values:

0 ...ready
1 ...the reading operation in progress
2 ...the reading operation completed successfully
3 ...the reading operation failed

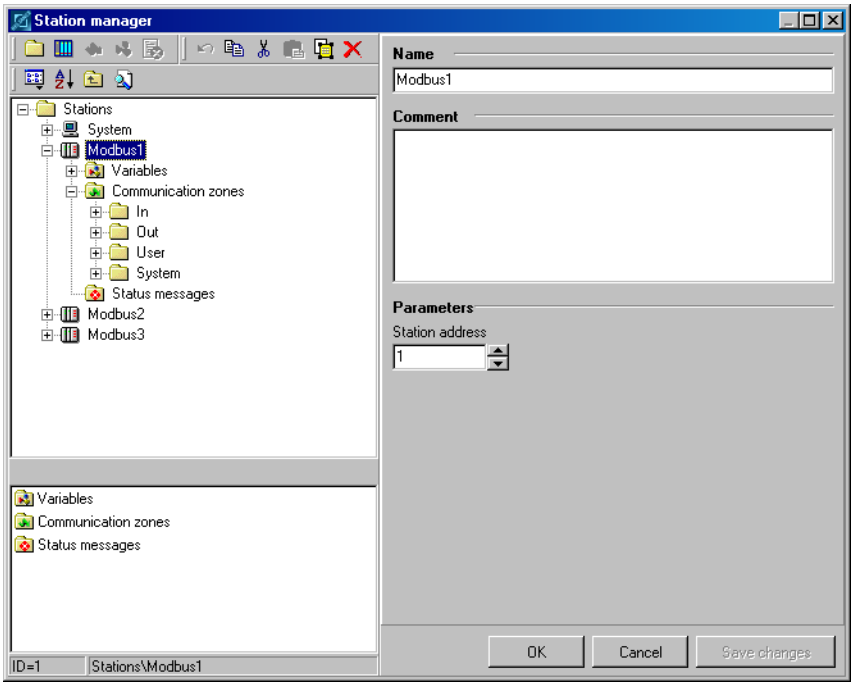
Defining an Allen Bradley station



Parameters

Station address This parameter specifies the station's address set on the communication channel used.

Defining a Modbus station



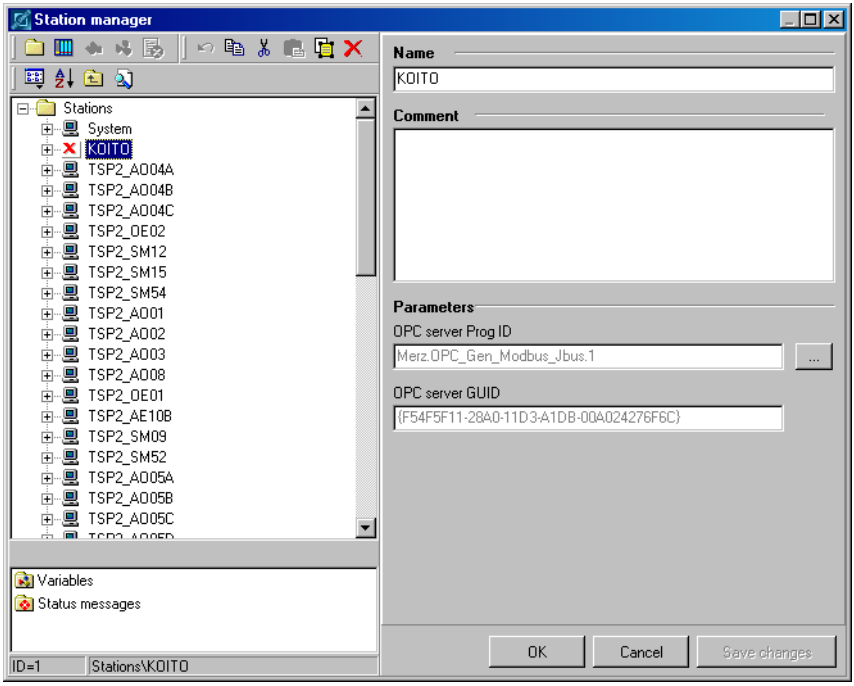
Parameters

Station address This parameter specifies the station's address set on the communication channel used.

Defining an OPC station

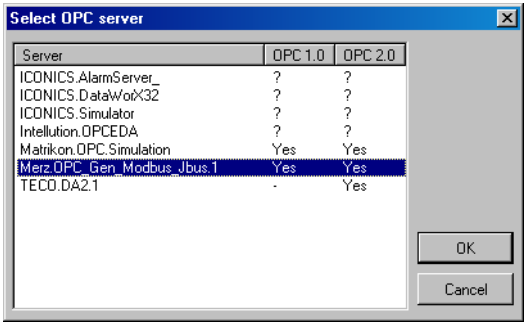
OPC (OLE for Process Control) is a worldwide standard of technological data exchange between software programs. Defining an *OPC station* allows *Reliance* to connect to any *OPC server* that meets the specification OPC DA 1.0 or 2.0. It means that if the manufacturer of a HW device (or a third party company) develops an *OPC server* for the device, then *Reliance* can use the *OPC server* to obtain technological 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 application (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*).



Parameters

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



OPC server GUID This parameter specifies a unique identifier of the *OPC server*. This parameter is supplied automatically after selecting the *OPC server*. To select an *OPC server*, click the ellipsis (...) button.

2.6.3.2 Importing and exporting variables

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

Teco

Variables of *Tecomat* and *Tecoreg* stations may be imported from a file in the **.pub* (xPro) or **.tdr* format (Mercur, Epos for Windows). A structure-type variable (which can be defined via Mercur or Epos) is imported as a group of variables corresponding to the structure items. Names of the variables are generated according to the syntax ***StructureName_ItemName***. An array-type variable (which can be defined via Mercur, Epos) is imported as a group of variables corresponding to the elements of the array. Names of the variables are generated according to the syntax ***FieldName_ElementIndex***.

Allen Bradley

Variables of *Allen Bradley* stations may be imported from a file in the **.csv* format (RS Logix) or exported in this format.

OPC

To import variables into the current *OPC group* directly from the *OPC server*, click the *Import variables from OPC server* button. However, the server must support the *IOPCBrowseServerAddressSpace* interface. Variables

may be imported to the current *OPC group* also from a file in the **.csv* format or exported in this format. The file must be in text format; each line must have the following structure:

OpcItemID; Variable name; Variable type; Comment

2.6.3.3 Defining a variable

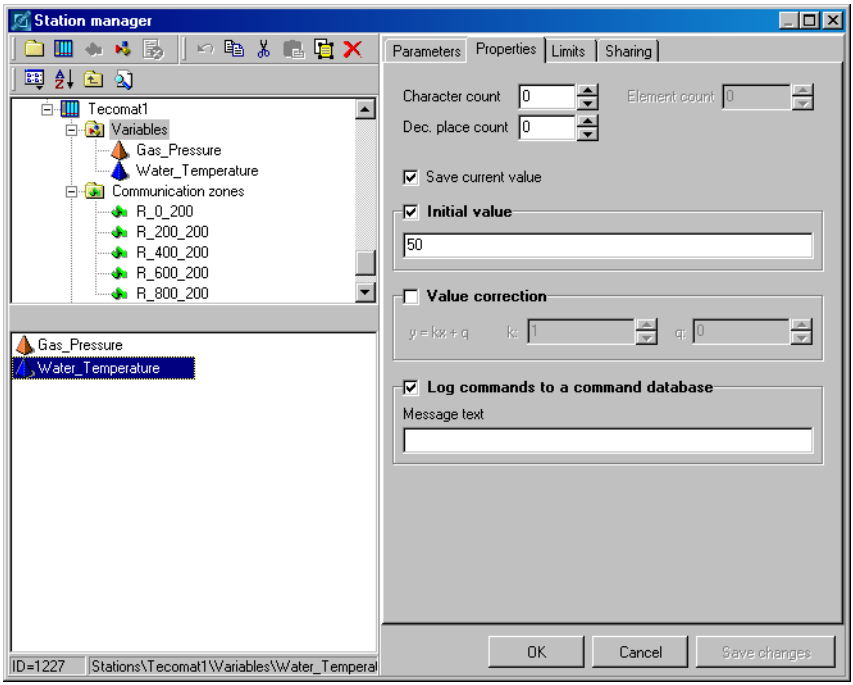
Parameters

Name — This parameter specifies the variable's name that must be unique within the station and cannot contain illegal characters.

Comment — This parameter can be used to specify an optional description of the variable.

Other parameters of variables (e.g. variable type) differ according to station type.

Parameters



Character count This parameter specifies the minimum number of characters displayed. If the number of characters of the value displayed is less than this parameter, the value will be filled in by space characters – for numeric-type variables from the left, for string-type variables from the right.

Dec. place count This parameter specifies the number of decimal places displayed. For a floating point-type variable, this is the number of decimal places the value will be rounded up to; in case of an integer-type variable, the value will be calculated using the formula $y = x/10^n$, where n is the value of this parameter. The runtime module uses the corrected value and the original value is not available anymore.

Element count This parameter specifies the number of elements of array-type variables.

Save current value This option determines whether to save the current value of the variable before terminating the visualizing project in runtime mode; when the project is started next time, the saved value is used to initialize the variable.

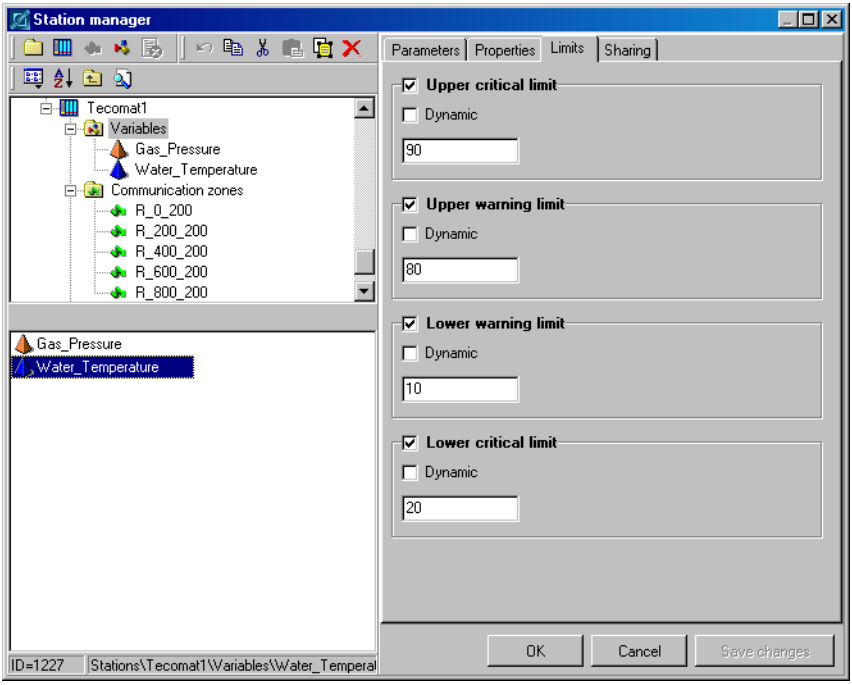
Initial value This option determines whether to initialize the variable with the specified value when starting the visualizing project in runtime mode. This value may later be replaced with the value saved as current (see the *Save current value* parameter).

Value correction This option determines whether to use a correction according to the formula $y = kx + q$ when computing the value of the variable. If this option is active, the runtime module uses the corrected value and the original value is not available anymore.

Log commands to a command database
This option determines whether to record information about all "write" operations performed on the variable to the database of commands. The *Message text* parameter is optional. If it is not specified, a default text will be used.

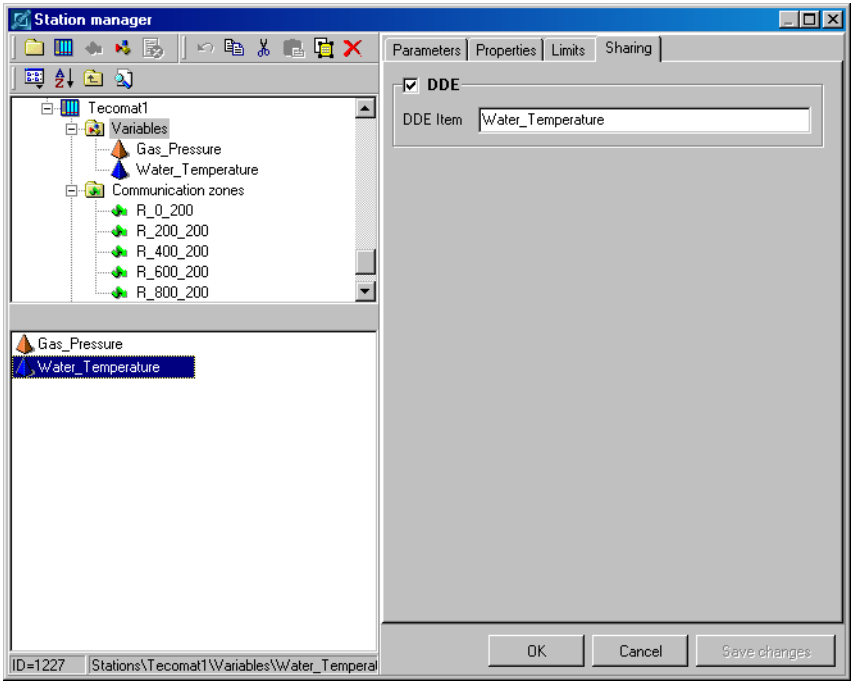
Limits

These parameters let you define upper and lower limits (critical and warning) for the variable. Limits may be either static or dynamic. The value of a static limit is defined as a constant parameter. The value of a dynamic limit is controlled by the specified numeric-type variable. A limit can be used to generate an alarm message when exceeding the limit.



Sharing

The runtime module of *Reliance* acts as a *DDE server*.



DDE This option determines whether the runtime module provides other programs (the so-called *DDE clients*, e.g. MS Excel, MS Word) with the current value of the variable using the DDE (Dynamic Data Exchange) standard.

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

{=runtime_module|DdeServer!DdeItem}

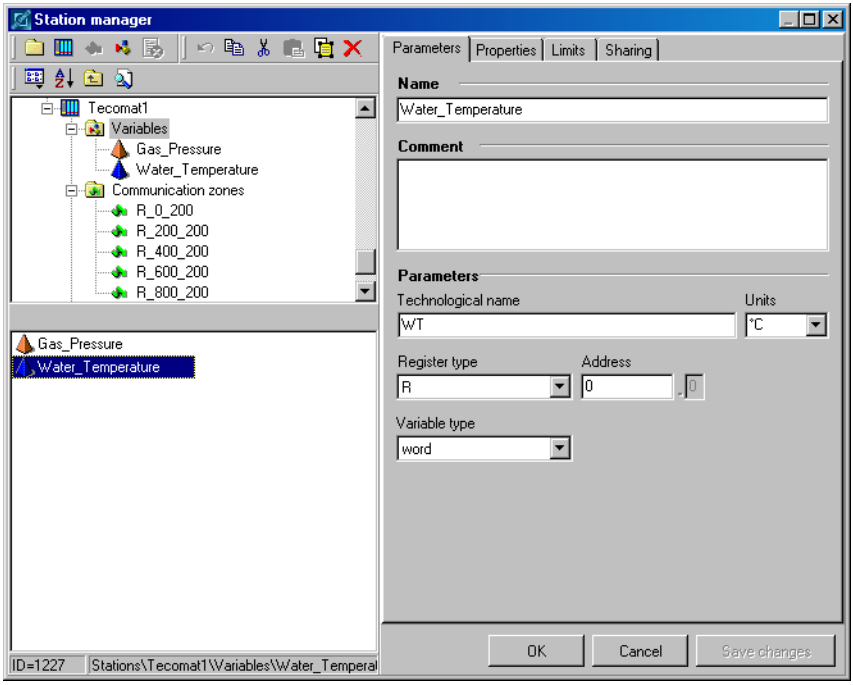
runtime_module .....the runtime module's file name without extension (*Reli_rt*, *Reli_rts*, *Reli_srv*)

DdeItem.....the name specified as the *DDE item* parameter

🗝 **EXAMPLE**

{=Reli_rt|DdeServer!Water_Temperature}

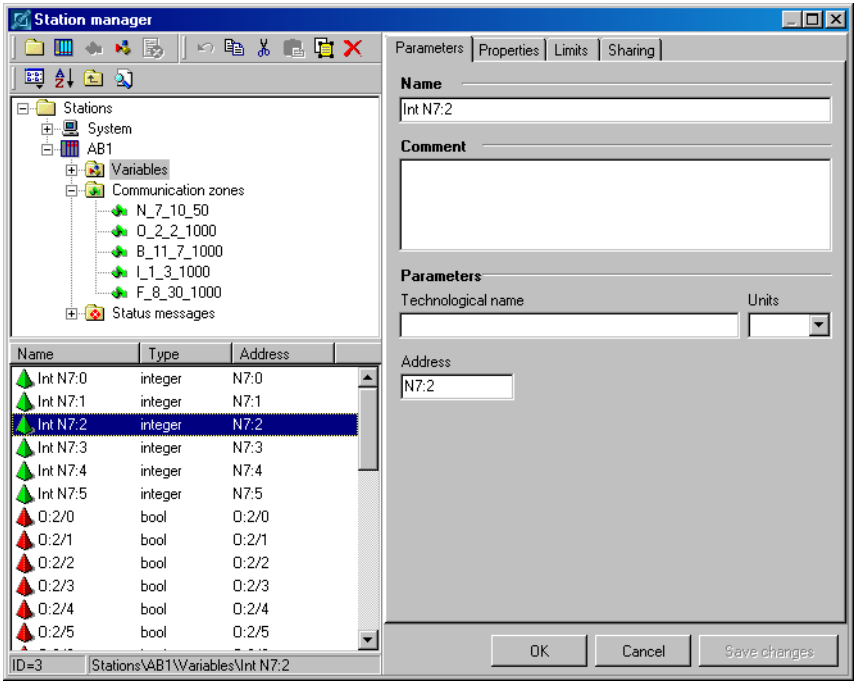
Defining a variable of a Tecomat and Tecoreg station



Parameters

Technological name	This parameter specifies an optional name related to the technology for the variable.
Units	This parameter specifies units of measurement for the variable.
Register type	This parameter specifies the type of register where the variable is stored in the station.
Address	This parameter specifies the register address where the variable is stored in the station.
Variable type	This parameter specifies the data type (format) for the variable.

Defining a variable of an Allen Bradley station



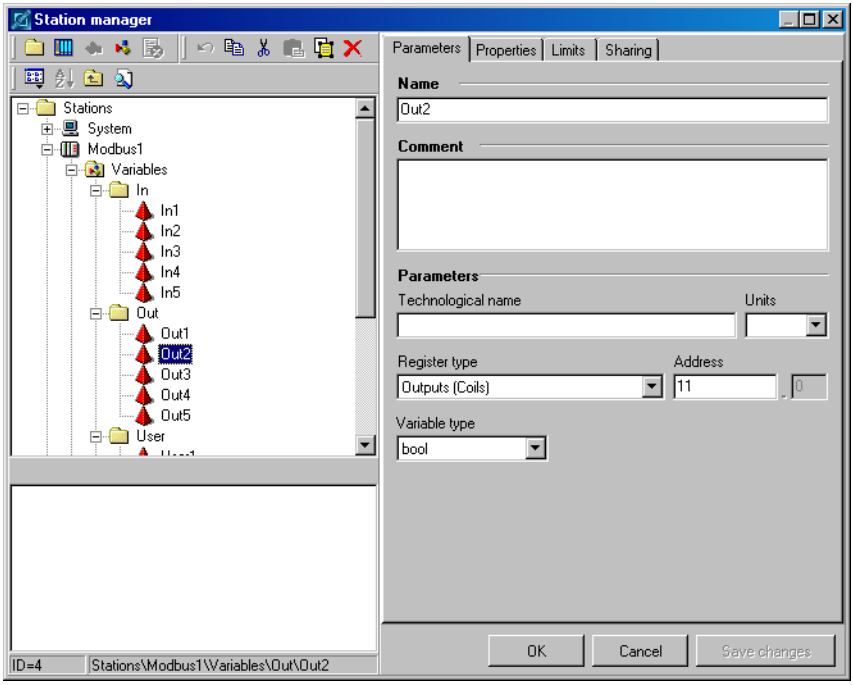
Parameters

Technological name — This parameter specifies an optional name related to the technology for the variable.

Units — This parameter specifies units of measurement for the variable.

Address — This parameter specifies the register address where the variable is stored in the station. The address consists of a register block type and an element address within this block. Variables stored in B, I, O, N, F and ST blocks are supported.

Defining a variable of a Modbus station



Parameters

Technological name — This parameter specifies an optional name related to the technology for the variable.

Units — This parameter specifies units of measurement for the variable.

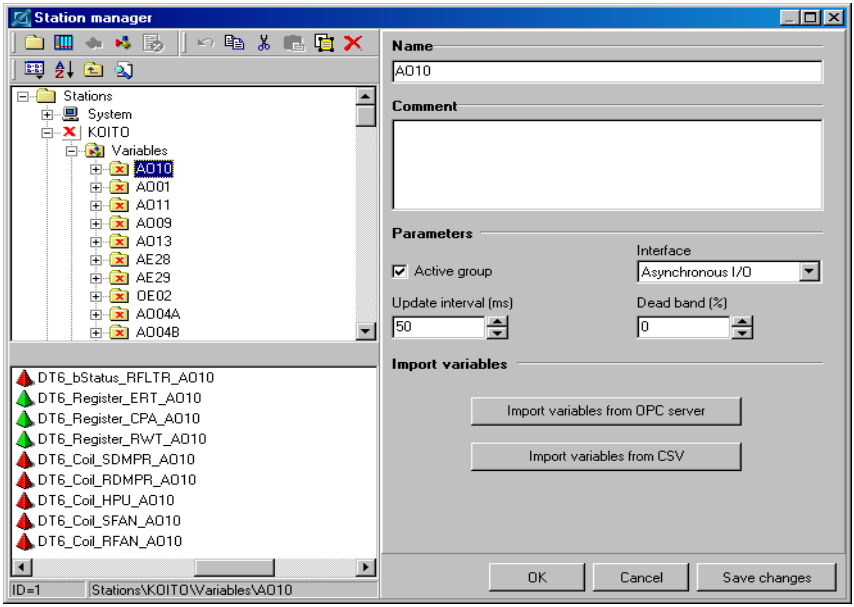
Register type — This parameter specifies the type of register where the variable is stored in the station. A *Modbus* station has 4 types of registers – Outputs (*Coils*), Inputs (*Inputs*), User registers (*Holding registers*) and System registers (*Input registers*).

Address — This parameter specifies the register address where the variable is stored in the PLC.

Variable type — This parameter specifies the data type (format) for the variable. The *Bool* format is supported only for variables in input and output registers, the *Bool*, *Word* and *Integer* formats are supported for variables in user and system registers.

Defining a variable of an OPC station

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



Name — This parameter specifies the group's name that must be unique within the station and cannot contain illegal characters.

Comment — This parameter can be used to specify an optional description of the group.

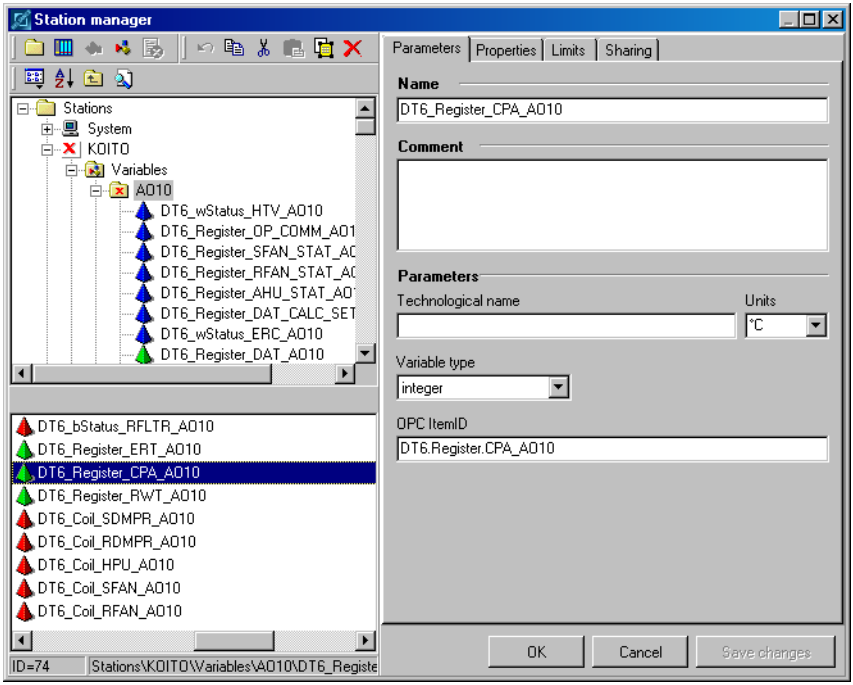
Parameters

Active group — This option determines whether the group is active, i.e. whether the variables defined within the group should be periodically updated by the *OPC server*.

Update interval (ms) — This parameter specifies the required update interval of variables by the *OPC server*. Certain *OPC servers* do not support arbitrary update intervals – it is necessary to study the documentation of the *OPC server* and use the closest interval supported.

Interface — This parameter specifies the interface for reading data from the *OPC server*. Older servers support the standard OPC DA 1.0, newer servers have also OPC DA 2.0 interface available. If *Asynchronous I/O* is selected, the data is obtained by the standard asynchronous interface.

Dead band (%) — This parameter specifies the percentage of change in the value of a variable defined within the group in order for the *OPC server* to send the new value of the variable to the runtime module.



Parameters

Technological name	This parameter specifies an optional name related to the technology for the variable.
Units	This parameter specifies units of measurement for the variable.
Variable type	This parameter specifies the data type (format) for the variable.
OPC ItemID	This parameter specifies the name used for the variable within the *OPC server*.

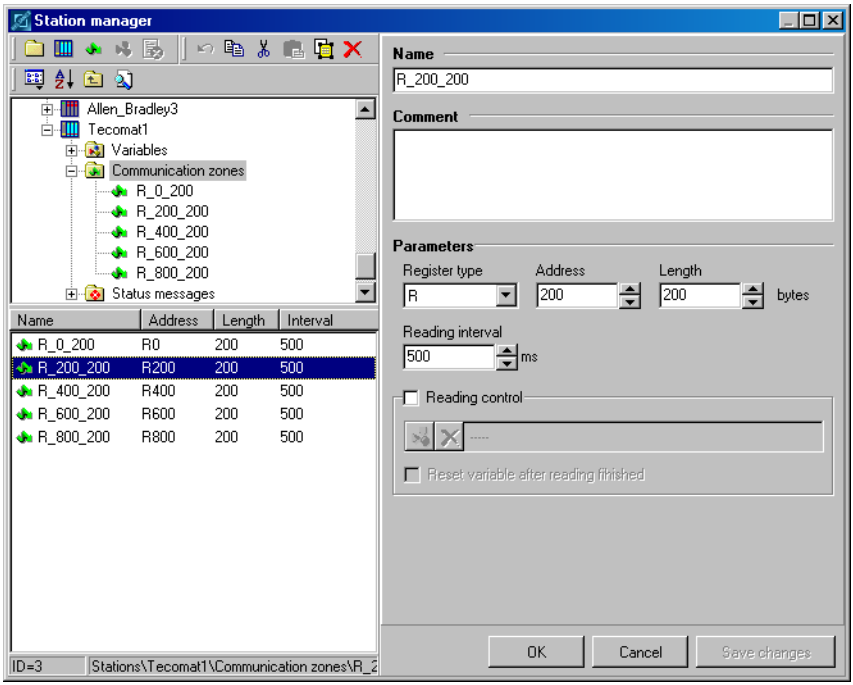
2.6.3.4 Defining a communication zone

Communication zones let you fully control the communication to the stations.

Name	This parameter specifies the zone's name that must be unique within the station and cannot contain illegal characters.

Comment This parameter can be used to specify an optional description of the zone.

Defining a communication zone of a Tecomat and Tecoreg station



Parameters

Register type This parameter specifies the type of register where the zone is located.

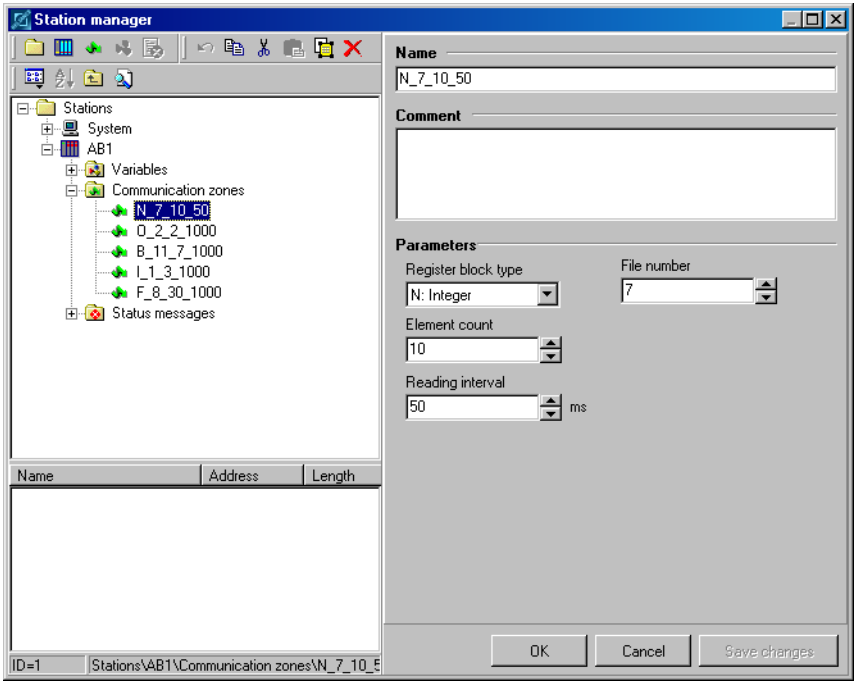
Address This parameter specifies the register address where the zone begins.

Length This parameter specifies the length of the zone in bytes.

Reading interval This parameter specifies the time interval used for reading the zone by the communication driver.

Reading control — This parameter determines whether to control reading the zone by the specified binary-type variable.

Defining a communication zone of an Allen Bradley station



Parameters

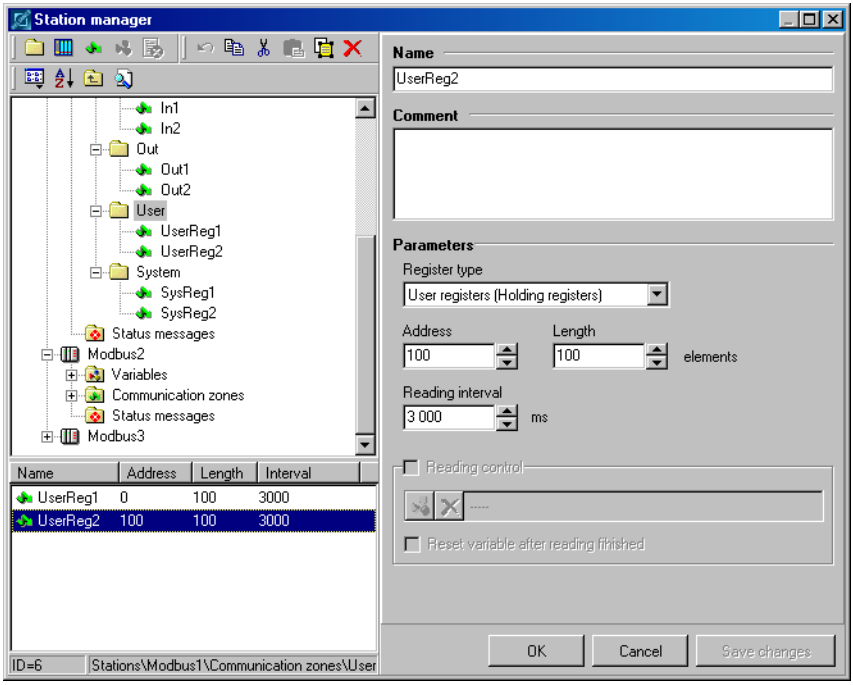
Register block type — This parameter specifies the type of register block where the zone is located.

File number — This parameter specifies the number of file (register block) where the zone is located.

Element count — This parameter specifies the length of the zone in elements.

Reading interval — This parameter specifies the time interval used for reading the zone by the communication driver.

Defining a communication zone of a Modbus station

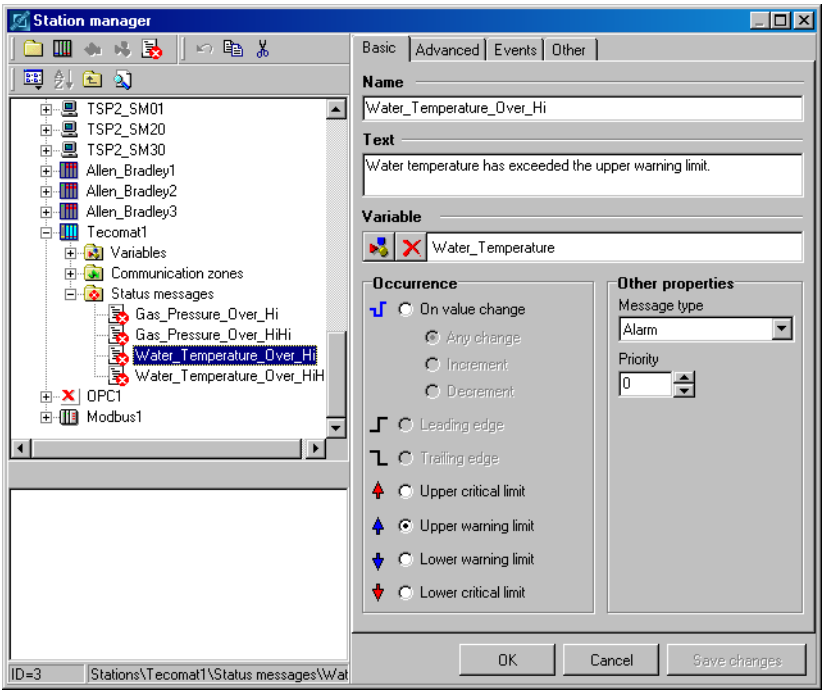


Parameters

Register type	This parameter specifies the type of register where the zone is located.
Address	This parameter specifies the register address where the zone begins.
Length	This parameter specifies the length of the zone in elements.
Reading interval	This parameter specifies the time interval used for reading the zone by the communication driver.
Reading control	This parameter determines whether to control reading the zone by the specified binary-type variable.

2.6.3.5 Defining a status message

Basic



Name — This parameter specifies the message's name that must be unique within the station and cannot contain illegal characters.

Text — This parameter specifies the text of the message.

Variable — This parameter specifies the link to the variable related to the message. The variable must belong to the same station as the message.

Occurrence — This parameter specifies the occurrence that generates the message.

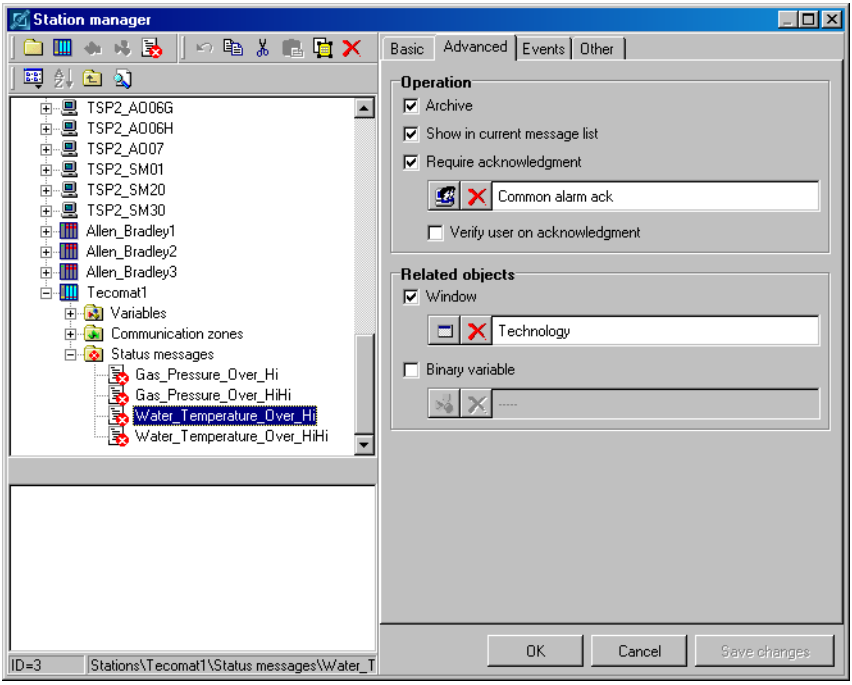
On value change — The message is generated when the value of the variable changes in the specified way (*Any change*, *Increment*, *Decrement*).

Leading edge	The message is generated when the value of the binary-type variable changes from 0 to 1 (the off-to-on transition). The message remains active until the variable changes its value back to 0.
Trailing edge	The message is generated when the value of the binary-type variable changes from 1 to 0 (the on-to-off transition). The message remains active until the variable changes its value back to 1.
Upper critical limit	The message is generated when the value of the variable becomes equal or greater than the upper critical limit of the variable.
Upper warning limit	The message is generated when the value of the variable becomes equal or greater than the upper warning limit of the variable.
Lower warning limit	The message is generated when the value of the variable becomes equal or less than the lower warning limit of the variable.
Lower critical limit	The message is generated when the value of the variable becomes equal or less than the lower critical limit of the variable.

Other properties

Message type	A status messages can be of one of the following types: alarm, command and system message. The runtime module creates a separate database for each of the types.
Priority	This parameter affects the order in which the sounds triggered by the message's events (i.e. origin and end) should be played. The message with higher value of this parameter has higher priority when playing the sounds.

Advanced



Operation

Archive This option determines whether to save the message to the database of status messages.

Show in current message list
This option determines whether to display the message in the list of current messages.

Require acknowledgement This option determines whether acknowledgement by the operator is required for the message before it can be removed from the list of current messages.

Verify user on acknowledgement
This option determines whether to verify the operator when acknowledging the message. If this option is active, the message can only be acknow-

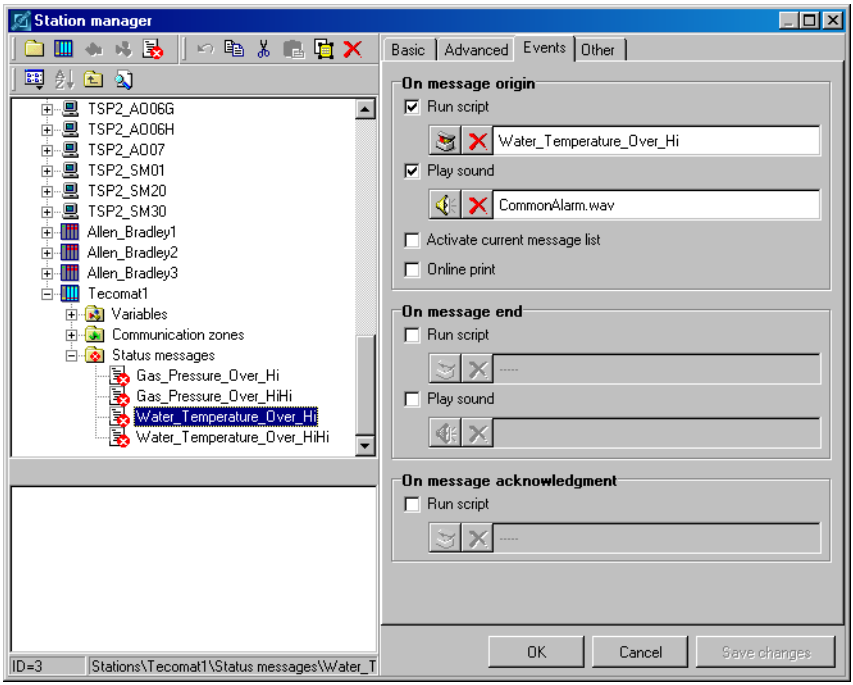
ledged upon successful verification of the operator's identity. A HW fingerprint sensor connected to the computer can perform the verification.

Related objects

Window — This option determines whether the specified window is related to the message. If this option is active, the operator can bring up the window by choosing a special command.

Binary variable — This option determines whether the specified binary-type variable is related to the message. The variable is set to 1 and kept on this value by the runtime module while the condition that generated the message persists. If the condition is not present, the variable is set to 0 and kept on this value by the runtime module.

Events



On message origin

Run script — This option determines whether to run the specified script when the message is generated.

Play sound — This option determines whether to play the specified sound file (in the **.wav* format) when the message is generated. If this option is active and the sound file is not specified, the runtime module uses the default sound (see the chapter 2.4.2.1 STATUS MESSAGE MANAGEMENT).

Activate current message list — This parameter determines whether to activate the list of current messages when the message is generated.

Online print — This parameter determines whether to print the message on the printer (intended for online printing, see the chapter 2.6.2.6 CONNECTING PRINTERS FOR PRINTING STATUS MESSAGES) when the message is generated. This feature is subject to global options for online print (see the chapter 2.4.2.1 STATUS MESSAGE MANAGEMENT).

On message end

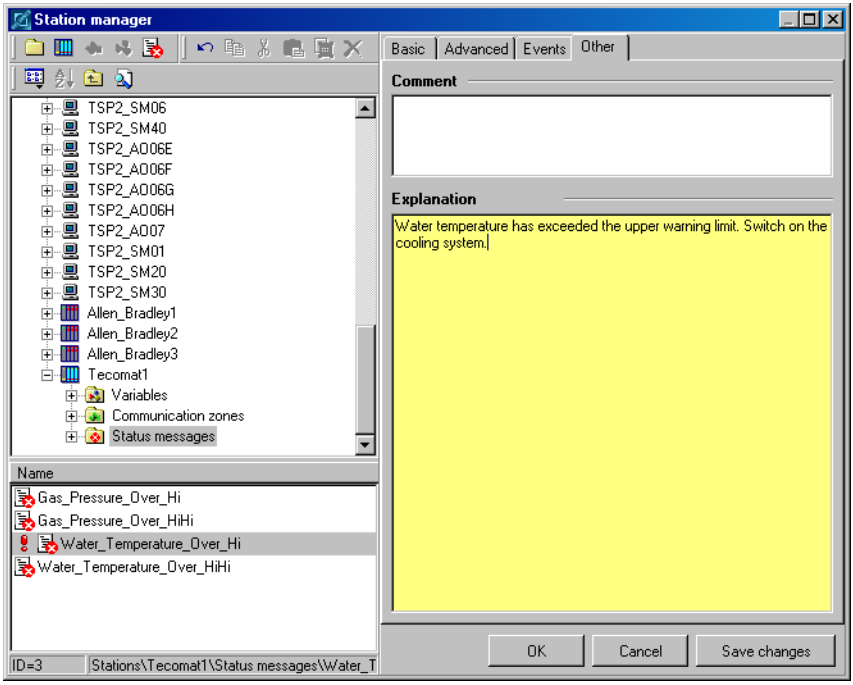
Run script — This option determines whether to run the specified script when the condition that generated the message ceases to exist.

Play sound — This option determines whether to play the specified sound file (in the **.wav* format) when the condition that generated the message ceases to exist. If this option is active and the sound file is not specified, the runtime module uses the default sound (see the chapter 2.4.2.1 STATUS MESSAGE MANAGEMENT).

On message acknowledgement

Run script — This option determines whether to run the specified script when the message is acknowledged by the operator.

Other



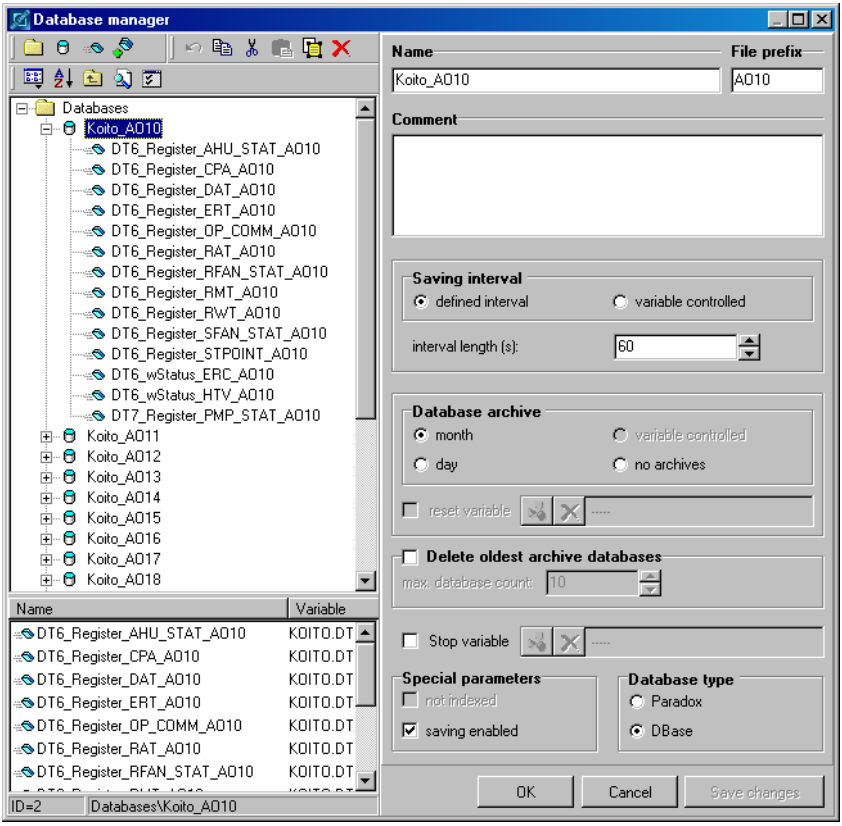
Comment — This parameter can be used to specify an optional description of the message.

Explanation — This parameter can be used to specify an optional explanation of the message. The operator can display the explanation by choosing a special command.

2.6.4 Database manager

The ***Database manager*** allows you to define and configure databases. The database is an object representing a physical database stored on a hard disk. By defining a database you enable the runtime module to access (read and/or write) the values of selected variables in time sequence.

Database parameters



Name — This parameter specifies the database's name that must be unique within the project and cannot contain illegal characters.

File prefix — This parameter specifies the prefix used for the database's files.

Comment — This parameter can be used to specify an optional description of the database.

Saving interval — This parameter determines when to save a new record to the database. Saving can be done periodically (*defined interval*) at the specified time in-

terval (*interval length (s)*) or on the leading edge (the off-to-on transition) of the specified binary-type variable (*variable controlled*). The *Reset variable* option determines whether the control variable should be set to 0 after detecting the leading edge of the variable by the runtime module. If this option is active, it is important that each database use a different control variable.

Database archive

This parameter determines whether and how to create archive files for the database. Archive files are created by copying the database's current files (the files to which the runtime module saves new records) to the database's archive directory and renaming them. If the operation is successful, the current files are deleted. Archiving can be performed periodically (*month* or *day*) or not at all (*no archives*).

Delete oldest archive databases

This parameter determines whether to delete the oldest archive databases so that the number of archive databases does not exceed the specified value. The oldest databases are deleted when creating a new archive database.

Stop variable

This parameter determines whether to use the specified binary-type variable to control saving records to the database. If the value of the variable is equal to zero, saving is enabled. Otherwise, saving is disabled.

Special parameters

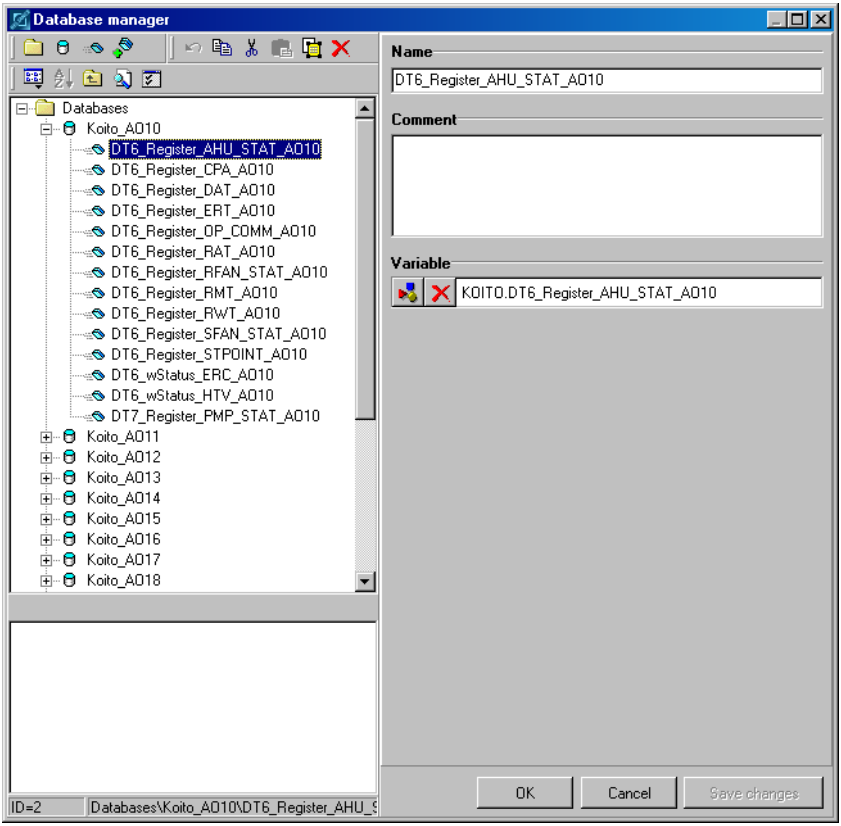
not indexed

This option determines whether to create and maintain the primary index file for the database. It only applies to Paradox-type databases. If this option is not active, the database 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 database. By default, this option is not active.

saving enabled This option determines whether saving new records to the database is enabled. By default, this option is active.

Database type This parameter determines the data format used for the database (*Paradox*, *DBase*).

Database field parameters



Name This parameter specifies the field's name that must be unique within the database and cannot contain illegal characters.

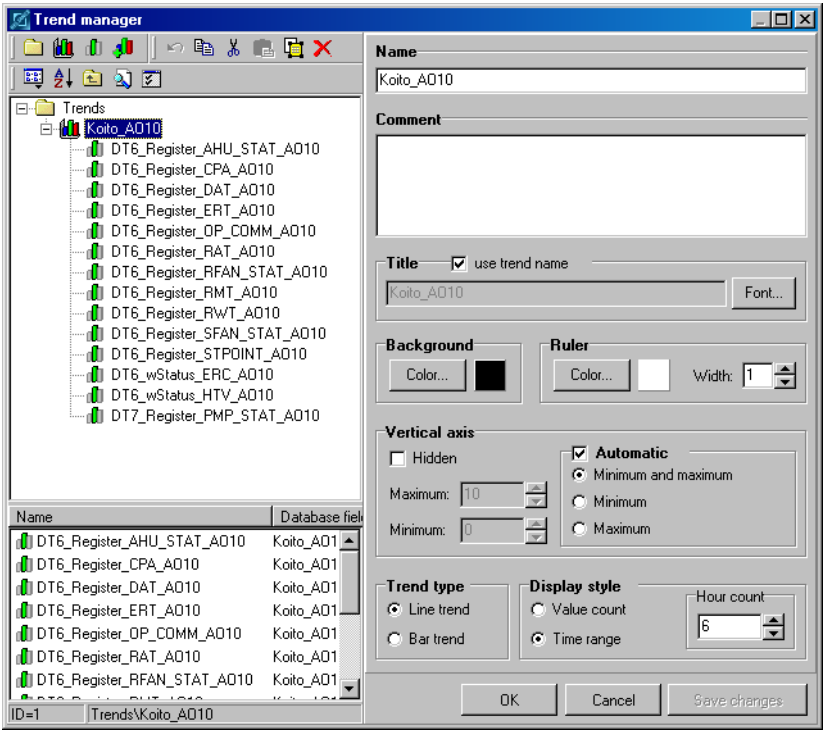
Comment This parameter can be used to specify an optional description of the field.

Variable This parameter specifies the link to the variable whose value is to be saved to the database.

2.6.5 Trend manager

The ***Trend manager*** allows you to define and configure trends. The trend is an object used for graphic presentation of the data stored in a database(s). Trends can be displayed in runtime mode via the trend viewer.

Trend parameters



Name This parameter specifies the trend's name that must be unique within the project and cannot contain illegal characters.

Comment This parameter can be used to specify an optional description of the trend.

Title This parameter specifies the text to be displayed as the title of the trend using the specified font. If the *use trend name* option is active, the trend's name is used as the title.

Background This parameter specifies the background color of the trend. The color should contrast with colors used for individual series.

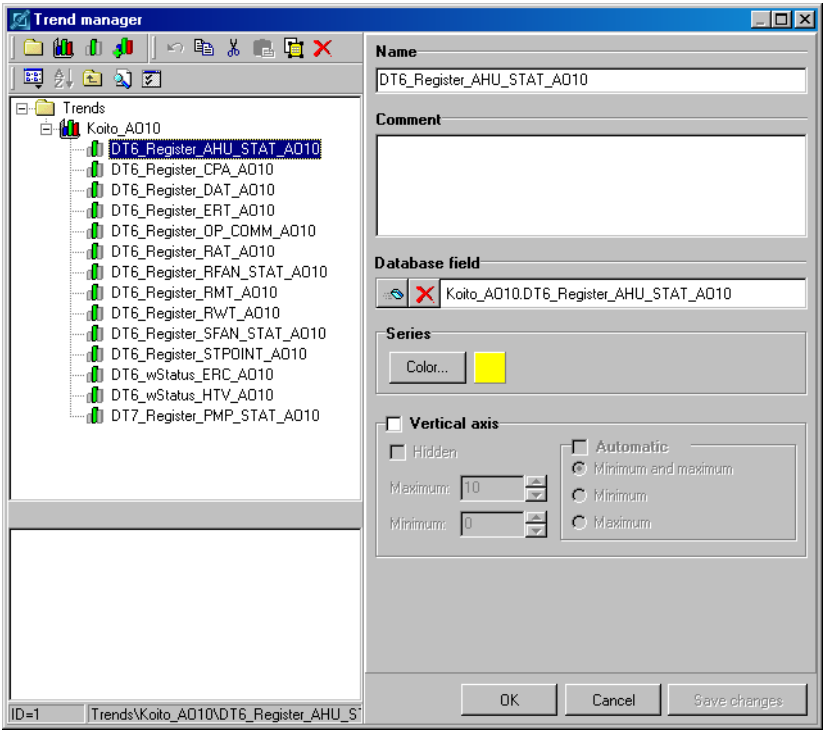
Ruler These parameters determine the color and width of the ruler. The ruler is a vertical straight line drawn on the background. It is designed for accurate reading of values of individual series at the crossing point of the ruler and the series. The color of the ruler should contrast with the background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

Vertical axis These parameters enable you to configure the behavior of the vertical axis common to all series. The *Hidden* option can be used to hide the axis in cases when each series uses its own vertical axis. The *Automatic* option 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* parameters.

Trend type This parameter determines how to graphically represent the series' values.

Display style These parameters determine the time range displayed on a single page of the trend. If *Value count* is active, the trend viewer always attempts to display the specified number of values (time samples) on a single page. In this case, it is required that all the series must be linked to the same database so that each series displays values 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 values (time samples).

Trend series parameters



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

Comment — This parameter can be used to specify an optional description of the series.

Database field — This parameter specifies the link to the database field whose values (time samples) are to be displayed by the series.

Series

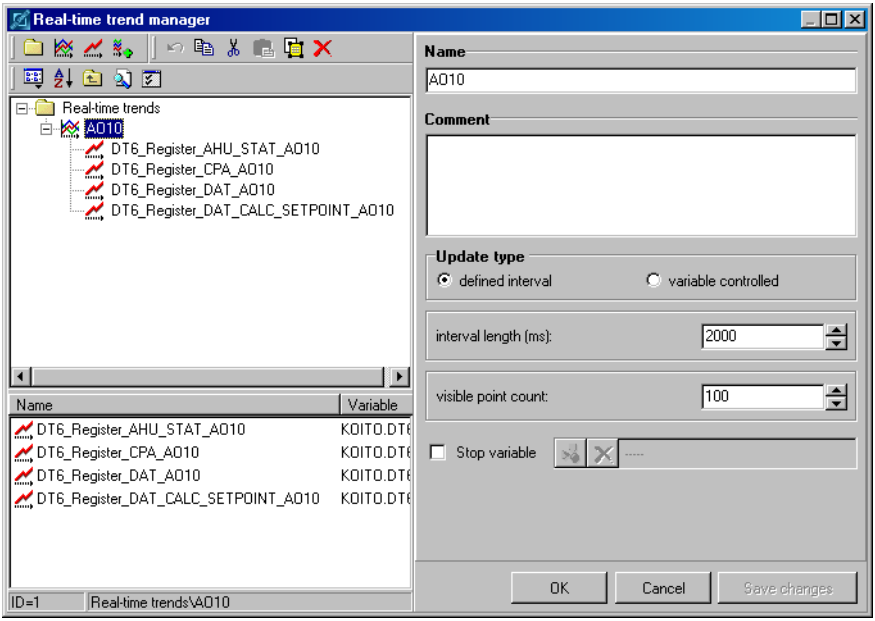
Color — This parameter determines the series' color. It should contrast with the background color of the trend.

Vertical axis This option determines whether the series has its own vertical axis or uses the vertical axis common to all series. The other parameters enable you to configure the behavior of the vertical axis private to the series. The *Hidden* option can be used to hide the axis. The *Automatic* option 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* parameters.

2.6.6 Real-time trend manager

The ***Real-time trend manager*** allows you to define and configure real-time trends. The real-time trend is an object used for graphic presentation of a sequence of the most recent values (time samples) of the selected variables. The values (time samples) are only stored in the memory, not in a database. The real-time trend can be displayed in runtime mode by the *Real-time trend* component placed into a project window.

Real-time trend parameters



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

Comment — This parameter can be used to specify an optional description of the trend.

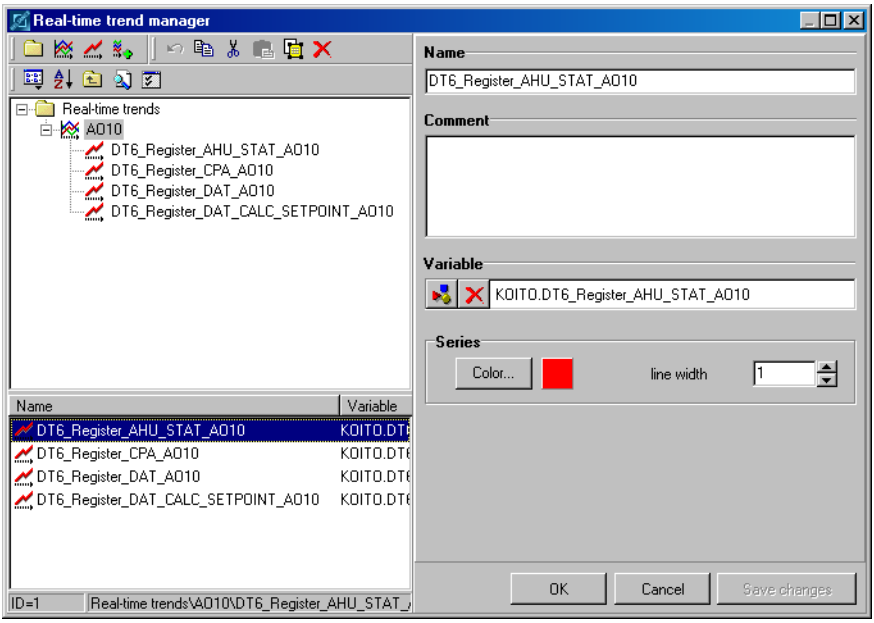
Update type — This parameter determines when to update the trend (i.e. when to add the next time sample to the trend). Updating can be done periodically (*defined interval*) at the specified time interval (*interval length (ms)*) or on the leading edge (the off-to-on transition) of the specified binary-type variable (*variable controlled*).

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

Stop variable — This parameter determines whether to use the specified binary-type variable to control updating the

trend. If the value of the variable is equal to zero, updating is enabled. Otherwise, updating is disabled.

Real-time trend series parameters



Name This parameter specifies the series' name that must be unique within the trend and cannot contain illegal characters.

Comment This parameter can be used to specify an optional description of the series.

Variable This parameter specifies the link to the variable whose values (time samples) are to be displayed by the series.

Series

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

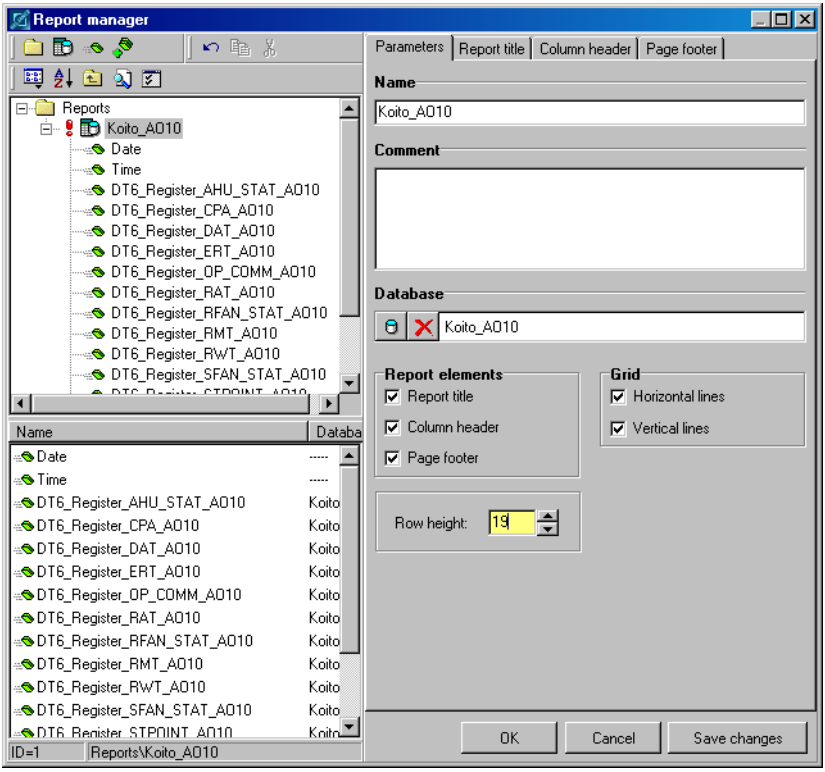
Line width This parameter determines the width (in pixels) of the line used for drawing the series.

2.6.7 Report manager

The ***Report manager*** allows you to define and configure reports. The report is an object used for graphic presentation of the data stored in a database(s) in a tabular format. Reports can be displayed, printed and exported in runtime mode via the report viewer.

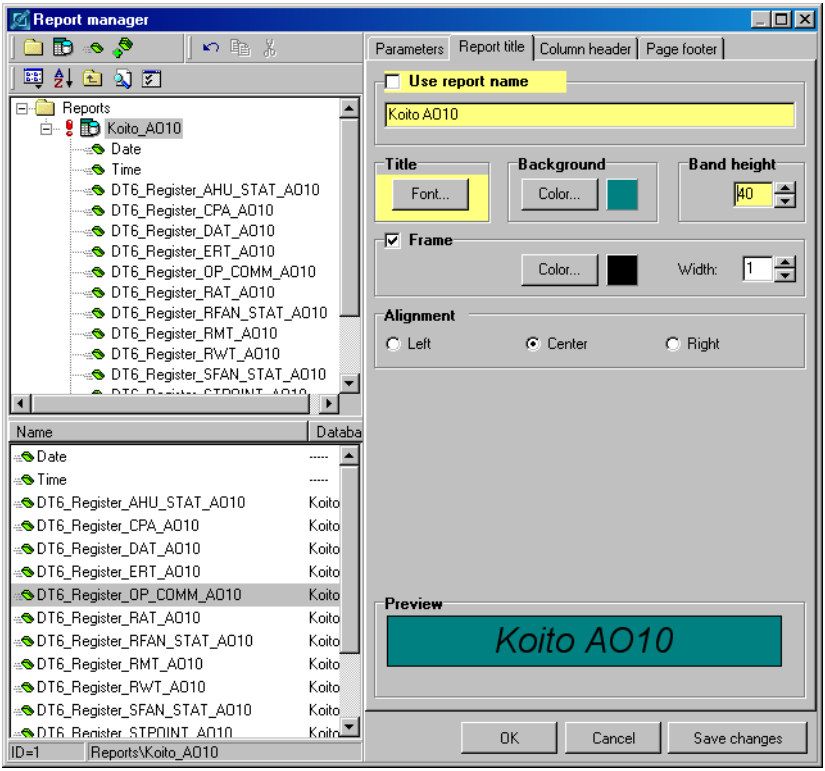
Report parameters

Parameters



Name	This parameter specifies the report's name that must be unique within the project and cannot contain illegal characters.
Comment	This parameter can be used to specify an optional description of the report.
Database	This parameter specifies the link to the database whose contents are to be displayed by the report.
Report elements	These parameters determine which elements to include in the report.
Grid	These parameters determine whether to display grid lines on the background.
Row height	This parameter 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 name — This option determines whether to use the report's name as the title.

Title — This parameter specifies the text to be displayed in the report header using the specified font.

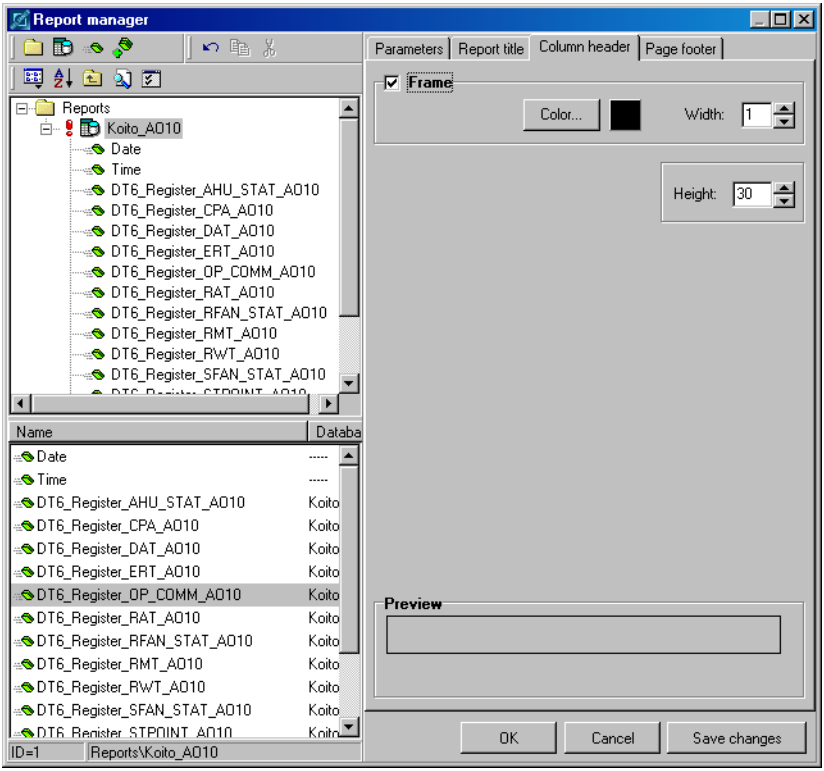
Background — This parameter specifies the background color of the title. The color should contrast with the text color.

Band height — This parameter specifies the height (in pixels) of the report header band.

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

Alignment This parameter determines the alignment of the title within the report header.

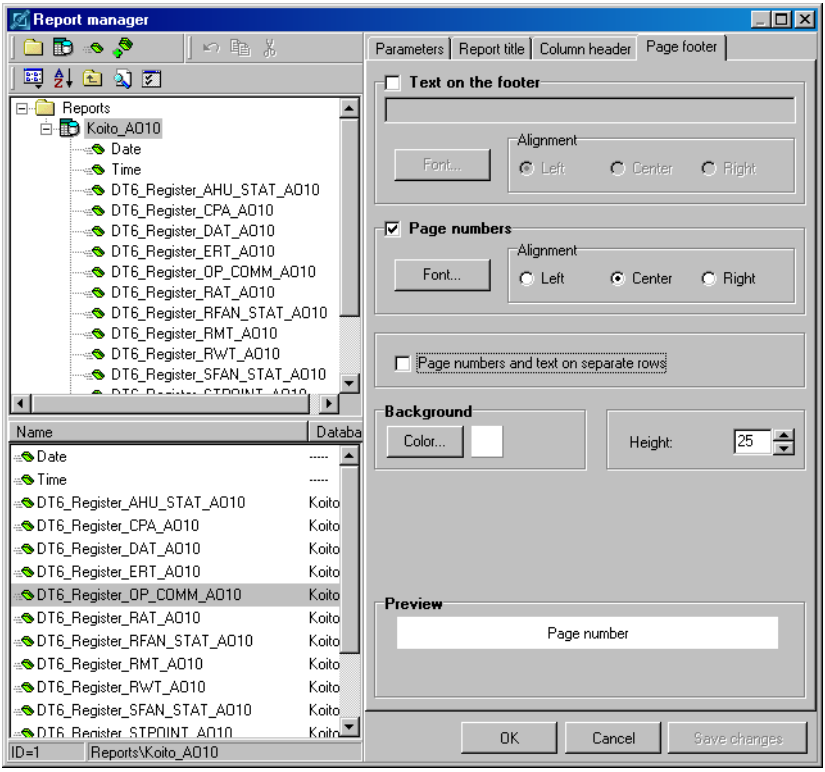
Column header



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

Height This parameter specifies the width (in pixels) of the column header.

Page footer



Text on the footer This option determines whether to display the specified text on the footer of each page of the report using the specified font and alignment.

Page numbers This option determines whether to display the page number on the footer of each page of the report using the specified font and alignment.

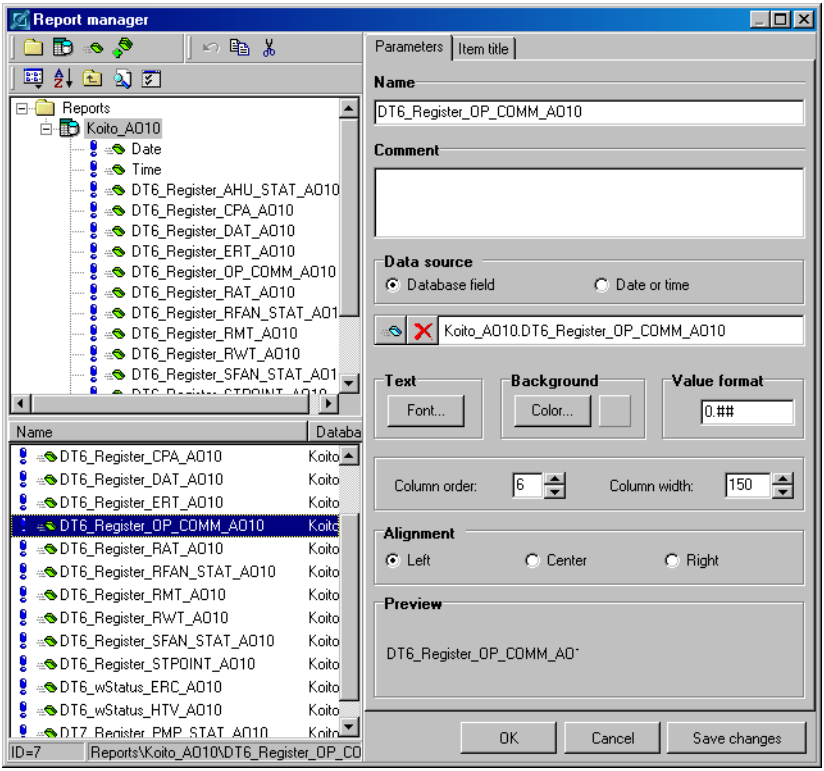
Page numbers and text on separate rows
This option determines whether to print the page number and footer text on separate rows. If this option is not active, the footer is printed on a single line.

Background This parameter specifies the background color of the footer.

Height This parameter specifies the height of the footer.

Report item parameters

Parameters



Name This parameter specifies the item's name that must be unique within the report and cannot contain illegal characters.

Comment This parameter can be used to specify an optional description of the item.

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

Text

Font — This parameter determines the font of the item's text.

Background

Color — This parameter determines the background color of the item's text.

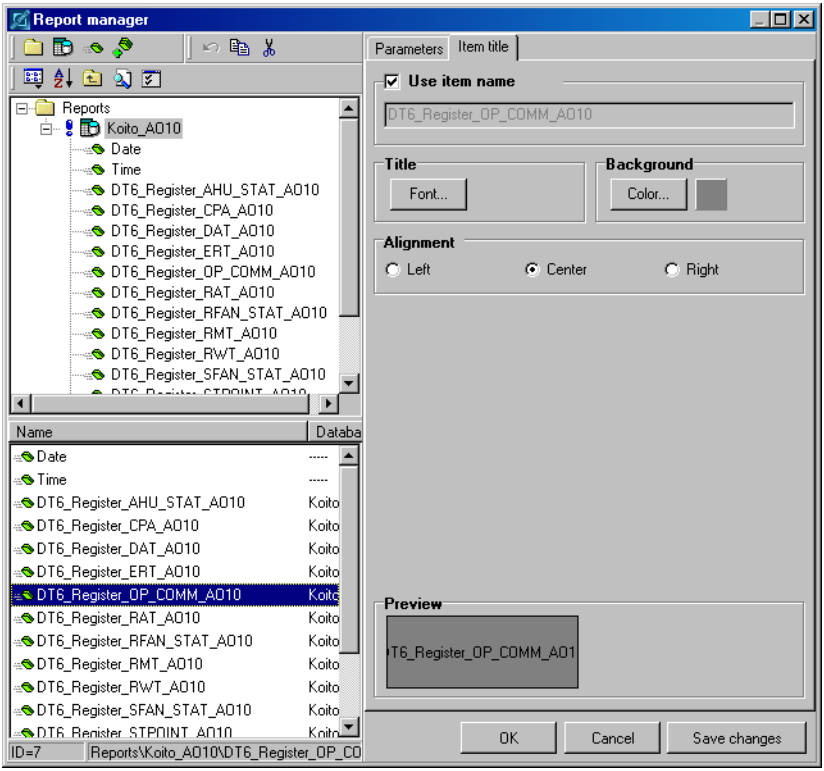
Value format — This parameter specifies the format string to be used when converting the item's value to text for display purposes. It is only used for report items linked to numeric database fields.

Column order — This parameter determines the order of the item's column within the report. By default, this parameter is equal to zero and column order within the report corresponds to the order, in which the items were added to the report (i.e. according to increasing ID). Column order can be changed by specifying this parameter.

Column width — This parameter specifies the width (in pixels) of the item's column.

Alignment — This parameter specifies the alignment of the item's text within the item's column.

Item title



Use item name This option determines whether to use the item's name as the column title.

Title

Font This parameter determines the font of the item's column title.

Background

Color This parameter determines the background color of the item's column title.

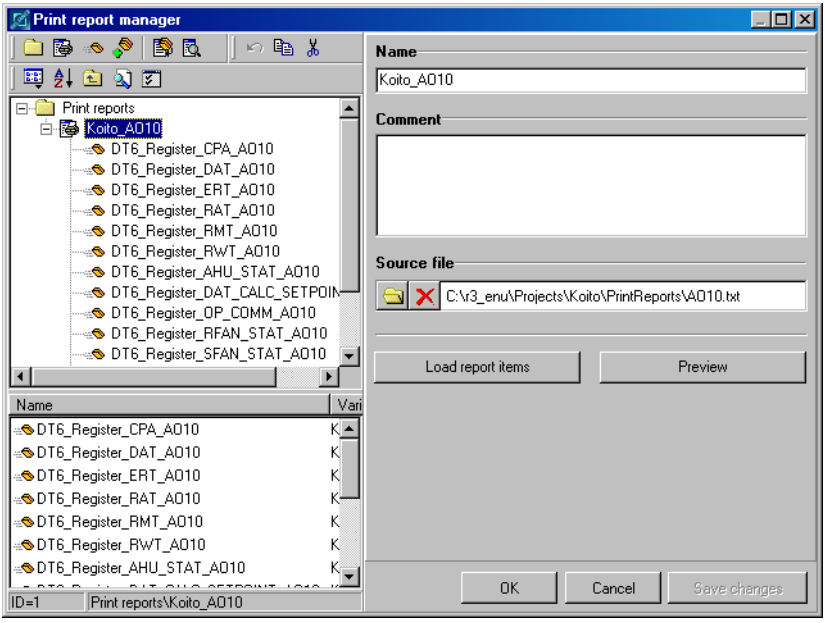
Alignment This parameter specifies the alignment of the item's column title within the column.

2.6.8 Print report manager

The ***Print report manager*** allows you to define and configure print reports. The print report is an object used for graphic presentation of current values of the selected variables in a user-defined format. Print reports can be displayed, printed and exported in runtime mode via the print report viewer.

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

Print report parameters



Name — This parameter specifies the report's name that must be unique within the project and cannot contain illegal characters.

Comment — This parameter can be used to specify an optional description of the report.

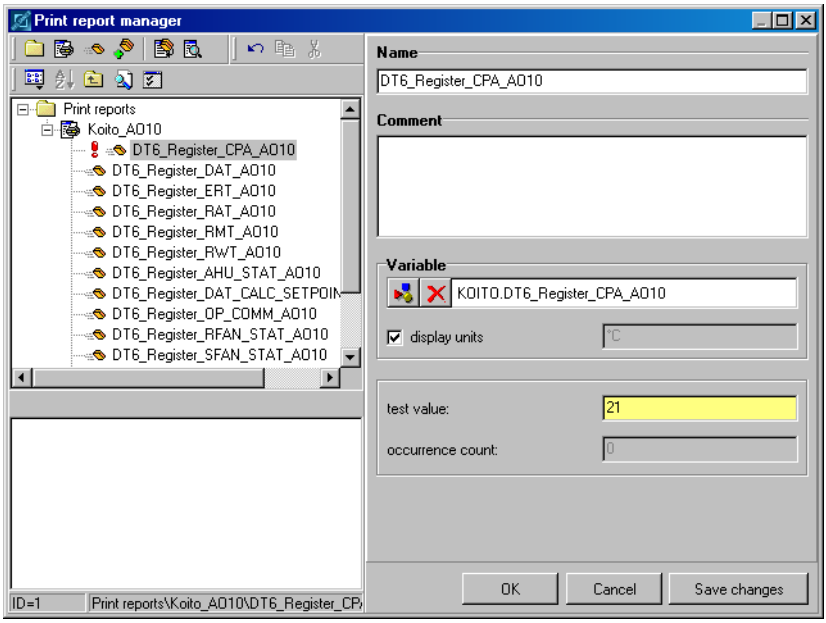
Source file This parameter specifies the file that contains the template for the report.

Load report items This command is used to load the contents of the source file to update the list of report items.

Preview This command is used to preview the report.

Print report item parameters

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



Name This parameter specifies the item's name that must be unique within the print report and cannot contain illegal characters.

Comment This parameter can be used to specify an optional description of the item.

Variable This parameter specifies the link to the variable whose value is to be displayed on the place(s) marked in the template.

display units This option determines whether to display units of measurement after the value of the variable.

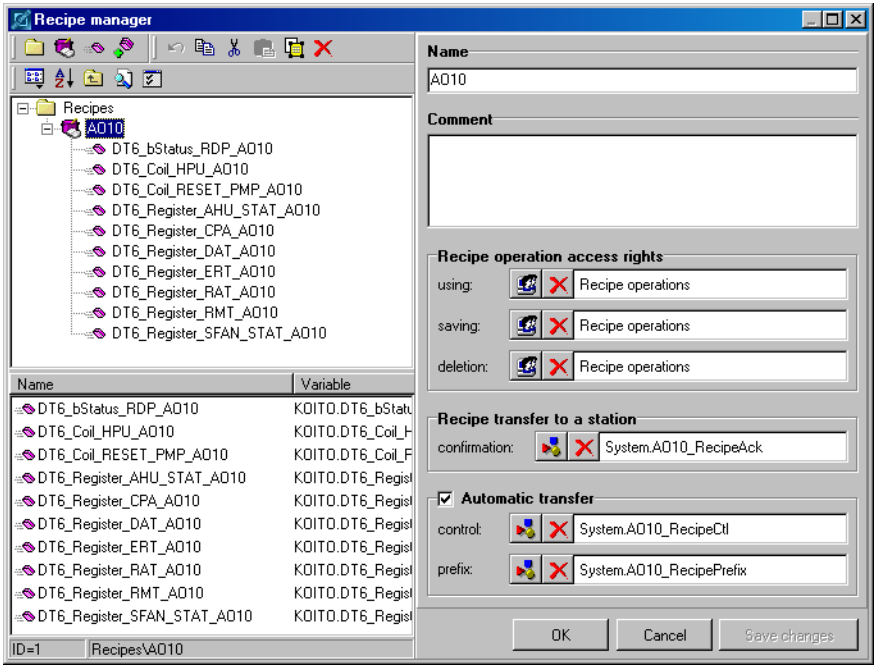
test value This parameter specifies the value to be used instead of the current value of the variable when previewing the report.

occurrence count This parameter 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. The recipe is an object representing a group of variables. It is used to store current values of the variables to a disk file. The stored recipe can later be loaded from the file and transferred to the respective stations.

Recipe parameters



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

Comment This parameter can be used to specify an optional description of the recipe.

Recipe operation access rights

using This parameter specifies the access rights required for using the recipe (i.e. loading the stored recipe from a file into the respective stations).

saving This parameter specifies the access rights required for storing the recipe to a file (i.e. creating a stored recipe).

deletion This parameter specifies the access rights required for deleting the recipe (i.e. deleting the file containing the stored recipe).

Recipe transfer to a station

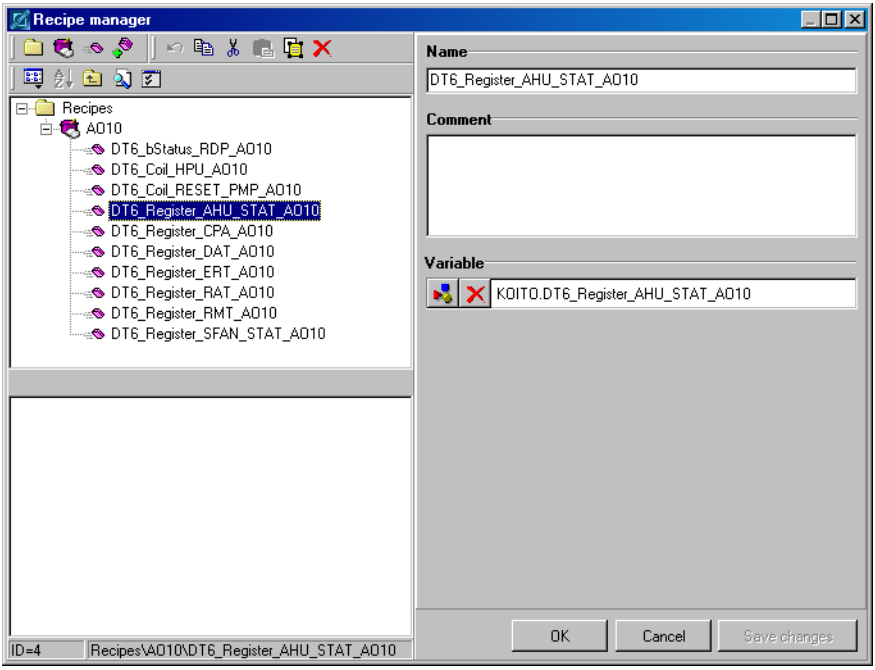
confirmation This parameter specifies the link to the binary-type variable that should receive confirmation (by setting the value to 1 by the runtime module) of loading the stored recipe from a file and transferring it to the respective stations. This parameter is optional.

Automatic transfer

control This parameter specifies the link to the binary-type variable to be used to control the automatic transfer of a stored recipe to the respective stations (the transfer does not require the operator's command). The transfer is performed on the leading edge of the variable (the off-to-on transition).

prefix This parameter specifies the link to the string-type variable to be used to control the prefix of the name of the file containing the stored recipe to be transferred to the respective stations.

Recipe item parameters



Name — This parameter specifies the item's name that must be unique within the recipe and cannot contain illegal characters.

Comment — This parameter can be used to specify an optional description of the item.

Variable — This parameter specifies the link to the variable whose value is to be stored as part of a stored recipe.

2.6.10 Picture manager

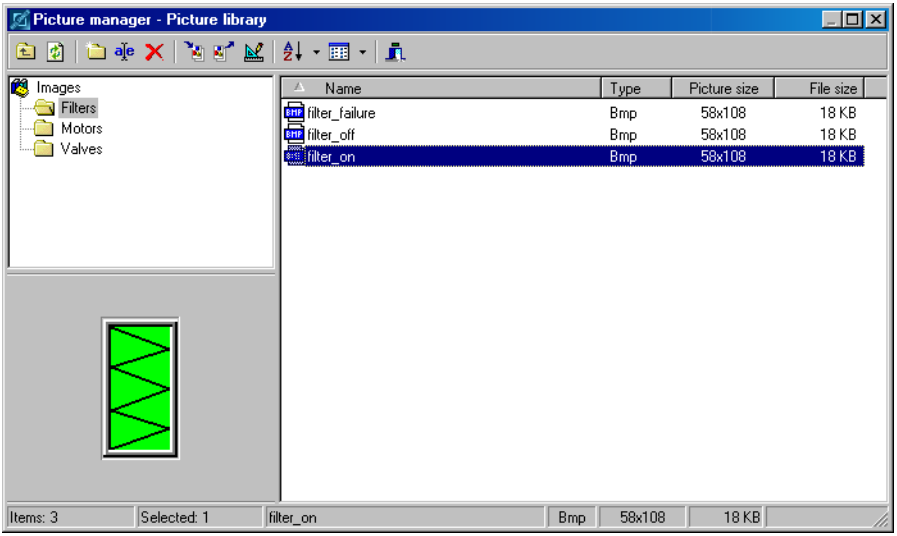
Graphic elements are usually used in visualizing projects for better illustration and more precise appearance. They can be divided into raster and vector graphics formats, i.e. bitmaps and vectors. *Reliance* works with raster files in the **.bmp*, **.jpg* and **.gif* format as well as **.wmf* and **.emf* 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. *Reliance* lets you manage pictures via the ***Picture manager***. The ***Picture manager*** can operate either on graphic files stored in a directory structure (the so-called Picture library, which is not part of a visualizing project and is common to all projects) or on a picture database (which is part of a visualizing project).

Using the Picture manager for managing the Picture library

To use the ***Picture manager*** for managing the ***Picture library***, choose the ▸ ***Tools*** ▸ ***Picture library...*** command. The ***Picture library*** is located in the directory defined via the ***Environment options*** dialog (see the chapter 2.3.3.1 PATHS). When operating on the ***Picture library***, the ***Picture manager*** enables you to create a directory structure (displayed as a folder structure in the top left pane of the dialog), import graphic files to the library from various locations using the appropriate target folder, view the list of graphic files (displayed as a list of pictures in the right pane of the dialog) located in the selected folder, and view and/or edit the selected picture. It is useful to prepare a library of frequently used pictures. You can use the pictures located in the »Pic« directory on the installation CD.



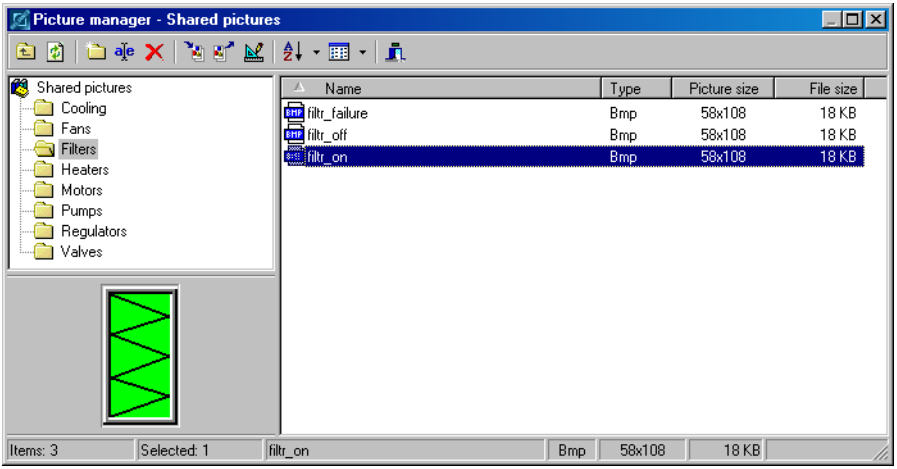
Description of toolbar commands

Up	This command is used to move the selection one level up in the folder (directory) structure.
Update	This command is used to: 1) Update the display of the pictures after modifying them via an external application run by choosing the *Edit* command. 2) Update the list of pictures after adding or deleting picture file(s) to/from the current folder (directory).
Add	This command is used to create a new folder (directory).
Rename	This command is used to rename the selected picture (picture file) or folder (directory).
Remove	This command is used to delete the selected pictures (picture files) and/or folders (directories).
Import	This command is used to copy graphic files (**.bmp*, **.gif*, **.jpg*, **.wmf*, **.emf*) from the source location to the current folder (directory).
Export	This command is used to export pictures to graphic files (**.bmp*, **.gif*, **.jpg*, **.wmf*, **.emf*).
Edit	This command is used to run an external application (defined via the ***Environment options*** dialog) in order to edit the picture. After editing the picture and closing the application, you have to choose the *Update* command in order for the changes to be reflected immediately in the preview.
Sort	This command is used to sort the pictures by various categories.
Display	This command is used to change the display style of the pictures (*List*, *Details*, *Miniatures*).
Exit	This command is used to close the ***Picture manager***.

Using the Picture manager for managing a picture database

If you want to use a picture in a visualizing project, it must be first imported to a picture database of the project via the ***Picture manager***. If the picture is to be only used in a single project window, it should be imported to the picture database of the window. Otherwise, it should be imported to the shared picture database. To use the ***Picture manager*** for managing the picture database of the window, choose the ***Window picture manager...*** command from the window's local menu. To use the ***Picture manager*** for managing the shared picture database, choose the ***Managers*** ***Shared picture manager*** command.

When operating on a picture database, the ***Picture manager*** enables you to create a single-level folder structure (displayed in the top left pane of the dialog), 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 dialog) included in the selected folder and view and/or edit the selected picture.



Description of toolbar commands

Up — This command is used to move the selection one level up in the folder structure.

Update	This command is used to update the pictures after modifying them via an external application run by choosing the *Edit* command.
Add	This command is used to create a new folder.
Rename	This command is used to rename the selected picture or folder.
Remove	This command is used to delete the selected pictures and/or folders.
Import	This command is used to import pictures from graphic files (**.bmp*, **.gif*, **.jpg*, **.wmf*, **.emf*) to the picture database.
Export	This command is used to export pictures from the picture database to graphic files (**.bmp*, **.gif*, **.jpg*, **.wmf*, **.emf*).
Edit	This command is used to run an external application (defined via the *Environment options* dialog) in order to edit the picture. After editing the picture and closing the application, you have to choose the *Update* command in order for the changes to be applied immediately. Otherwise, you will be asked whether you want to apply the changes when closing the ***Picture manager***.
Sort	This command is used to sort the pictures by various categories.
Display	This command is used to change the display style of the pictures (*List*, *Details*, *Miniatures*).
Exit	This command is used to close the ***Picture manager***.

2.6.11 Script manager

In *Reliance*, it is possible to configure most of the behavior and properties of the runtime environment by setting visualizing 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.

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

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

- Time scripts
- Key scripts
- On value change scripts
- Periodic scripts
- Condition scripts
- Event scripts

Common script parameters

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

Name — This parameter specifies the script's name that must be unique within the project and cannot contain illegal characters.

Priority — This parameter specifies the script's priority of processing. The priority increases with the value of this parameter.

Comment — This parameter can be used to specify an optional description of the script.

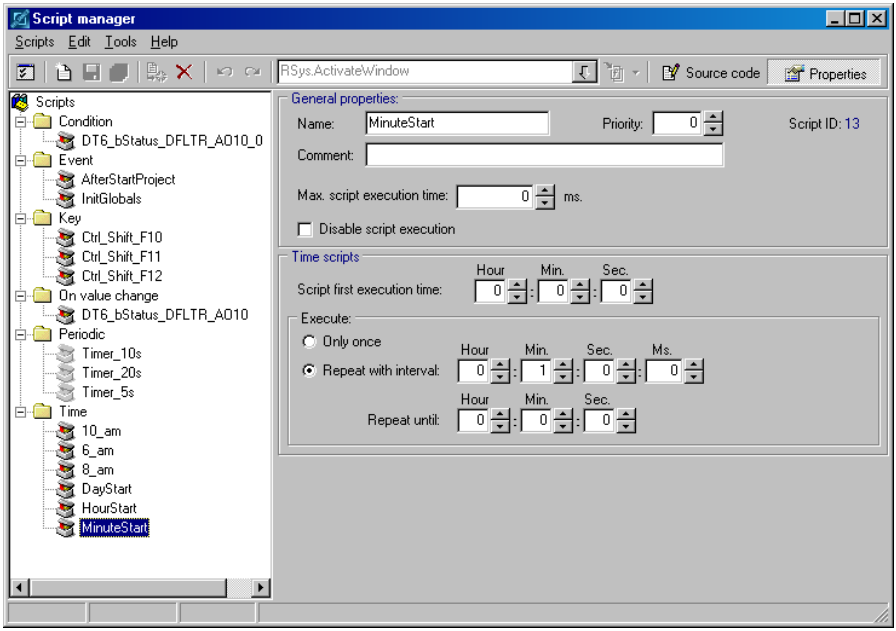
Max. script execution time — This parameter specifies the maximum execution time for the script. If the time is exceeded, the script is forcibly terminated. This feature might be useful when debugging the application to prevent the script from hanging (e.g. due to a continuous loop). However, terminating the script is a very dangerous operation, which may cause the run-time module to hang or crash. Therefore, it is strongly recommended to revert to the default va-

lue of zero in the final version of the application. If the parameter is equal to zero, the execution time is not limited.

Disable script execution This option determines whether the script is disabled by default. If this option is active, the script cannot be executed. However, the script can get enabled or disabled from another script's code (see the Script help).

2.6.11.1 Time scripts

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



Script first execution time

This parameter determines the time of the first execution of the script.

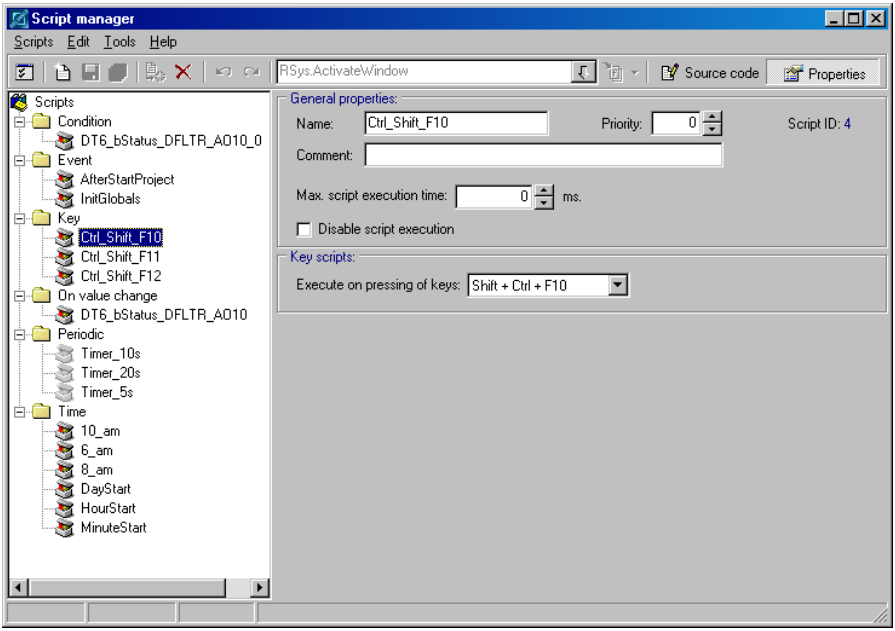
Execute

Only once This option determines whether to execute the script only once at the specified time.

Repeat with interval This option determines whether to execute the script periodically with the specified time interval until the time specified by the *Repeat until* parameter. If this parameter has the 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.

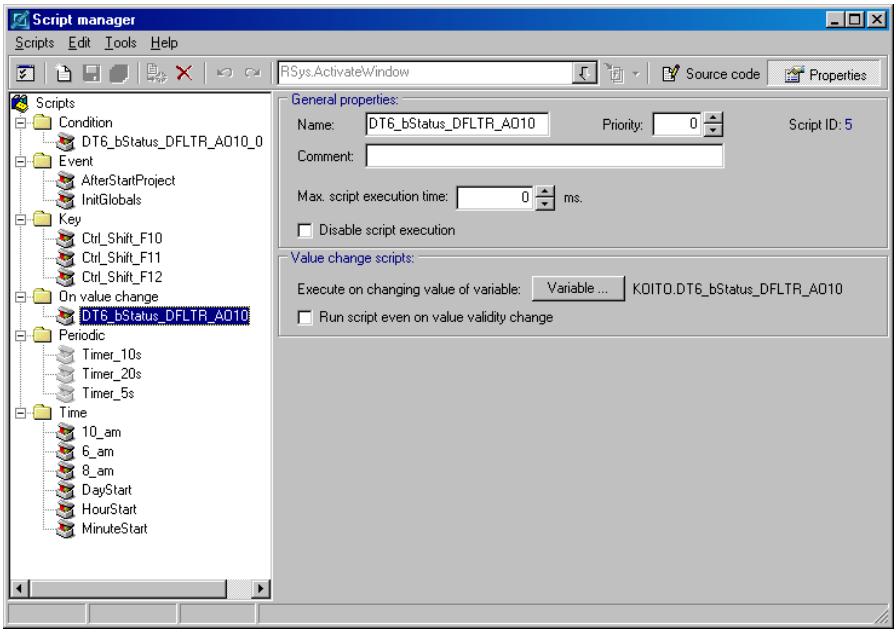


Execute on pressing of keys

This parameter specifies the key shortcut to be pressed in order for the script to be executed.

2.6.11.3 On value change scripts

On value change scripts get executed when the specified variable changes its value or validity.



Execute on changing value of variable

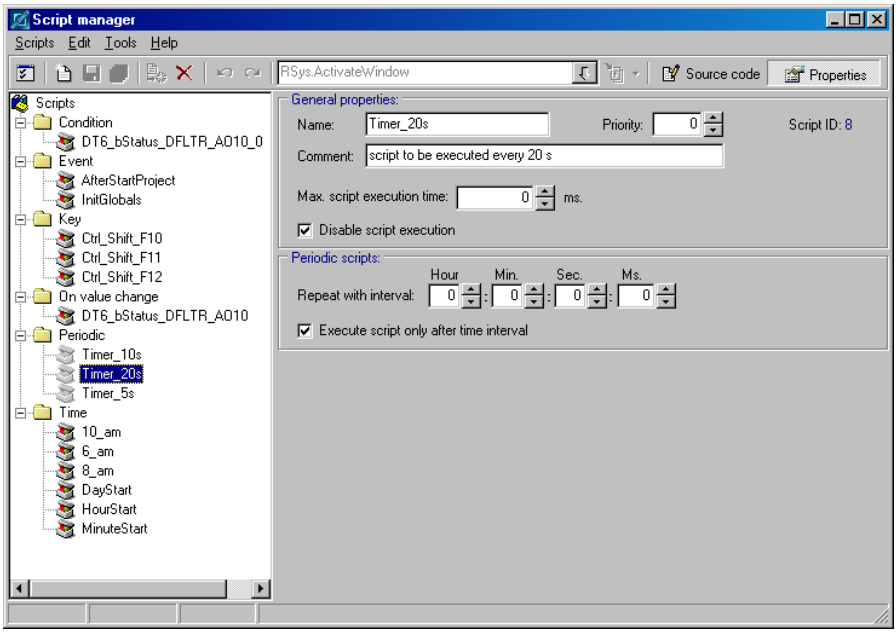
This parameter specifies the link to the variable related to the script.

Run script even on value validity change

This parameter determines whether to execute the script even when the variable becomes valid (although the value remains unchanged).

2.6.11.4 Periodic scripts

Periodic scripts get executed periodically with the specified time interval.

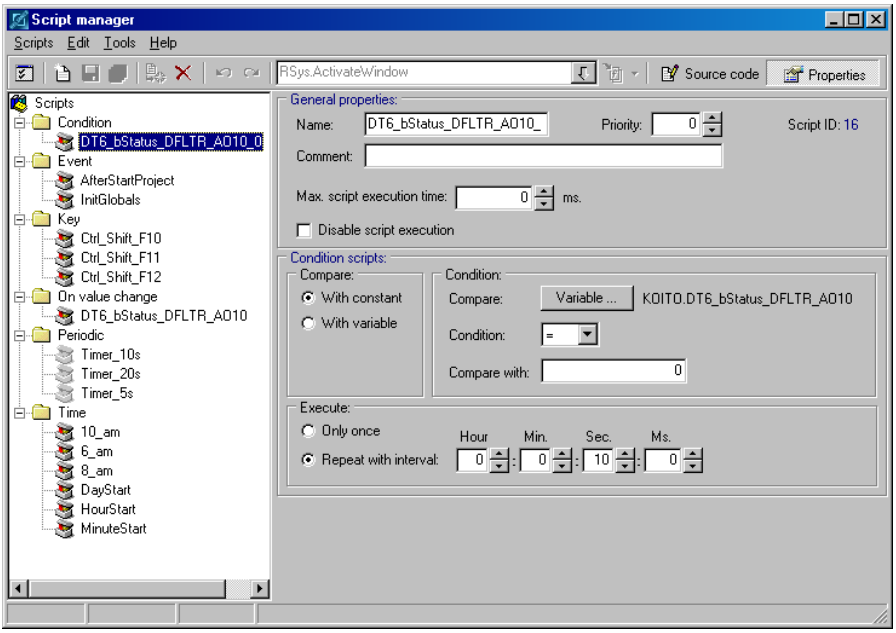


Repeat with interval This parameter specifies the time interval to be used for periodical execution of the script.

Execute script only after time interval
This option determines when to execute the script for the first time after it gets enabled. If this option 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 with the specified time interval.



Compare — This option determines whether to compare the value of the variable (specified by the *Compare* parameter) with the value of the constant or with the value of the other variable (specified by the *Compare with* parameter).

Condition

Compare — This parameter specifies the link to the variable whose value is to be compared with the value of the constant or other variable (specified by the *Compare with* parameter).

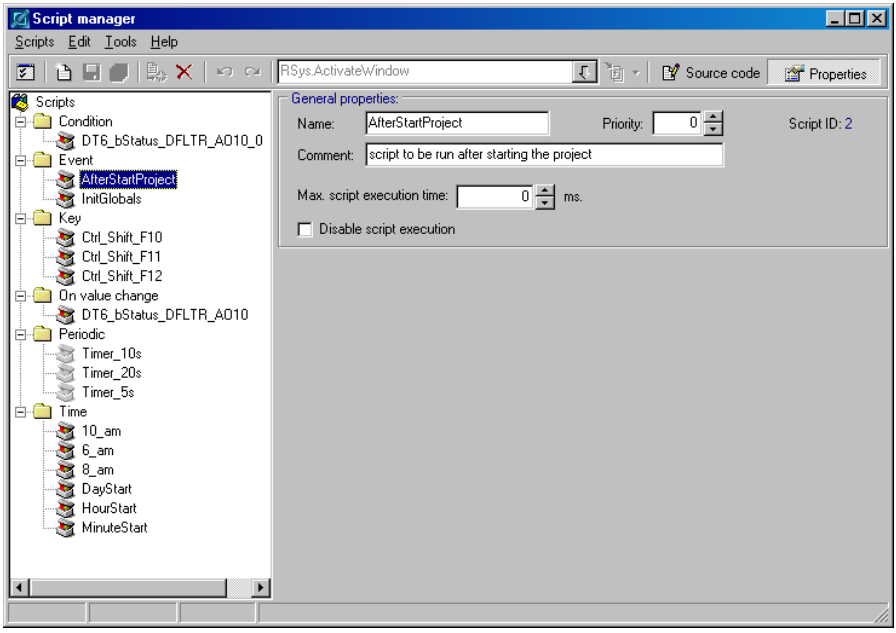
Condition — This parameter determines the logic condition required for the script to get executed.

Compare with This parameter specifies the constant or link to the other variable whose value is to be used for comparison.

Execute This option determines whether to execute the script only once after the specified logic condition gets met, or periodically with the specified time interval while the specified logic condition is still met.

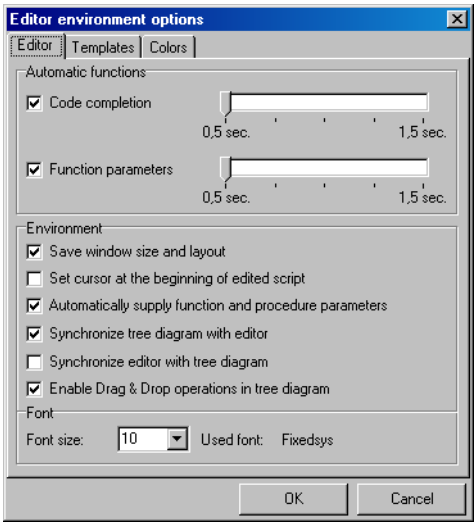
2.6.11.6 Event scripts

Event scripts get executed when a certain event occurs (e.g. when a button is pressed).



2.6.11.7 Setting up the Script manager

Editor



Automatic functions — These parameters enable you to configure features that make code writing easier.

Code completion — This parameter determines whether the code editor should help you complete the code you type in the editor. If this parameter 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 parameter is not active.

Function parameters — This parameter determines whether the code editor should give you help on parameters of functions used in the code you type in the editor. If this parameter 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 parameter is not active.

Environment

Save window size and layout

This option determines whether to save the size, position and layout of the editor window before closing the ***Script manager***. When the ***Script manager*** is opened later, the settings are restored.

Set cursor at the beginning of edited script

This option determines whether to set the cursor at the beginning of the script code after opening the script in the editor.

Automatically supply function and procedure parameters

This option 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.

Synchronize tree diagram with editor

This option determines whether to mark the script in the tree diagram when switching to the script's code in the code editor.

Synchronize editor with tree diagram

This option determines whether to switch to the script's code in the code editor when marking the script in the tree diagram.

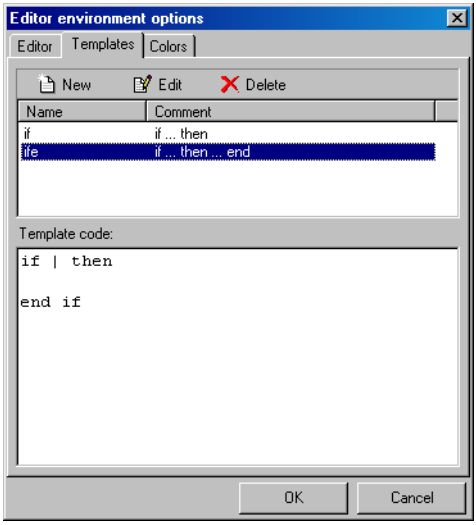
Enable Drag & Drop operations in tree diagram
This option determines whether to enable Drag & Drop operations in the tree diagram. If this parameter is active, you can use the mouse to move (by dragging) or copy (by dragging while holding the *Ctrl* key) the script to another folder.

Font size This parameter specifies the size of the font used in the code editor.

Templates

Templates include commonly used programming statements (such as *if*, *select*, and *for* statements) that you can insert into your script.

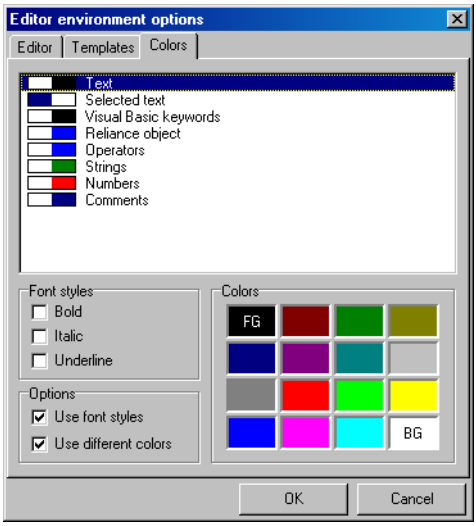
For each template, you must specify a name and code. If you type the name of a template in your script and press *Ctrl+J*, the template's name will be replaced by the template's code.



To define the position of the cursor in the code editor after inserting a template use the pipe character ("|") in the template's code.

Colors

This dialog page enables you to configure the color and font to be used in the code editor to display the script code.



2.6.12 Component manager

The ***Component manager*** is a tool window that enables you to view and edit component properties 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 more than one component is selected in the active window, the page only displays the properties common to all the selected components. All the displayed properties can be edited.

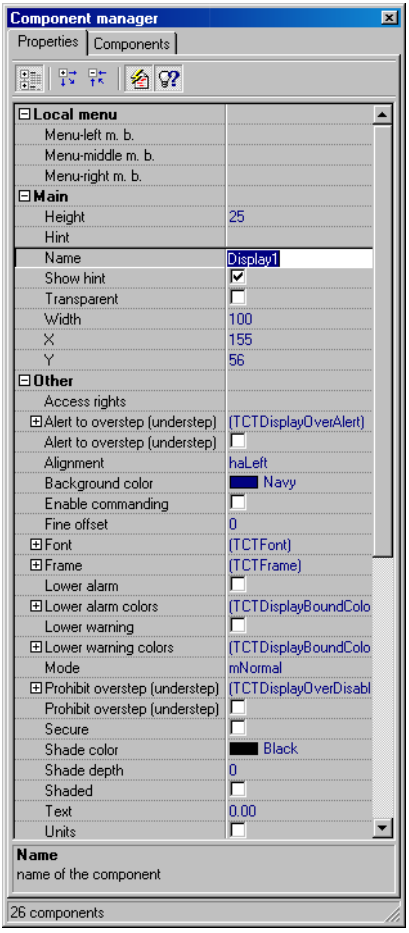
On the ***Components*** page, there is a list of components contained in the active window. The page is divided into two panes.

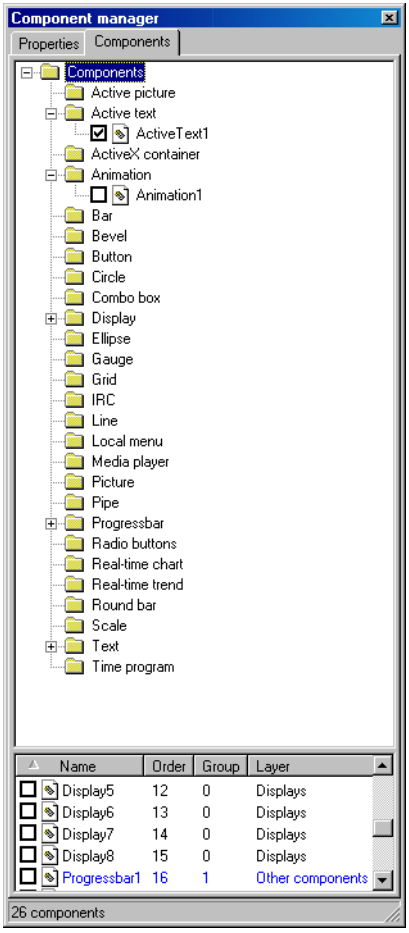
Top pane

The top pane displays the components grouped into folders according to their type.

Bottom pane

The bottom left pane displays the components in a list. The components in the list can be sorted by one of the columns (*Name*, *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.





Display groups This option determines whether to arrange the properties into groups. If this option is not active, the properties are sorted alphabetically.

Expand groups This command is used to expand the groups of properties.

Collapse groups This command is used to collapse the groups of properties.

Display localized property names
This option determines whether to display the localized names (aliases dependent on the language version of the development environment) of properties instead of the actual names.

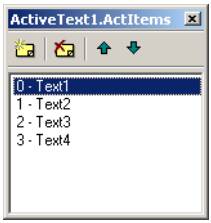
Display brief property descriptions
This option determines whether to display a brief description of the selected property in the bottom section of the ***Properties*** page.

Collection editor

Some components have properties of a special type called a collection. The 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*.

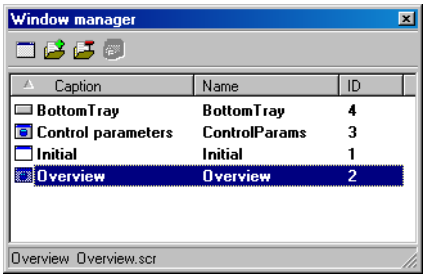
An example of a collection-type property is the *States* property of the *Active text* component. The *States* property represents a list of texts that can be displayed by the component depending on the value of a control variable. To bring up the *Collection editor* for this property, select the component 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***.



2.6.13 Window manager

The ***Window manager*** is a tool window that enables you to manage project windows (i.e. open, activate and close the windows, remove the windows from the project, add new or existing windows to the project, etc.). The ***Window manager*** contains a toolbar, a status bar and a list of project windows. The toolbar contains several buttons representing commands described later in this chapter. The status bar displays the name and file name of the window currently selected in the list. The list of project windows displays information in three columns (*Caption*, *Name* and *ID*) and can be sorted by one of the columns. Each row in the list represents a project window. A project window can be in one of two states: opened (i.e. loaded into memory) and closed (i.e. not loaded into memory). In design mode, 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 or tray) and the way of loading into memory (dynamic or not).



New — This command is used to create a new window and add it to the project.

Add — This command is used to add an existing window to the project. The command brings up an open dialog and prompts you to select a project window file (the file to which the window has previously been stored). The file can be selected from any location but it is not copied to the project's »Win« directory. Therefore, the source location must always be accessible in order for *Reliance* to be able to load the window (for example, you cannot add a project window from a CD or floppy disk).

Remove — This command is used to remove the window from the project. By default, *Reliance* also deletes

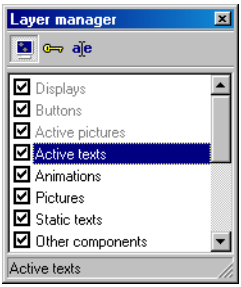
the window's file and picture database after removing the window.

Open — This command is used to open the window (i.e. load the window into memory and activate it).

2.6.14 Layer manager

In design mode, each component in a project window is located on a certain layer. Each project window has 16 layers named *Layer0* to *Layer15* by default. The system of layers gives one more dimension to a project 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 project 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 — This option 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.

Lock — This option 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 — This command 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.

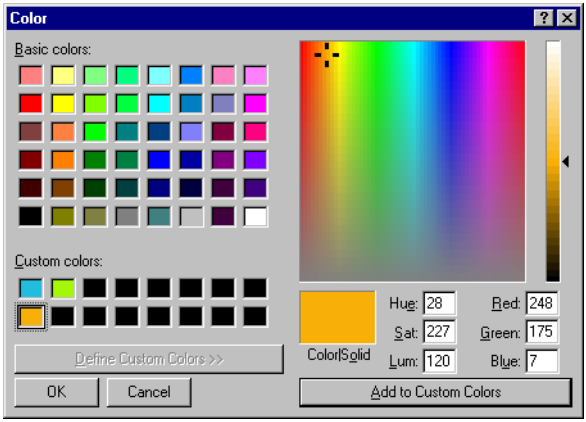
2.7 INFORMATION WINDOW

The ***Information window*** is a tool window located at the bottom of the screen. When opening and closing a visualizing project, the ***Information window*** displays the current status of the operation. If the project is open and the mouse cursor is placed above a project window, the ***Information window*** displays the current cursor position (in pixels), the layer that the component under the cursor is located on or the active layer of the project window (if no component is under the cursor), and basic properties of the component under the cursor (name, link to the main variable and the script).

2.8 STANDARD DIALOG BOXES

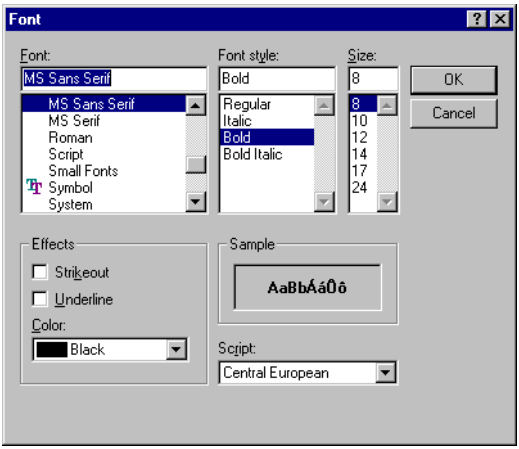
2.8.1 Color selection dialog box

This is the standard Windows 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 visualizing project.



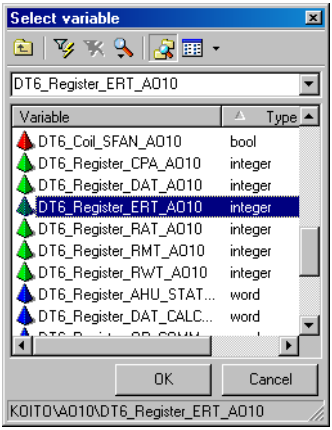
2.8.2 Font selection dialog box

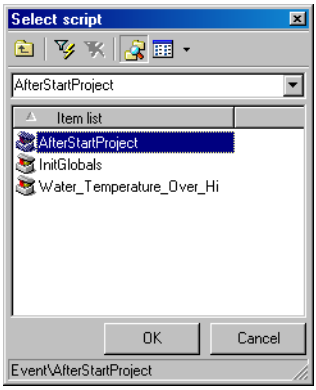
This is the standard Windows dialog box for selecting fonts.



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 — This command is used to display the objects on the immediate superior level.

Set filter — This command is used to apply the filter specified in the edit box to limit 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 — This command is used to cancel the applied filter.

Variables — This command displays a drop-down menu that contains a list of data types. It enables you to filter the list of variables according to data type by deactivating particular types. This command is only available when selecting a variable.

Folders — This option determines whether to display folders in the list. If this option is active and the objects

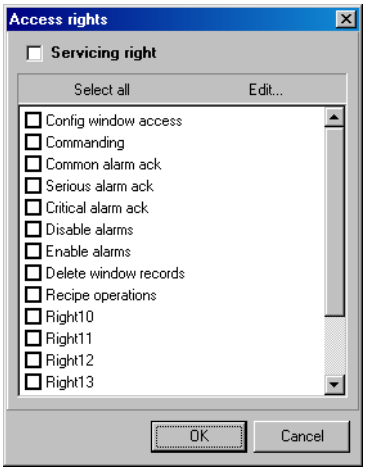
(e.g. variables) are arranged in folders, the list also displays the folders.

Display This command is used to switch between different display types.

Shared pictures This option determines whether the list of objects contains shared pictures. If this option is active, the pictures displayed in the list come from the shared picture database. Otherwise, they come from the picture database of the project window from which the dialog was activated. This option is only available when selecting a picture.

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 acknowledgment).



Servicing right This parameter determines whether to include the *Servicing right* in the specified set of rights.

Select all This command is used to select all the rights in the list.

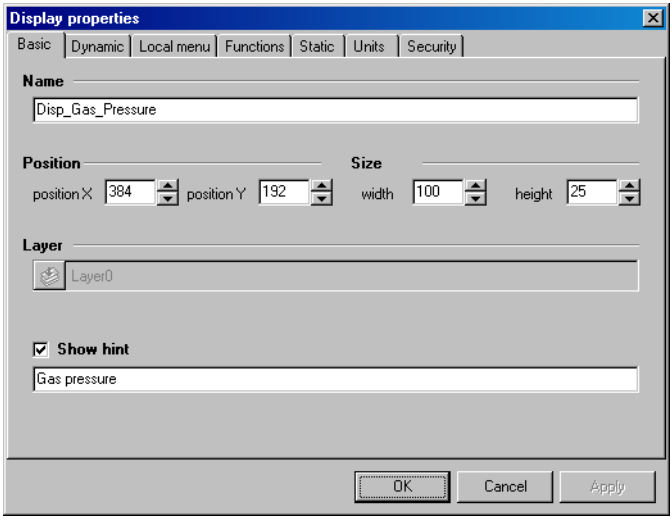
Edit... This command is used to bring up the ***Access right editor*** to rename the rights (see the chapter 2.4.3).

2.9 COMPONENTS

Components are the building blocks of every *Reliance* visualizing project and are used to design project windows. After adding the appropriate components to a project window, you need to configure their properties. To configure the properties of several 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 local 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 This parameter specifies the component's name that must be unique within the window (i.e. the

project window in which the component is placed) and cannot contain illegal characters.

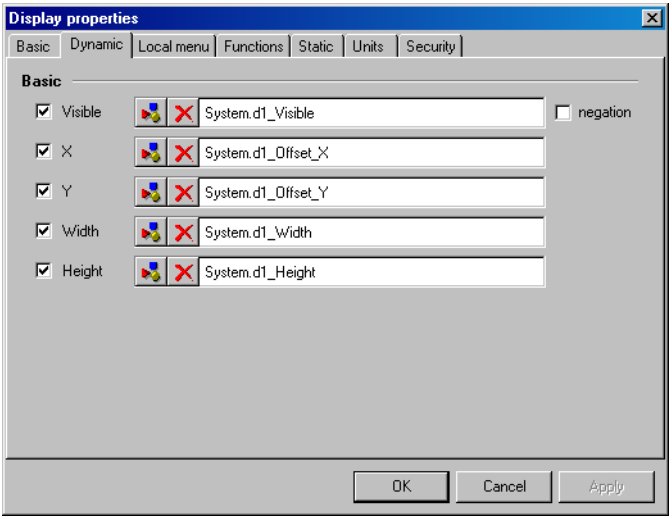
Position These parameters specify the co-ordinates (in pixels) of the upper left corner of the component. The X co-ordinate (*position X*) increases from left to right and the Y co-ordinate (*position Y*) from top to bottom.

Size These parameters specify the width and height (in pixels) of the component.

Layer This parameter specifies the layer on which the component is located.

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

Dynamic



Visible This option determines whether to control the visibility of the component in runtime mode by the specified variable. If this option is active, the component is only visible, when the value of the control variable is not equal to zero.

X This option determines whether to control the horizontal position (the X co-ordinate) of the component in runtime mode by the specified variable. The control variable only determines an increment (decrement) in the horizontal position relative to the design mode position, not the absolute horizontal position.

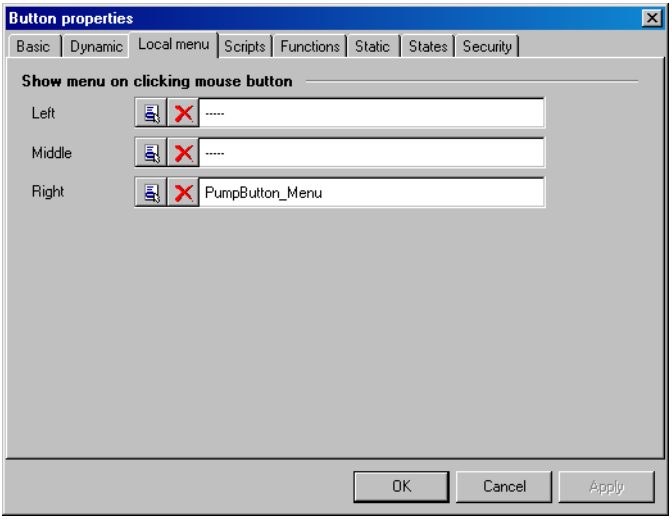
Y This option determines whether to control the vertical position (the Y co-ordinate) of the component in runtime mode by the specified variable. The control variable only determines an increment (decrement) in the vertical position relative to the design mode position, not the absolute vertical position.

Width This option determines whether to control the width of the component in runtime mode by the specified variable. The control variable only determines an increment (decrement) in the width relative to the design mode width, not the absolute width.

Height This option determines whether to control the height of the component in runtime mode by the specified variable. The control variable only determines an increment (decrement) in the height relative to the design mode height, not the absolute height.

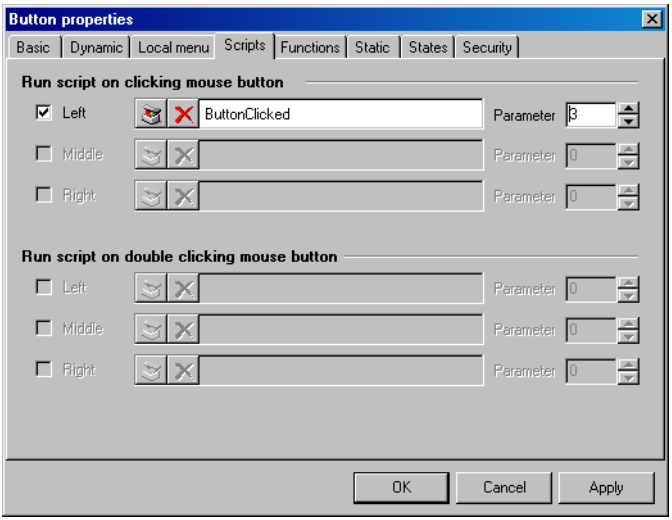
Local menu

These parameters enable you to specify the local menus to be shown when clicking individual mouse buttons on the component in runtime mode.



Scripts

These parameters enable you to specify the scripts 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 several components of the same type to a window

To add several components of the same type to a project 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 terminate 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 project 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 project 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 several components

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

2.10.4 Defining a link to a variable or item

When invoking a selection dialog to define a link to an object, such as a variable 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 visualizing project in runtime mode 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 the project* dialog (see the chapter 2.4.4 CREATING A SHORTCUT TO THE PROJECT). Specify a runtime module (*Reliance runtime, Reliance runtime server* or *Reliance server*) as the value of the *Application* parameter.
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* application.
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 visualizing 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 visualizing 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 the *Animation* component): It is controlled by the 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 the component'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 the *Animation* component). 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 component's *Layout* property which affects whether the picture is drawn in its original size (*glNormal, glByImage, glTile*) or not (*glByObject*). Drawing pictures as stretched or compressed increases graphical load. It is strongly recommended to use pictures in their original size whenever possible (especially with the *Animation* component).
- Relative position of components: If several 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 the *Animation* component) with components whose drawing is time-consuming (e.g. the *Picture* component displaying a large picture).

2.10.8 Optimization of communications to subordinate stations

This chapter contains information that can help you develop applications with efficient communications to subordinate stations (e.g. PLC controllers). 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 stations to the application.

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

When defining communication zones for a station, try to cover as many variables (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 several shorter zones.

3. TECHNOLOGICAL DATA DISTRIBUTION OVER THE INTERNET/INTRANET

3.1 BASIC PRINCIPLES OF COMMUNICATION ON THE INTERNET

Computers in the Internet can be divided into two basic groups – clients and servers. The client is a computer that requests data from a server. The server is a computer that processes the requests and sends a response back to the client. This principle also applies when viewing a Web page via a Web browser running on the client. The requested page is determined by an address specified in the browser. The address contains a unique name – the server address. The server returns the content of the page as a response and the client browser displays it.

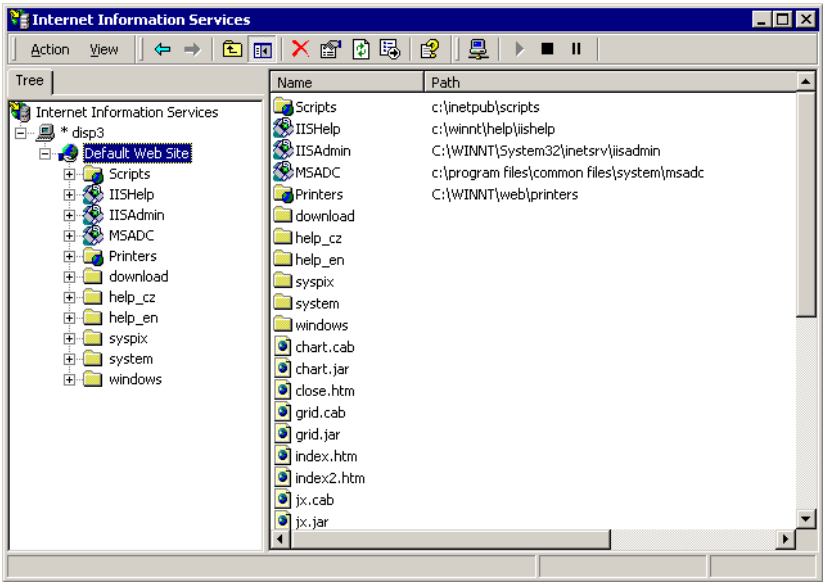
In order to be able to receive and process the client's requests, the server must run a special type of program called a service. If the service provides clients with access to Web pages, the server may be called a Web server.

3.1.1 Services for publishing on the Internet

In Windows, publishing on the Internet is provided by *Internet Information Services* (IIS for short, in Windows 2000 Professional, Server and Windows NT 4 Server) and *PeerWeb Server* (Windows NT 4 Workstation). These programs change the computer into a fully functional Web server.

Upon installing these programs, a home directory for all the HTML pages provided by this server is created on the hard disk of the computer (typically »c:\InetPub\wwwroot«).

To display the IIS console in Windows 2000, choose the ***Start ▸ Settings ▸ Control Panels ▸ Administration Tools ▸ Internet Services Manager*** command.



After selecting the *Default Web Site* item in the left pane of the dialog, start the corresponding service (if it is not already running) by clicking on the *Start service* button on the toolbar. Then it is possible to open any HTML document located in the home directory using a Web browser (for example, to open the document *Test.htm* located in the home directory of the computer *Server1*, specify the address http://Server1/Test.htm in the Web browser). If the computer is not networked currently, it is possible to open the document locally (specify the address http://localhost/Test.htm in the Web browser).

For more information regarding IIS see the Windows Help.

3.1.2 Java applets

The applet is a program written in the *Java* programming language and is designed to run inside a Web browser. It can be included in an HTML page and stored on a Web server. When you view the HTML page using a Web browser running on a client computer, the applet is downloaded along with the page and run inside the browser. Java applets are platform independent, i.e. they can run on any operating system that supports Java applications.

3.2 RELIANCE AND THE INTERNET/INTRANET

Once the *Reliance* application is developed, it can be easily made accessible to remote users over the Internet/intranet through the *Reliance J* Web client. First, it is necessary to export the visualizing project to a special format suitable for the Web client (the so-called WWW format). Open the project in the development environment and choose the ***▸ Project ▸ Export to WWW format...*** command. This will bring up the *WWW format export wizard*, which will guide you through the export process. The project in the WWW format consists of compressed project files, applet program files and an HTML page (*index.htm*) that references the applet.

3.2.1 Reliance J

The *Reliance J* Web client is a Java applet designed for running a visualizing project using a standard Web browser that supports the Java programming language (e.g. Microsoft Internet Explorer 4 and higher, Netscape Communicator 4 and higher). The Web client provides a powerful and easy way to access the technology from remote locations over the Internet/intranet.

3.2.2 Starting Reliance J

The *Reliance J* Web client (i.e. the applet) can be started in two ways.

From a Web page (the preferred way)

To start the applet from a Web page, run a Web browser and specify the address of the Web server and path to the *index.htm* page. For example, if the project is located in the *App1* subdirectory of the home directory for the HTML pages on the Web server *www.appserver.com*, specify the address http://www.appserver.com/App1/index.htm in the Web browser. The rest is done automatically. First, the browser downloads the applet (along with the *index.htm* page) from the Web server and starts it. Next, the applet downloads the visualizing project from the server and opens it. If the applet is configured to use a data server (*Reliance server* or *Reliance runtime server*), it tries to establish a TCP/IP connection to the data server running on the Web server. Otherwise, the applet searches in the project directory on the Web server for the MEM files to obtain current technological data.

Locally

If the project has been exported to a local drive, run the batch file *run.bat* located in the same directory as the project in the WWW format. First, the batch file starts the applet in a Web browser. Next, the applet loads the visualizing project from the local drive and opens it. If the applet is configured to use a data server (*Reliance server* or *Reliance runtime server*), it tries to establish a TCP/IP connection to the data server running on the computer specified through the ***WWW format export wizard*** (see the chapter 3.3 WWW FORMAT EXPORT WIZARD). Otherwise, the applet searches in the project directory on the local drive for the MEM files to obtain current technological data. No Web server is required.

This way of starting the applet can be useful in these situations:

- When developing and debugging the application; this way, you can try data communication between the Web client and a data server even on a single, non-networked computer.
- When the connection to the Internet is very slow; since the project in the WWW format is located on a local drive, the Web client starts very quickly. The connection to the Internet is only used for data communication.

3.2.3 Reliance J and data communication

The *Reliance J* Web client (i.e. the applet) can obtain current technological data in two ways.

TCP/IP (the preferred way)

If the applet is configured to use a data server (*Reliance server* or *Reliance runtime server*), it tries to establish a TCP/IP connection to the data server running on the Web server (if the applet has been downloaded from a Web page) or any specified computer (if the applet has been started locally). Once the connection is established, it can be used by the applet to update current technological data and alarms, download historical trends and alarms, send commands and acknowledge alarms.

Communication on the TPC/IP connection is optimized for size of transferred data. For each Web client, there is a so-called virtual account maintained by the data server. The virtual account contains information on changes in current data and alarms of all the stations provided to the Web client by the data server. This feature enables the Web client to request only the changes.

The number of Web clients that can concurrently be connected to the data server is limited by the license purchased for the data server.

MEM files

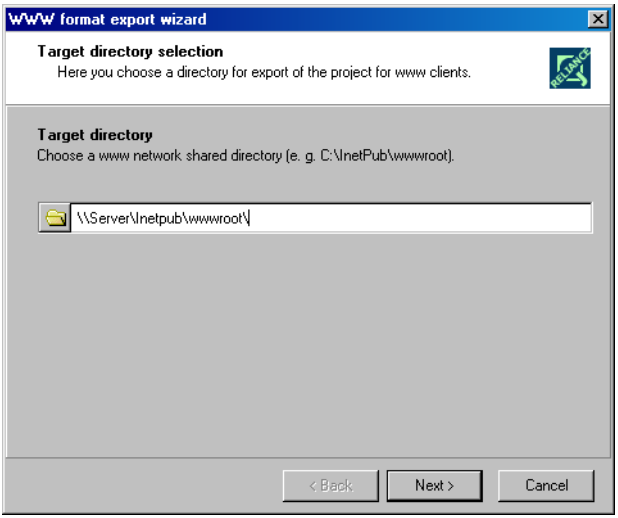
If the applet is configured to use the MEM files, then it can only obtain current data of the stations. It cannot obtain current alarms, download historical trends and alarms, send commands and acknowledge alarms. These limitations originate from the fact that no data server is required to be running on the Web server. The applet obtains current data of the stations by periodically reading the MEM files updated by a runtime module running on the Web server or another computer. The MEM files contain a binary image of the stations' memory and must be located in the same directory on the Web server as the visualizing project in the WWW format. The following two files are always created:

- *sys.mem* (contains the *System* station's current data)
- *web.mem* (contains current data of the other stations)

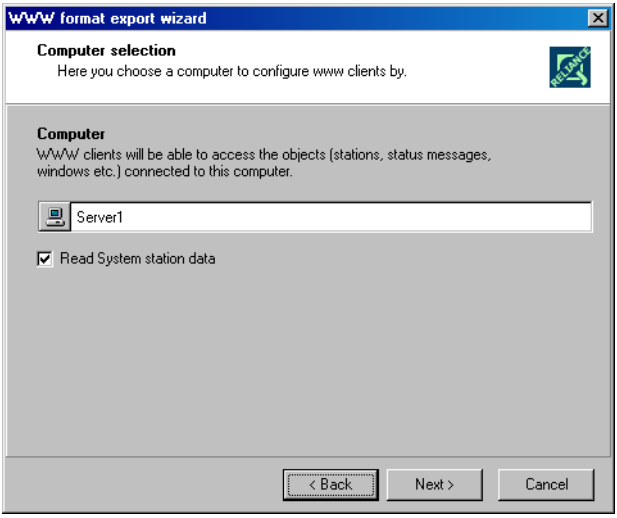
For information on how to configure saving current technological data to the MEM files for the Web clients, see the chapters 2.6.2.2 DEFINING COMPUTERS, 2.6.2.4 CONNECTING STATIONS and 2.6.2.9 DEFINING A STATION PROVIDED THROUGH A NETWORK CONNECTION.

3.3 WWW FORMAT EXPORT WIZARD

This chapter describes the process of exporting the existing visualizing project to the WWW format.

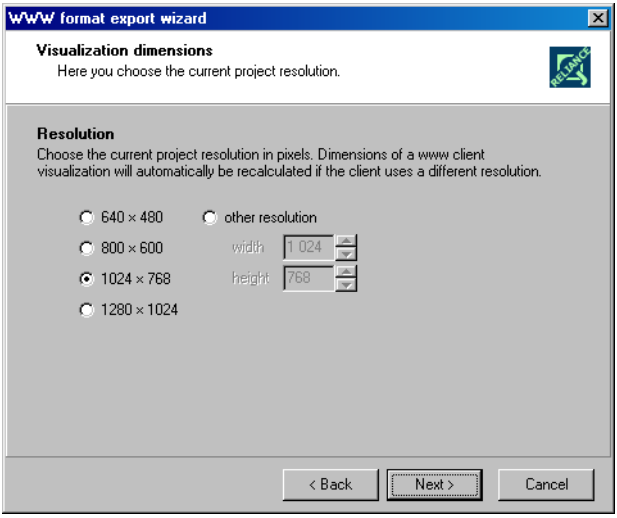


This step enables you to choose the target directory for the project in the WWW format. You can specify a network path as shown in the picture. However, you can also export the project to a local drive and copy it to the Web server later. In order for the Web clients to be able to access the project over the Internet/intranet, the project directory on the Web server must be shared on the WWW network. Preferably, choose the home directory for the HTML pages or its subdirectory.

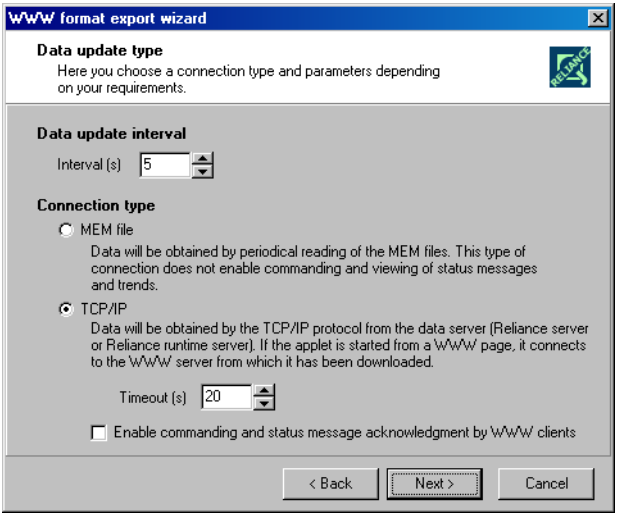


This step enables you to choose the computer (i.e. logical computer defined via the ***Project structure manager***) by which the Web clients should be configured. The Web clients will be able to access only the objects (stations, windows, etc.) available to this computer.

The *Read System station data* parameter determines whether the Web clients should request current data of the *System* station.



This step enables you to choose the point resolution (in pixels) of the project (the resolution, for which the project has been designed). If the Web client is run on a computer with a different resolution, all the graphic elements will be adjusted accordingly.



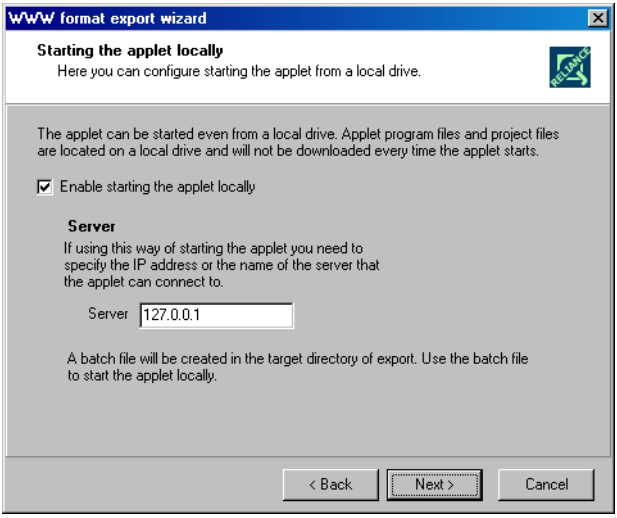
This step enables you to configure data communication for the Web clients.

Data update interval This parameter specifies the time interval to be used by the Web clients for updating current technological data (and alarms, if the *Connection type* parameter is set to TCP/IP).

Connection type This parameter determines the type of connection of the Web clients to a data source (see the chapter 3.2.2 RELIANCE J AND DATA COMMUNICATION).

Timeout This parameter specifies the maximum time period between sending a request to and receiving a response from the data server by the Web client. If the data server does not respond to the request within the time specified, the communication is recognized as faulty and the Web client's connection status changes to offline.

Enable commanding and message acknowledgment by a WWW client
This parameter determines whether to enable the Web clients to send commands and acknowledge status messages (e.g. alarms).



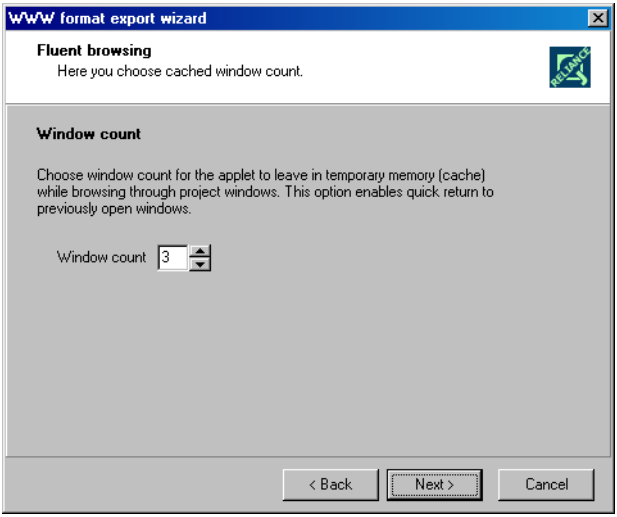
This step enables you to configure starting the applet from a local drive.

Enable starting the applet locally

This option determines whether the applet can be started from a local drive. If this option is active, the project in the WWW format will also include the batch file *run.bat* for starting the applet locally.

Server

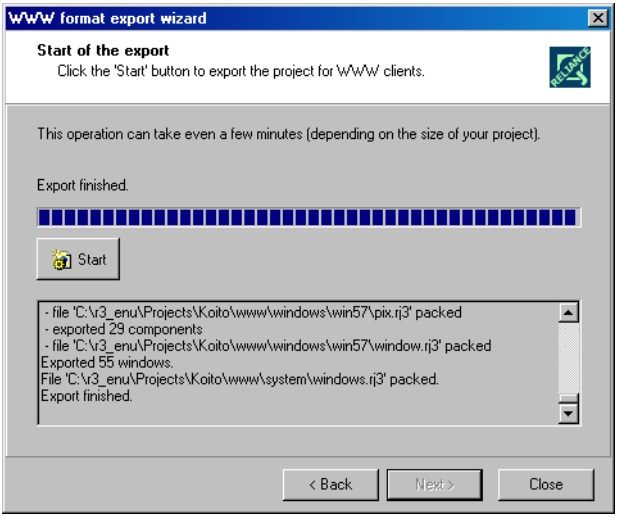
This parameter specifies the IP address or system name of the computer on which the data server (*Reliance server* or *Reliance runtime server*) will be running. You can even specify the local IP address (*127.0.0.1*) in order to be able to run both the applet and the data server locally (i.e. on a single computer).



This step enables you to configure the fluency of browsing through project windows in the Web client.

Window count

This parameter specifies the number of project windows for the applet to store in a temporary memory while browsing through project windows. This feature enables you to quickly return to previously open windows.



This step enables you to generate the project in the WWW format. After the completion of the operation, the following files and subdirectories should be present in the target directory:

Applet files:

- *reli.jar*
- *reliobj.jar*
- *relictrl.jar*
- *reliimg.jar*
- *mmedia.jar*
- *jx.jar*
- *chart.jar*
- *grid.jar*
- *reli.cab*
- *reliobj.cab*
- *relictrl.cab*
- *reliimg.cab*
- *mmedia.cab*
- *jx.cab*
- *chart.cab*
- *grid.cab*
- *index.htm*
- *index2.htm*

- *run.bat*
- *run2.bat*
- *close.htm*
- *loading.gif*

Project files in directories:

»...\windows«
»...\system«
»...\syspix«
»...\dowload«
»...\help_cz«
»...\help_en«

3.4 SW REQUIREMENTS

SW required on a server side

- If the Web client is to be started from a Web page located on the server (the usual and preferred way), a service for publishing on the Internet (e.g. *Internet Information Services*, *PeerWeb Server*, *Apache* or similar) must be running on the server.
- If the Web client is configured to use a data server (*Reliance server* or *Reliance runtime server*), the data server must be running on the server. The WWW version of the project must correspond to the project running in the data server (i.e. the WWW version of the project must be generated from the same project as the one running in the data server) and to the version of the data server program.

SW required on a client side

A Web browser capable of displaying Java applets must be installed on the client. It is recommended to use Internet Explorer (IE) version 5 and higher with Java Virtual Machine (JVM) build 3802. The version of JVM can be detected through the Java language console in IE (see the IE Help). The update of JVM or IE can be downloaded from www.microsoft.com.

4. APPENDIXES

4.1 ILLEGAL CHARACTERS

The names of non-visual objects (e.g. stations, variables, databases, etc.) cannot contain the character to be used to separate name parts when building the complete name of an object (see the *Naming rules* section in the chapter 2.4.2.6 MISCELLANEOUS).

The names of visual objects (i.e. project windows and components) can only contain the characters of the English alphabet, digits and the underscore character. The first character must not be a digit.

When naming files and objects of any kind in *Reliance* visualizing 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

4.2 FILE AND DIRECTORY STRUCTURE

4.2.1 Reliance program files

Install directory

(»c:\Reliance3« by default)

Reliance.exe – *Reliance design*
Reli_rt.exe – *Reliance runtime*
Reli_rts.exe – *Reliance runtime server*
Reli_srv.exe – *Reliance server*

Applet subdirectory

(»c:\Reliance3\Applet« by default)

Applet.zip – *Reliance J* applet program files
Extras.zip – *Reliance J* applet additional files
Help.zip – *Reliance J* applet help files

Components subdirectory

(»c:\Reliance3\Components« by default)

AcPicture.dll – Active picture
AcText.dll – Active text
Anim.dll – Animation
AXCont.dll – ActiveX container
Bar.dll – Bar
Bevel.dll – Bevel
Button.dll – Button
Circle.dll – Circle
ComboBox.dll – Combo box
Display.dll – Display
Ellipse.dll – Ellipse
Gauge.dll – Gauge
Grid.dll – Grid

IRC.dll – IRC
Line.dll – Line
MMPlayer.dll – Media player
Picture.dll – Picture
Pipe.dll – Pipe
Popup.dll – Local menu
Progress.dll – Progressbar
Radio.dll – Radio buttons
RoundBar.dll – Round bar
RTChart.dll – Real-time chart
RTTrend.dll – Real-time trend
SauHol.dll – Sauter - holiday program
SauProg.dll – Sauter - time program
Scale.dll – Scale
Text.dll – Text
TimeProg.dll – Time program

4.2.2 Visualizing project files

Project root directory

**.prj* – the visualizing project's main file containing basic parameters of the project
**.dsk* – the visualizing project's desktop file containing development environment settings related to the project

Alarms subdirectory

This is the default directory for storing the databases of status messages.

*CmdMsg.** – command messages
*ErrMsg.** – alarm messages
*SysMsg.** – system messages

Apps subdirectory

This directory is intended to contain external programs to be run from the visualizing project.

Data subdirectory

This is the default directory for storing the data databases.

MMedia subdirectory

This directory is intended to contain multi-media files used by the visualizing project (e.g. **.wav* format sound files).

Pix subdirectory

This directory contains picture databases of project windows and the shared picture database.

*Pix_*.**	– picture database of a project window
*Pix_Shared.**	– shared picture database

Profiles subdirectory

This directory contains runtime environment settings for individual users defined in the visualizing project.

Recipes subdirectory

This directory is intended for storing recipe files.

System subdirectory

This directory contains system databases of the visualizing project.

*Alarms.**	– defined status messages
*CommZones.**	– defined communication zones
*Computers.**	– defined computers
*DatabaseFields.**	– defined database fields
*Databases.**	– defined databases
*DatabasesConn.**	– connected databases
*DbReportItems.**	– defined report items
*DbReports.**	– defined reports

*DbReportsConn.** – connected reports
*DbTrends.** – defined trends
*DbTrendsConn.** – connected trends
*DbTrendSeries.** – defined trend series
*Dispatchings.** – defined control areas
*Folders.** – defined folders
IdReg.dat – information on assigned IDs
*NetConnectionGroups.** – defined network connection groups
*NetConnections.** – defined network connections
*Printers.** – defined printers
*PrintReportItems.** – defined print report items
*PrintReports.** – defined print reports
*PrintReportsConn.** – connected print reports
*RecipeItems.** – defined recipe items
*Recipes.** – defined recipes
*RecipesConn.** – connected recipes
*RTTrends.** – defined real-time trends
*RTTrendSeries.** – defined real-time trend series
*Scripts.** – defined scripts
*Stations.** – defined stations
*StationsConn.** – connected stations
*StationsNetConn.** – network-connected stations
*Users.** – defined users
*Variables.** – defined variables
*Windows.** – defined windows

Win subdirectory

This directory contains project window files.

**.scr* – project window file

4.3 POINT RATING OF RELIANCE VISUALIZING PROJECTS

The size of the visualizing 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 variables defined in the project.

The following rules hold true:

- Variables defined within the *System* station do not affect the number of data points.
- Each variable of a simple data type (e.g. *Bool*, *Byte*, *Word*, *String*, etc.) uses one data point.
- Each array-type variable uses one or more data points. If the element count is less than or equal to five, the variable 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 in design and runtime mode can differ. In design mode, all the stations defined via the ***Station manager*** are considered. In runtime mode, only the stations connected to the computer via the ***Project structure manager*** are considered.

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