TM812

APROL Operator Management

PERFECTION IN AUTOMATION
www.br-automation.com

Requirements	
Training modules	TM811 – APROL Runtime System
Software	APROL SuSE LINUX
Hardware	1 control computer, 1 controller

Table of Contents

1 Introduction....4
1.1 Objectives....5
2 Operator Management....6
3 Operator Rights....8
4 The OperatorManager....9
4.1 Operator Groups....10
4.2 Operator....16
4.3 User Management Active....20
5 Creation of a Graphic Block....22
5.1 Creating a Graphic Block in a Library....22
5.2 Linking Logic and the Process Graphic....29
5.3 Dynamic Element Ready for Use....31
6 Rights Management in the Library....32
6.1 Graphic Block Outputs....32
6.2 Graphic Block Outputs....37
6.3 Default Library Right Active....38
7 Summary....39
8 Appendix....40
8.1 Objectives Checklist (a few On-topic Questions)....40
8.2 Solutions to the Objectives Checklist....40

1 Introduction

A system is operated and controlled by a wide range of coworkers. Their levels of qualification vary, which raises the question: Who should have access to what?

We can start by dividing personnel into two groups: operators and service technicians. Already we have identified two unique profiles (groups) for system access. This is fundamental in preventing incorrect operating and system states.

User profiles are essential in every kind of process control. It is therefore important to integrate operator management into the dynamic elements during engineering.

Figure 1: Rights and responsibilities of system personnel

1.1 Objectives

After completing this seminar, you will be able to integrate operator management elements into the runtime and operating environments. In addition, you will gain insight into the process of library engineering, where operator management begins before ending in project engineering with the actual operators. Strictly speaking, the operator management finishes in the runtime or operator environment, because interventions can also be made here according to the assigned rights.

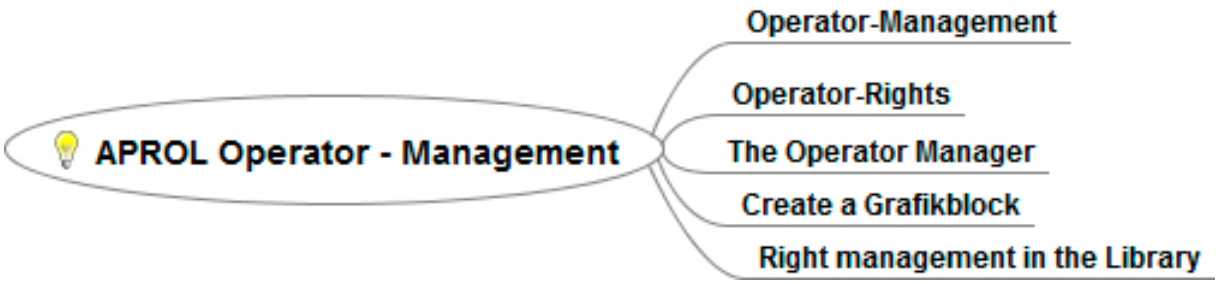

Figure 2: Overview

2 Operator Management

Operating a process control system requires a certain degree of know-how. For this reason, APROL provides protected access to the operating and monitoring levels. This allows the impact on the processes to be regulated. The user has the possibility to define ***an unlimited number of different rights***, which are then managed in ***operator groups***.

Operating processes take place in the process graphic using operator templates. These are the graphic elements of picture function blocks which are integrated into the system's logic diagram. Operator template keys and input fields are picture function block outputs which pass on the input value or keystroke to the rest of the logic on the function chart as a pulse or signal. One of the possible rights is assigned to each of the picture function block outputs. Picture function block outputs are generally located in a hyper macro, which allows the user to set operator rights using the hyper macro properties (or the graphic element properties).

Operator rights are not organized hierarchically. As a result, the configured rights are available to each operator group.

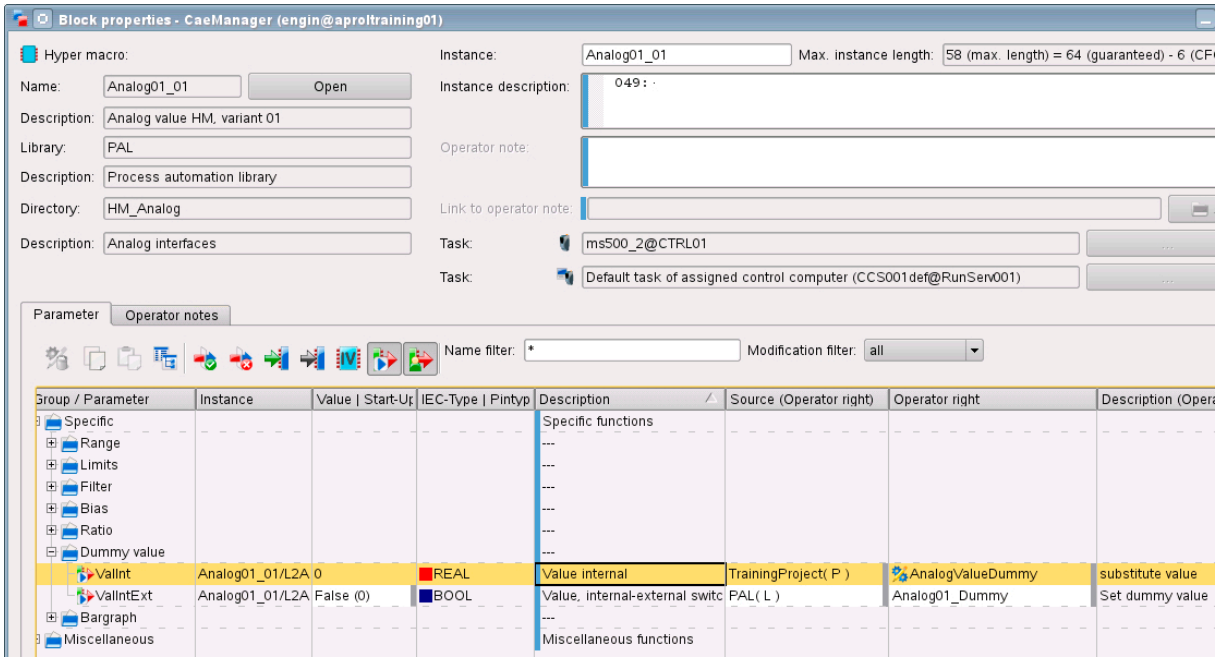

Figure 3: Library and project rights

Generally, there must be a differentiation between library and project rights. All function blocks receive default rights during library engineering which are naturally used in the project. If it is necessary to change these rights in the project, then project rights are referred to. This allows you to top up each action with additional rights. This considerably simplifies the structure of the operator management system.

Note:

Not every operation should have its own rights defined since later on there will be a maximum of 4 hierarchy levels (operator / shift foreman / maintenance / service technician). With this in mind, a hierarchically organized user management structure should be developed with as few rights as possible.

'Geographic' hierarchies are also normal. For example, somebody only obtains access rights to their own plant area. This is especially the case in the building services technology.

The following illustration should make this clear and shows an internal hyper macro view with graphic blocks and the respective outputs, which have the described project rights.

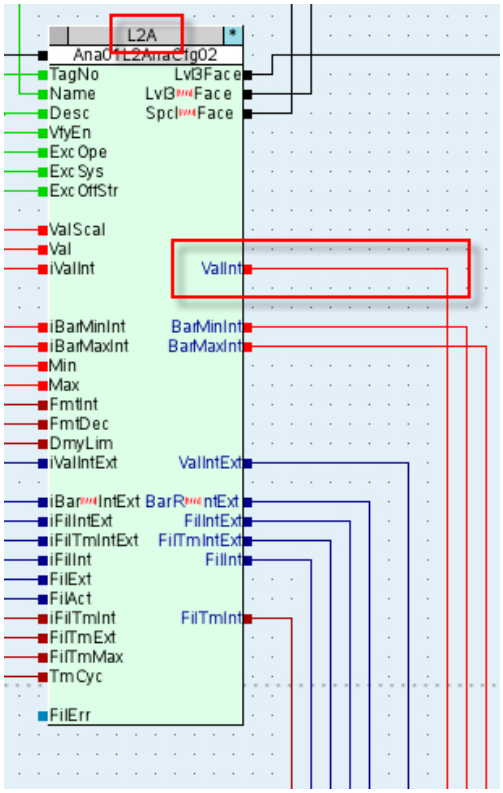

Figure 4: Graphic block view in HM (see L2A/ValInt in HM properties)

3 Operator Rights

Any number of rights can be managed during library engineering. In addition, the user can manage any number of rights in project engineering. The corresponding dialog is opened using the ***"Extras / Operator Rights"*** menu item. The ***"New"*** button appears after pressing the right mouse button and a new line is inserted. The line with the focus is deleted by selecting the ***"Delete"*** button. The name is checked by a validator, which prevents entering the wrong characters. The description can be anything (no syntax conventions).

Only rights which have been assigned a unique name can be assigned to online operator groups.

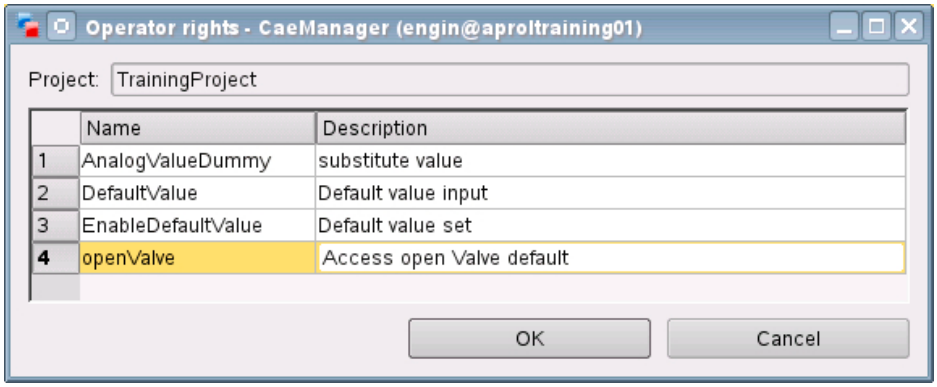

Figure 5: Operator rights

A name rule for rights is not mandatory, but it makes sense to name the rights according to the action. The additional purpose should also be combined for necessary project rights (mainly equipment, etc.).

4 The OperatorManager

The OperatorManager is started with the **"Extras / OperatorManager"** menu item. This manages the operators and operator groups. In the Operator Manager, you can define groups (a type of profile for rights), and you can then assign corresponding operators (operators, service technicians, project engineers, etc.) to them.

Figure 6: Starting the OperatorManager

The OperatorManager is split into two areas (left and right manipulation area), to firstly create groups (quasi user profiles) and secondly the operators.

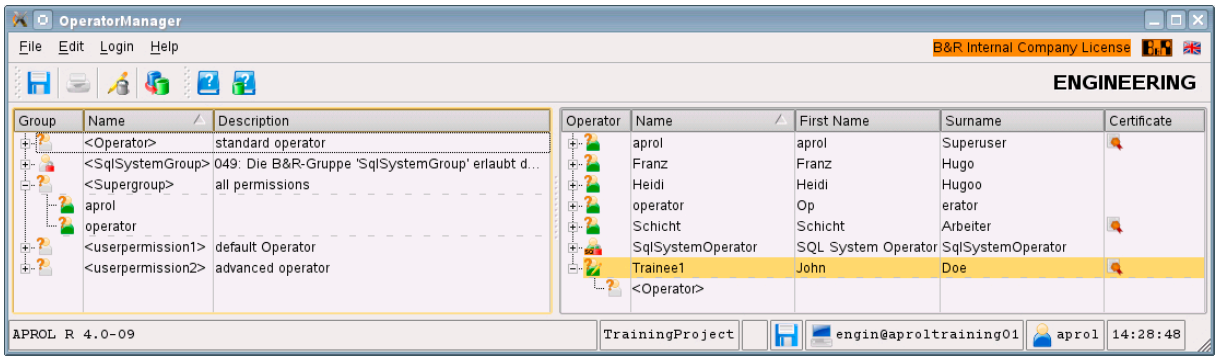

Figure 7: OperatorManager

4.1 Operator Groups

Operator groups are managed in the left area of the OperatorManager. Various editing and setting options can be accessed with a right mouse click.

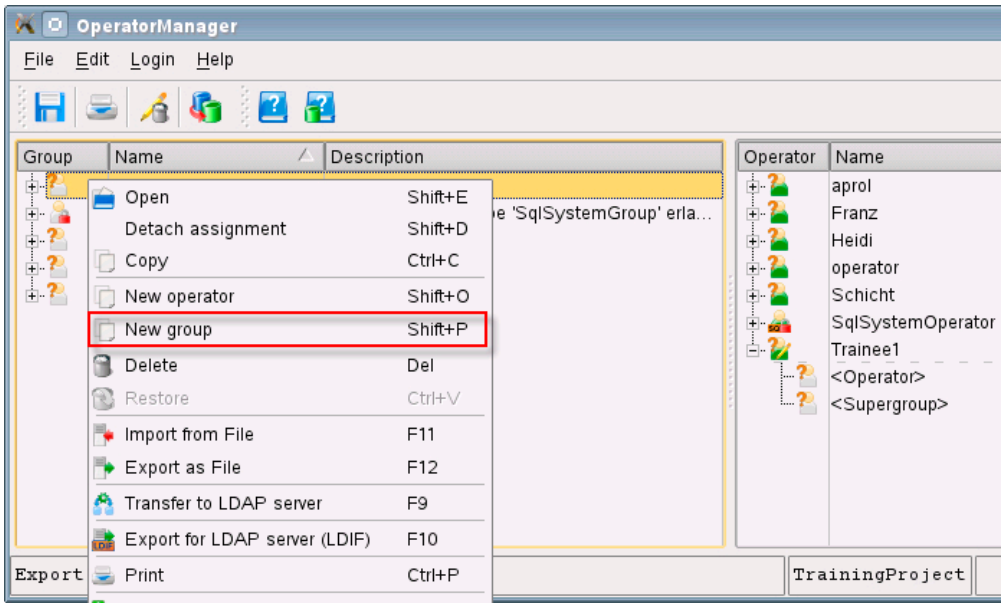

Figure 8: Creating a new user group

This dialog box defines several things: operator groups ***for runtime and operator system user interfaces***, as well as their ***access rights*** in the ***process graphics***, to the ***alarm and message system***, and to the **APROL *tools*** available in the user interface itself. The number of operator groups which can be created is unlimited. In the first tab, you can enter the name of the group and a detailed description.

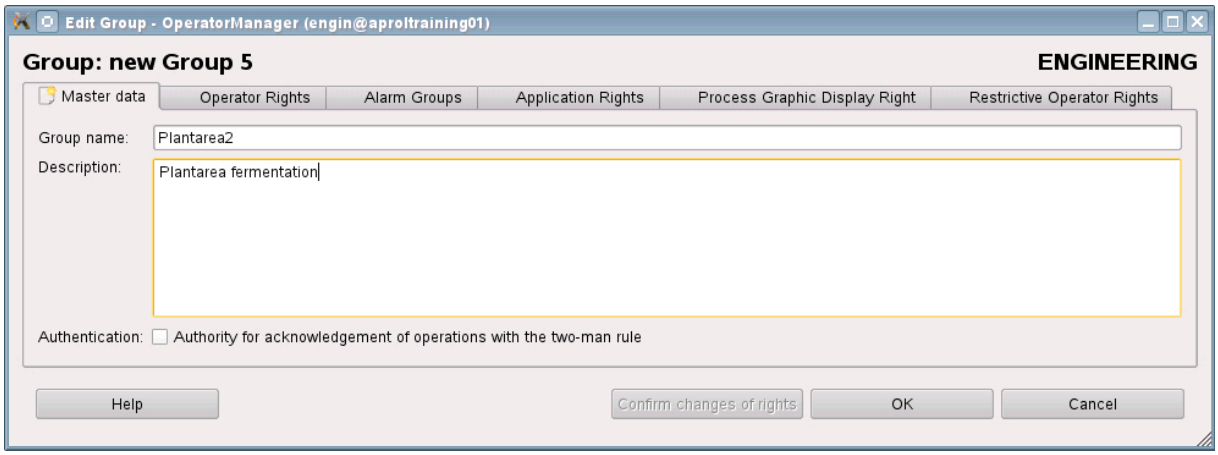

Figure 9: Creating a new operator group

Operator rights are assigned to the new operator groups under the ***"Operator Rights"*** tab. The newly created rights obtain a tri-state status (?). This means that these rights must be allowed or revoked explicitly in this group. If nothing is done with a tri-state right, it is allowed per default.

The assignment is made with a right mouse click. If this is done in the root of a library or project, then all rights can be set at once. Then these settings have to be confirmed.

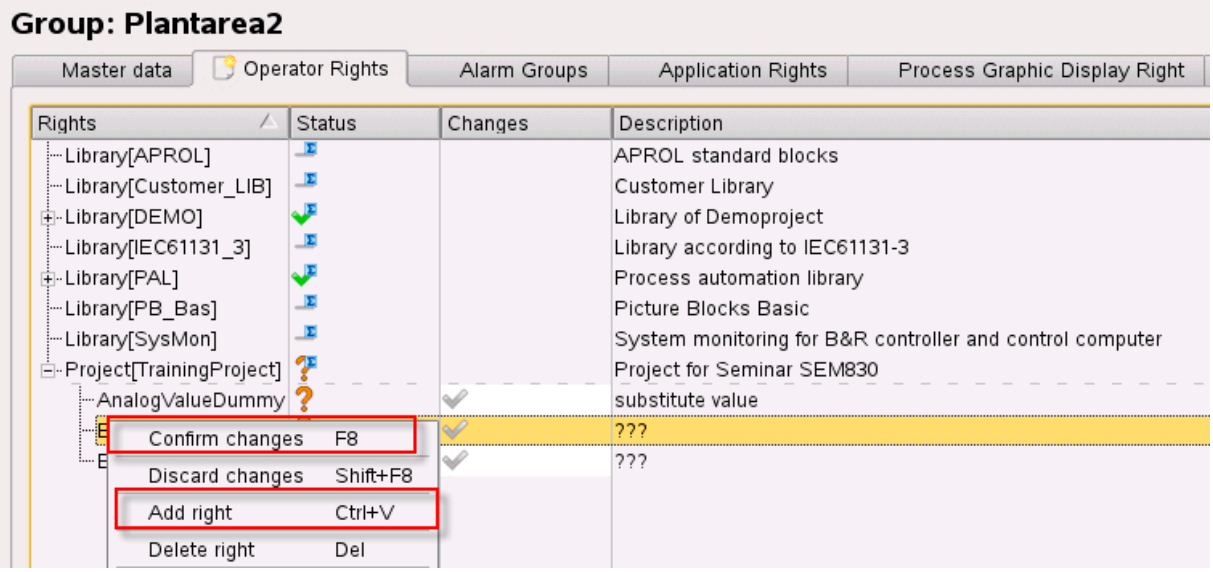

Figure 10: Online user's right in the process graphic

Note:

As previously mentioned, only clearly labeled rights are displayed. The rights can be freely assigned and deleted from the selected operator profiles. It is important to pay attention to the selection of the operator profile.

The ***"Alarm group"*** tab lets you define whether an operator in the selected group is offered alarms to be viewed ***"Show alarm"*** , can acknowledge them ***"Acknowledge alarm"*** , or lock them ***"Lock alarm"*** in the runtime and operator systems. The selection is made with the mouse. The selection must then be confirmed **(right mouse button / Confirm Changes)**

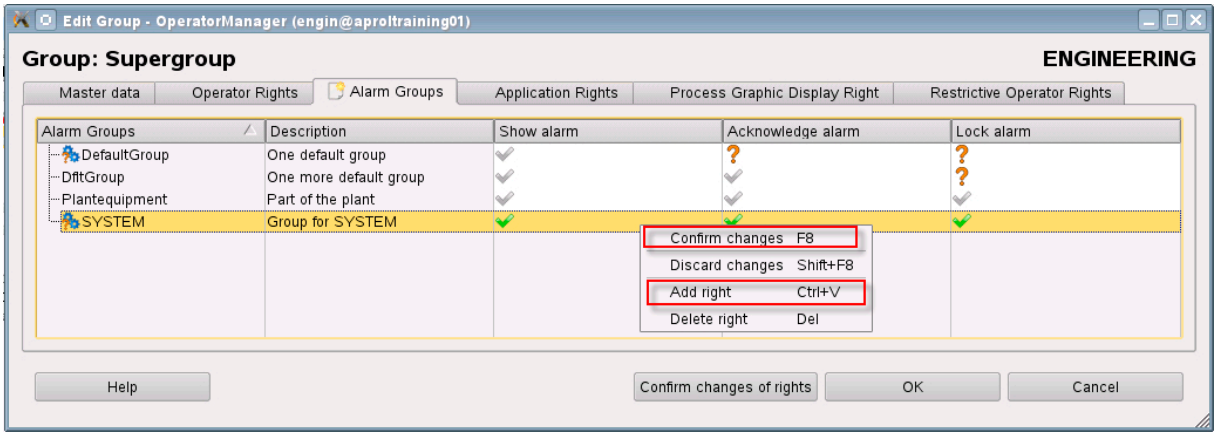

Figure 11: Online user rights messages and alarms

Alarms and message groups have to be defined when alarms/messages are configured. Alarm/message displays in the user interfaces for the Runtime System can be sorted or masked according to these groups.

Alarm groups which are used later have to be created and edited in an additional dialog box. The newly configured alarm groups then appear in the 'Alarm groups' view and must be configured in the respective groups which were created (default is a question mark to show the tri-state status). If this is not done, this alarm group can be operated without restrictions in the runtime environment.

Creating alarm groups

A dialog for managing alarm groups is opened using the ***"Extras>>Alarm Group Definition"*** CaeManager menu item.

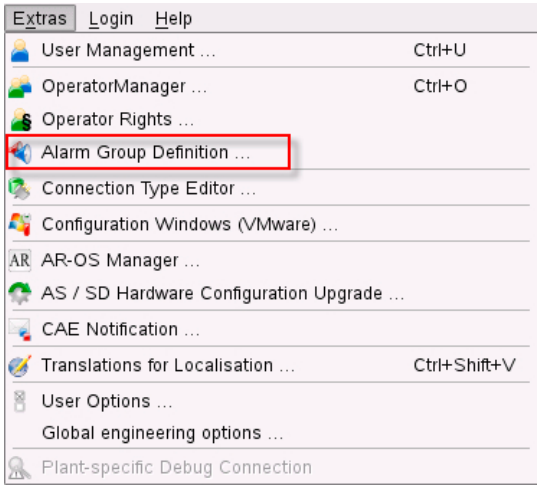

Figure 12: Opening the alarm group definition

In the open window, it's possible to create new alarm groups or manage existing ones.

To do this, right-click with the mouse in the alarm management window to open the following dialog box.

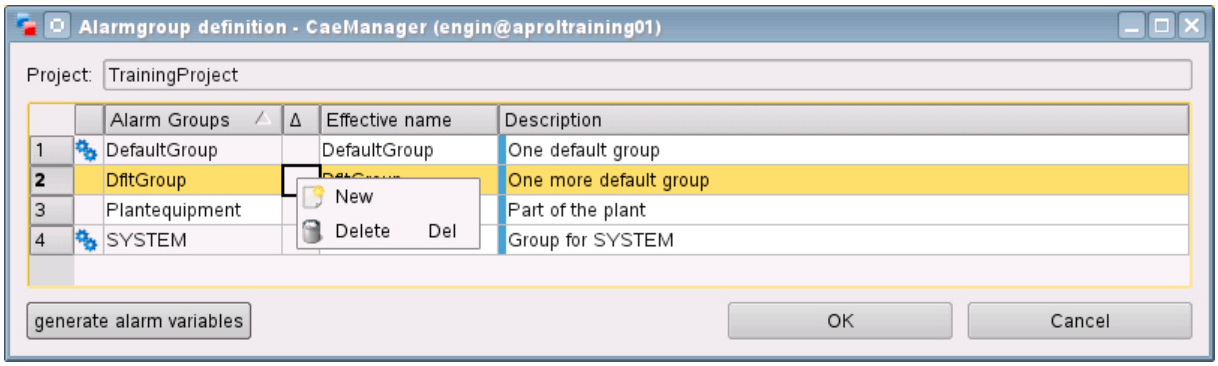

Figure 13: Creating a new alarm group

An alarm group is composed of the group identification, the effective name, and the description. The effective name and description can be changed by selecting them. The effective name is required for the alarm group variables, because these are C conform and are not allowed to contain umlauts or special characters. (Example (German) Alarm group: System**ü**berwachung >> Effective name: Syste-m**ue**berwachung). This is normally not necessary in the English language, but more so in many other languages.

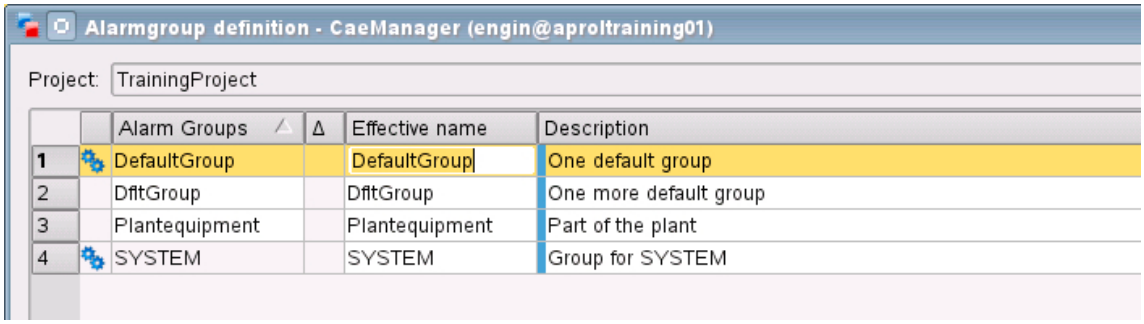

Figure 14: Editing alarm groups

Note:

Grouping alarms makes them easier to work with, since they can be organized according to function, location, or individual. This makes it possible for the alarms of specific system components to be made visible and/or acknowledgeable to only those users who will be operating those systems.

Example:

Disturbances of a fire alarm system can be reported directly on a terminal for the fire fighters. System operators are not informed about this, because this message does not apply directly to the process being monitored . An actual fire alarm, on the other hand, is signaled to both system operators and to the fire department, although only an authorized member of the fire department may acknowledge and deactivate it.

The APROL applications available in the runtime and operator system user interface can be configured for operator profiles under the ***"Applications"*** tab. Selection takes place with a mouse click, similar to the procedure for online operator rights.

Once you have selected the tab, use the Status column to define access to the individual functions when assigning this group, i.e. which applications may be started and which functions within these applications may be executed. Each operator group can have a different access option.

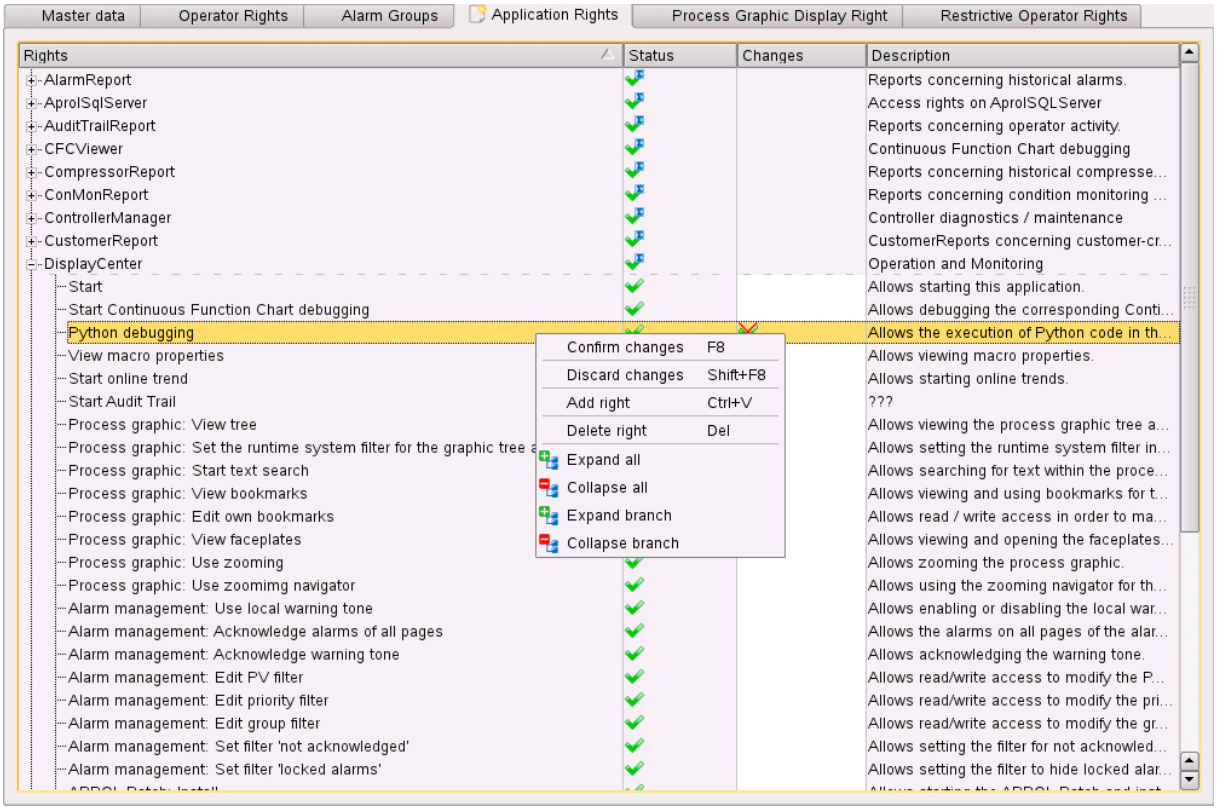

Figure 15: Online user rights for applications

Different actions are to be allowed or declined depending on the application. Looking at the DisplayCenter as an example, the actions "Start, Start online trend, Process graphic tree, Debugging, etc." can be found and evaluated per group (and thus qualification).

The next tab **"Process graphic display rights"** manages the access to process graphics.

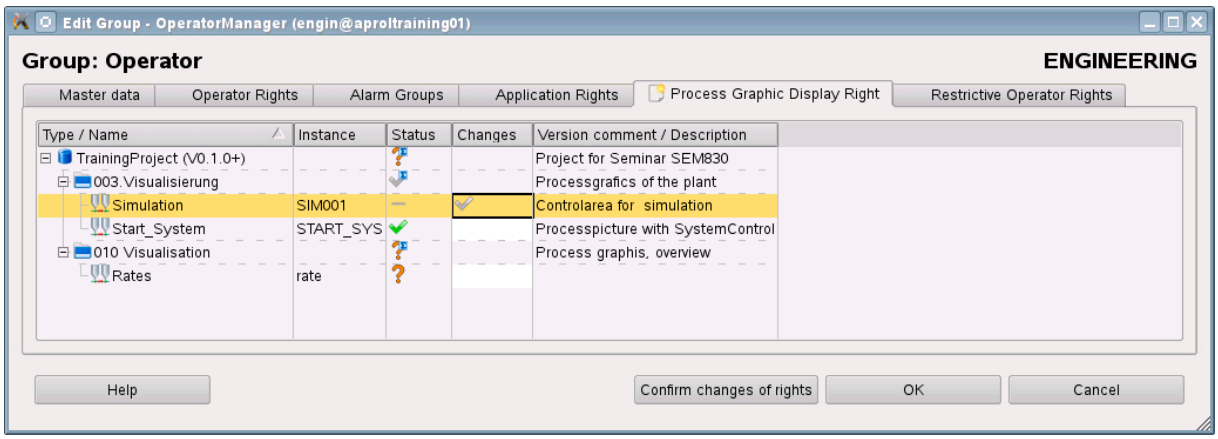

Figure 16: Management of access to process graphics

The operation of large and/or geographically branched plants is normally divided into more or less encased plant parts. They are accommodated here. An operator then only obtains access to certain plant parts and the necessary process graphics. Another aspect is that a service technician needs access to extra service process graphics which are not granted to plant operators.

Configuration of the access is done in the same way here as in the previously described procedures.

Only the **"Restrictive Operator Rights"** tab is left available.

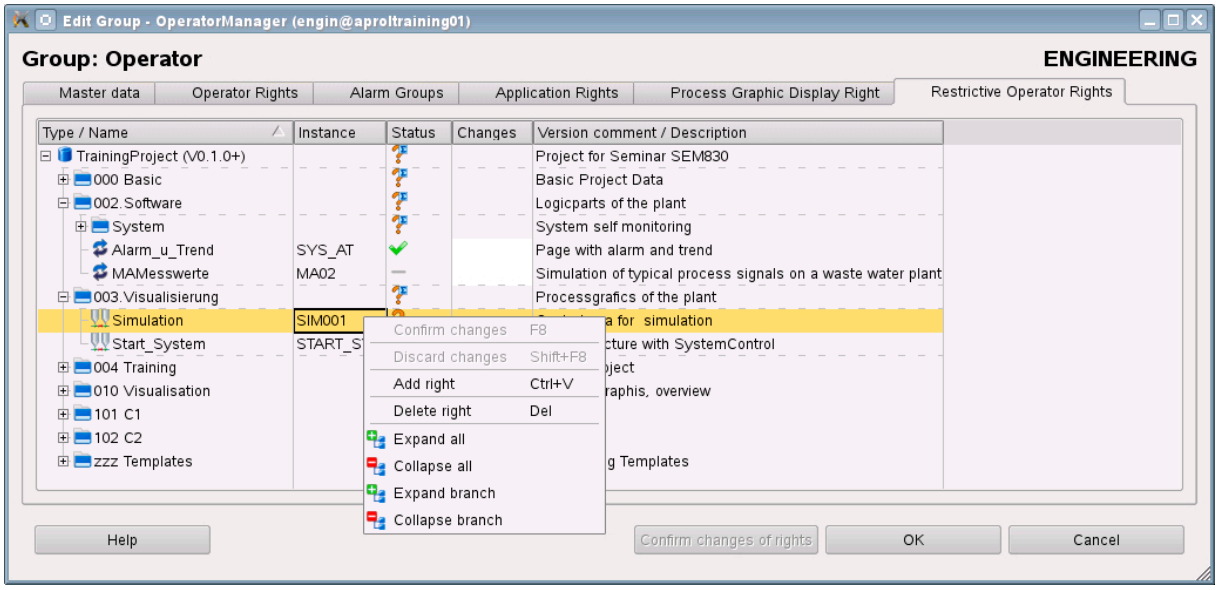

Figure 17: Configuration restrictive operator rights

Access to individual operating elements can be limited with this functionality. This serves to deny an operator access to a valve to which he has library rights granted in equipment 1, but are denied in equipment 2, even though it seems as though the same rights apply. This is done by listing the available CFCs and their assigned and limited rights.

Further details can be found in the APROL product documentation.

4.2 Operator

The **OperatorManager** can be used to create all operators – from the engineer to the system operator – who should receive access to the system configured in the project. Restricting access is done by assigning operators to the previously configured **operator groups**. It is possible to assign one operator to several operator groups. Assignment is done via drag-and-drop or an allocation in an open operator.

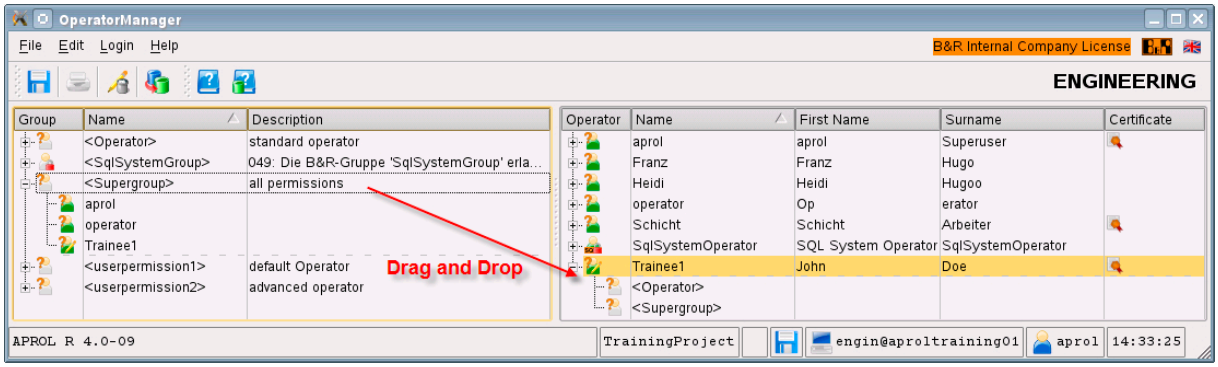

Figure 18: Assignment between a group and an operator

The **OperatorManager** is also an independent application that is available both on the Engineering System and the **Runtime System**. This makes it possible to add runtime or operator system operators, even without access rights to the **engineering system**. This may be necessary if new employees are hired and no engineering is planned at that stage. Thus, a temporary access can be granted for these employees. This must of course be returned back to the engineering. The export/import functionality supports this.

Note:

If the OperatorManager were a fixed component of the CaeManager, then an engineering process for operator management would be necessary.

The OperatorManager can be started from the KDE control bar or directly from the CaeManager **(Extras / OperatorManager)** . One of the available projects can be selected In the **"Project selection"** dialog. (the dialog only appears if more than one project is in the database)

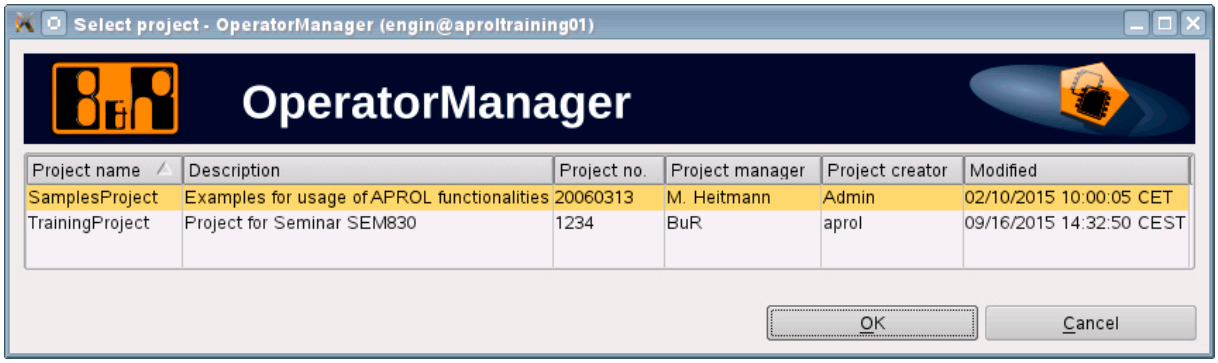

Figure 19: Selecting a project in OperatorManager

The **"Edit operator"** dialog is opened in the right area via the **"New operator"** context menu item in the **OperatorManager**.

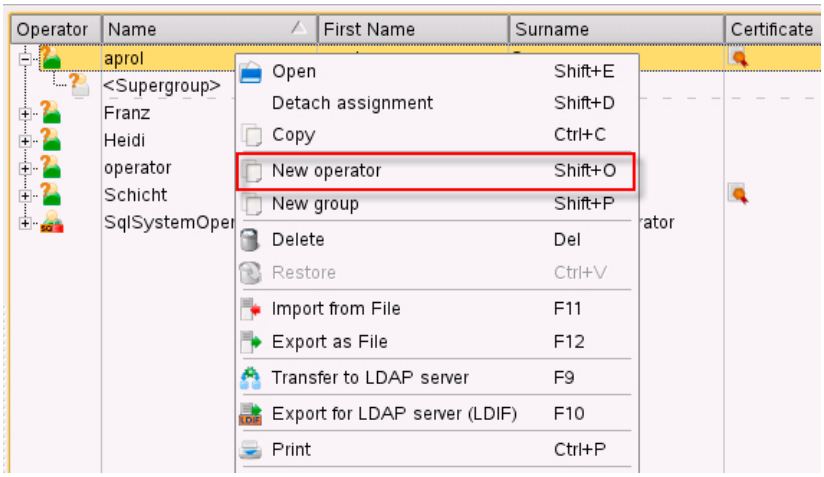

Figure 20: Creating a new operator

A new operator is created in this dialog in a similar manner to that of the user management. Personal data including login and password information is entered in the first tab. The preferred web browser and an "idle time" can also be defined.

Each operator can also be allowed to sign PDF documents. This is especially important when production batches have to be confirmed. Each report (standard APROL and customer-specific reports) which are stored as a PDF can be signed and validated in this way. A personal certificate must be created for the operator to realize this.

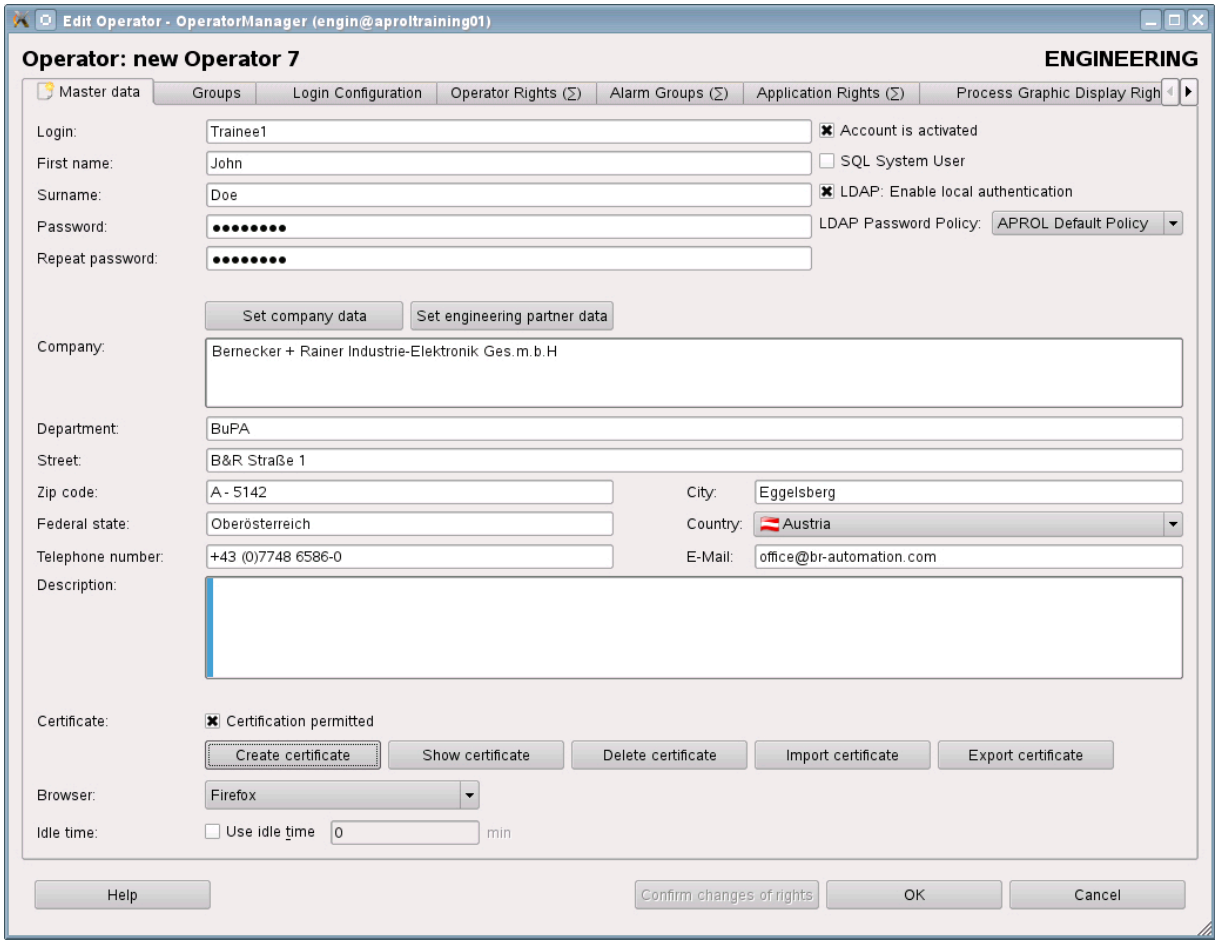

Figure 21: Creating an operator in the OperatorManager

The operator groups can be assigned in the **"Groups"** tab. This can be done by using **drag-and-drop** as described before or with the right mouse button and the respective action. (**"Attach"** and **"Confirm changes"**)

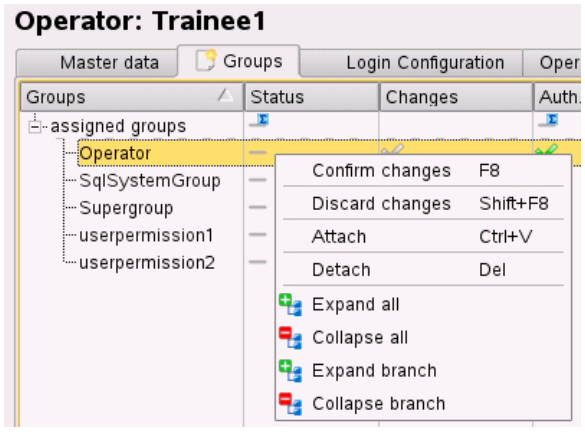

Figure 22: Assigning operator groups

The **"Login configuration"** tab contains options for configuring the way the operator logs on to the runtime system. The different login hardware do not exclude each other.

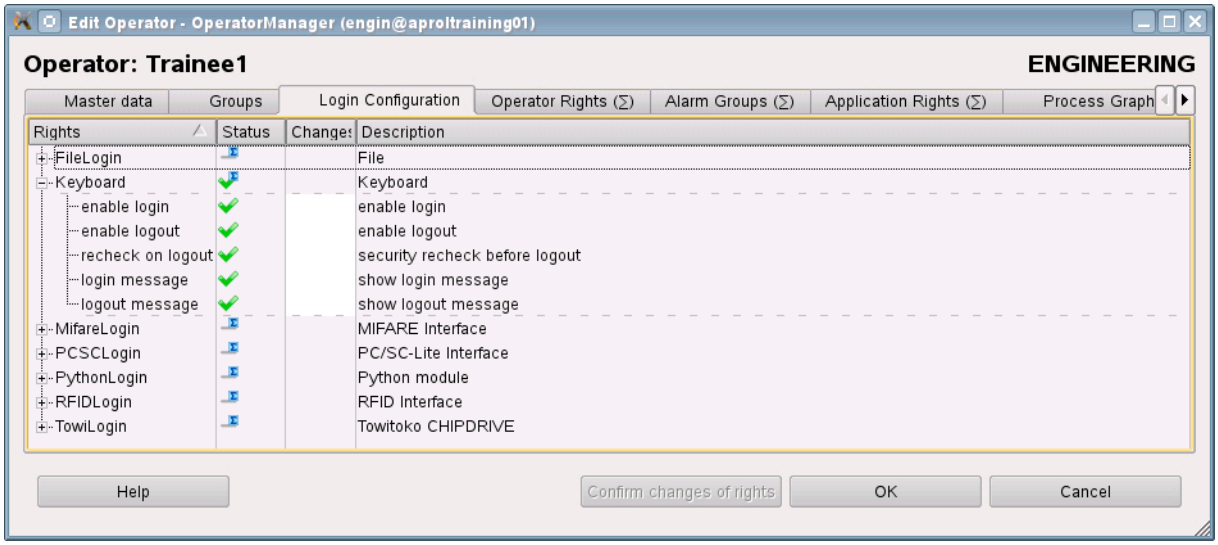

Figure 23: Hardware login configuration

Functionality	Description
enable login	The operator is authorized to log in using the selected hardware.
enable logout	The operator is authorized to log out using the selected hardware.
check login	The password is requested in addition to the hardware login.
recheck on logout	If the user logs out, then he must confirm this separately.
login message	The operator receives a message when logging in.
logout message	The operator receives a message when logging out.
operator id	An "operator id" must be entered for FileLogin, PCSCLogin,RFIDLogin and TowiLogin

When logging in with the ***"keyboard"***, the name and password must be entered into the login fields using the keyboard. With ***"PythonLogin"***, the login mechanism can be programmed freely. ***"FileLogin"*** lets the user login using a file. This procedure is used e.g. if the user logs in over a modem connection, and is not covered in this seminar. ***"TowiLogin"*** allows you to login using a card reading device from Towitoko. Further hardware login possibilities are RFID, PCSC and Mifrare.

Note: Hardware login

An operator can also be logged on by a hardware-supported login. APROL currently supports Towitoko card readers, which can be used as either "stand-alone" devices or integrated into the keyboard or computer. Using a hardware-supported login that must be combined with a keyboard login and configured with or without a password query makes the best use of the user stack advantage.

In the other tabs (Operator rights, Alarm groups, Application rights, etc.), the sum of the assigned operator groups are displayed in a simple manner and allow the cumulative rights to be checked. Only read access is provided here. You must switch to the operator groups which were mentioned in order to change a right.

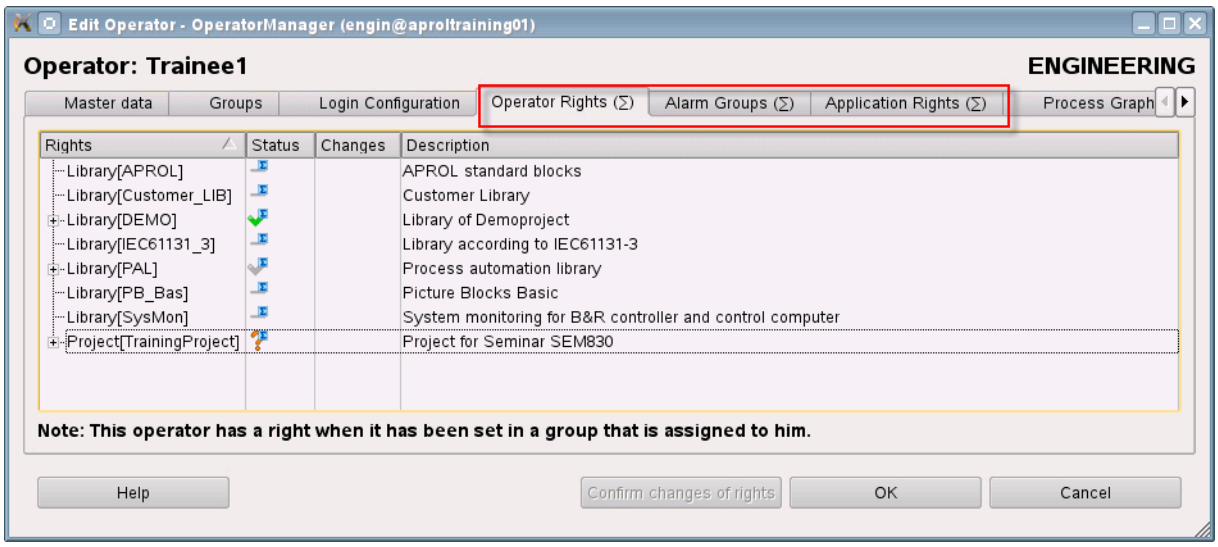

Figure 24: Assigned rights view

4.3 User Management Active

This completes the implementation of operator management for runtime and operator environments. Every employee now only has access to the functions that he or she needs and is trained for.

In most cases, the following levels are appropriate:

- Guest
- Normal system conditions (operator)
- Service technician
- Engineer
- System administrator

Most often, the operator groups are arranged in a similar fashion. The organization will vary from system to system. Some prefer individually named assignments, while others arrange by levels. Whatever method is chosen, the most important thing is flexibility.

Figure 25: Operation and access are defined

Exercise 1:

Create a new project access right and use it in one of the hyper macros you have created. Overwrite the right of the dummy value of the PAL analog hyper macro. (substitute value specification)

Exercise 2:

Create a new operator group and assign all operator rights to it and one limited right. Also assign alarms and applications.

Exercise 3:

Create two new operators and assign the created operator groups to them. Then check the 'different' functionalities in the runtime and operating environments.

5 Creation of a Graphic Block

In APROL, both controller and control computer logic, as well as all picture elements necessary for display, are configured in a function chart. This type of programming prevents a ***hidden engineering***. Because both logic and graphic parts are displayed, and are made apparent to the bus traffic between the systems. Because the image (e.g. an LED display) cannot be linked with the picture function block, the graphics section is placed in the chart with a ***graphic block***, and the variables used for dynamization (animation) are designed as PINs.

The procedure is explained step-by-step using a simple example.

5.1 Creating a Graphic Block in a Library

A new block can be inserted in a library with the ***„File>>New/“*** menu or ***„right mouse button >>Create project part/“***. A function block can be a function, function block, ST function, ST function block, SFC function block, hyper macro, control system function block, or as in this case, a **graphic block** or a UCB block. All of these and further possibilities will be explained in a future seminar.

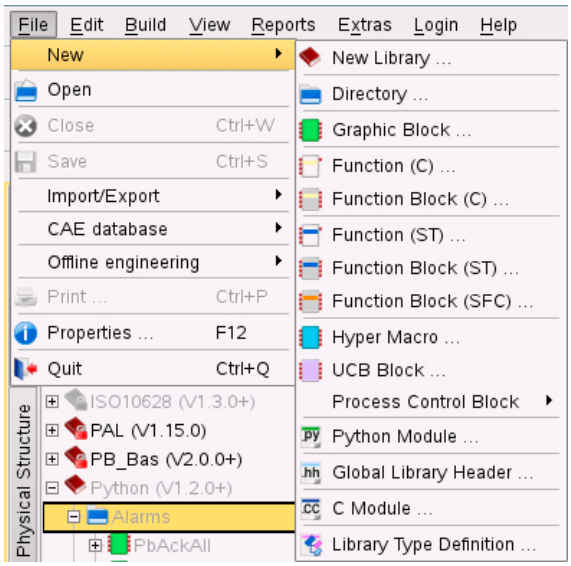

Figure 26: Creating a new graphic block

You must first change from the **Projects** working section to the **Libraries** working section in the Cae-Manager.

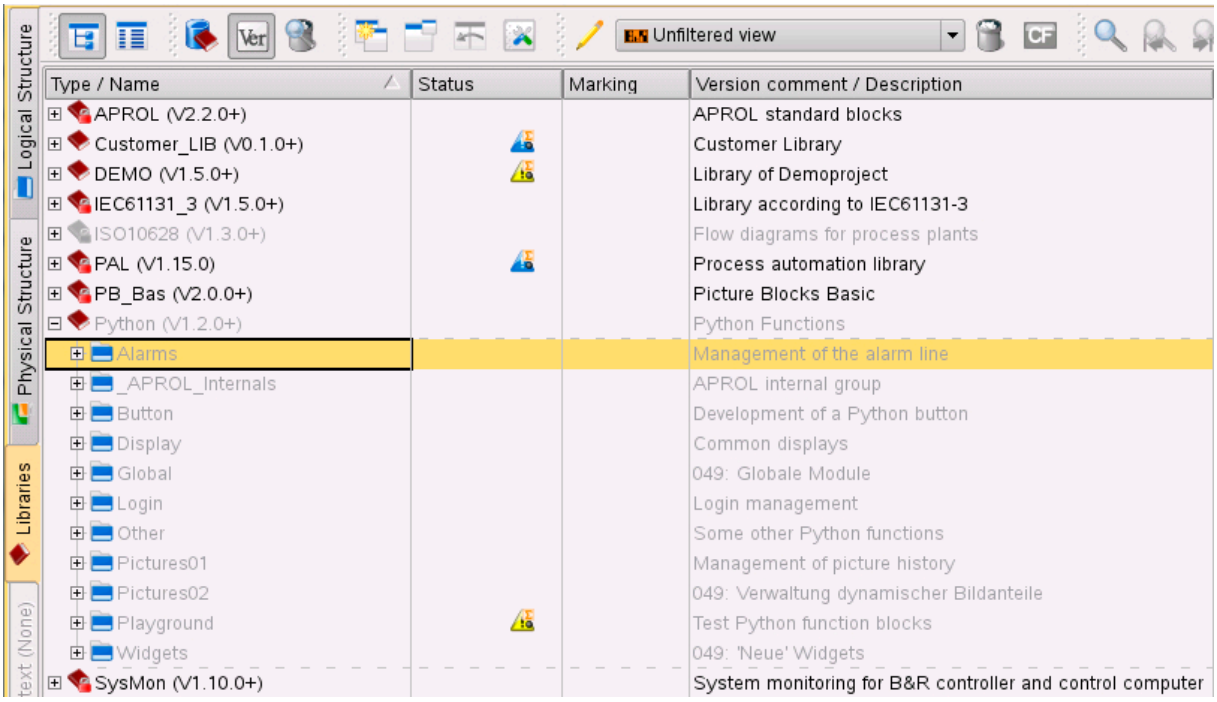

Figure 27: Selecting a library

After selecting a library in CaeManager and selecting a directory, blocks can be inserted into the project directory structure, as is the case with project engineering.

Note:

The nesting depth of the directory structure is not limited. Blocks and other directories can be created in a directory.

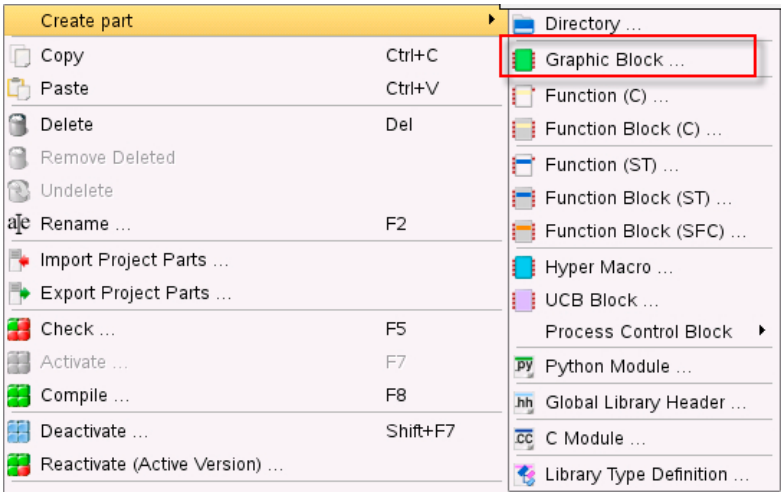

Figure 28: Adding a graphic block

The window that then opens is where the name and description now have to be specified:

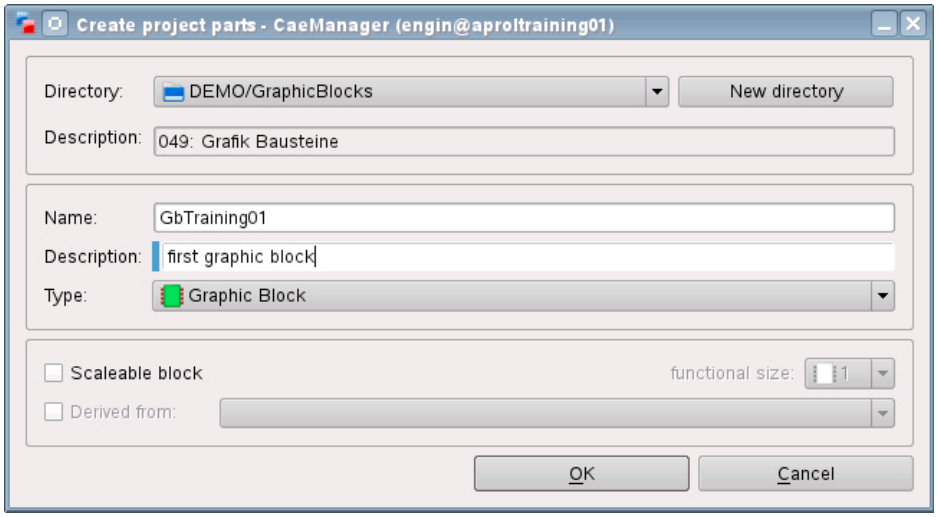

Figure 29: Creating a graphic block

The block type can be changed with the combo box. Scalable blocks can be created, and their size and functionality can be changed in the CFC. (pull down). Typical example for this is an AND_BOOL.

The graphic block is now created and can be opened for editing.

Double-clicking on the preview opens the graphics editor. The user can create a new picture here or modify an existing one. All changes are naturally subject to APROL version management.

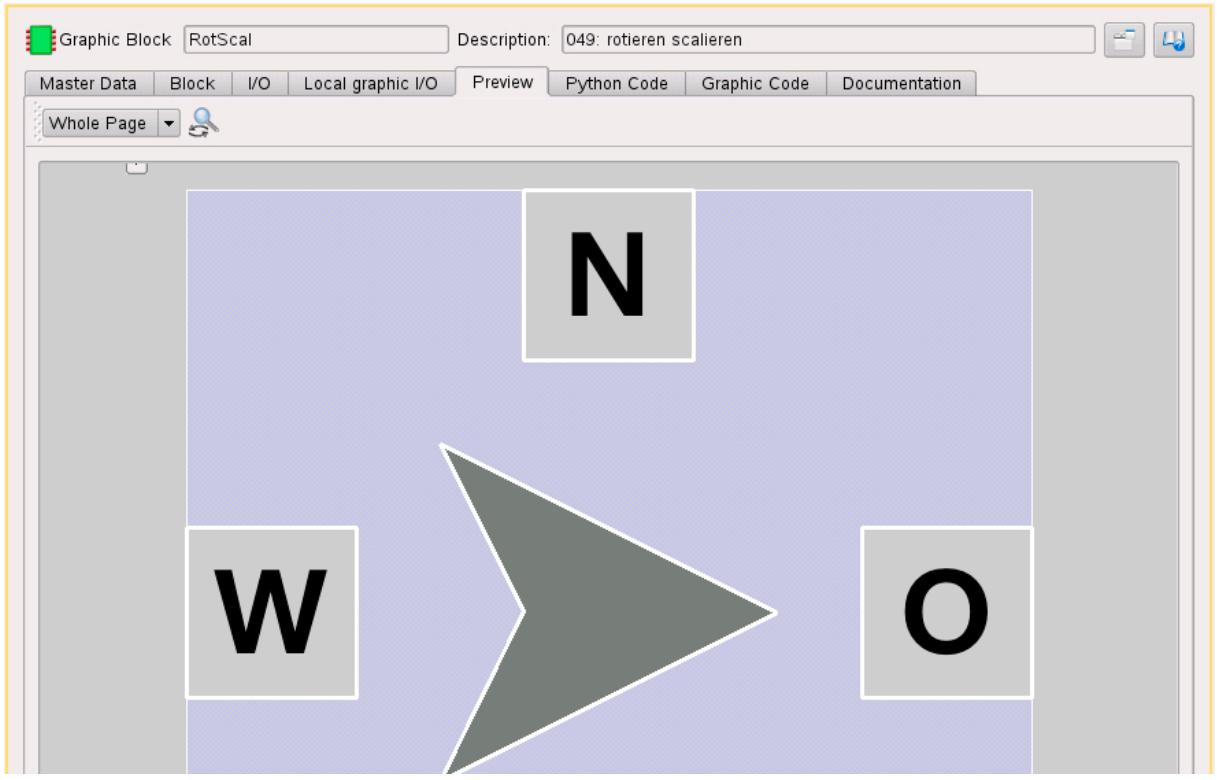

Figure 30: Picture block preview

After selecting the object(s) created in the Display Editor (multiple selection also possible), they can be animated.

Figure 31: Animation toolbar

The following functions are available for this dynamization:

Type of dynamization	Description
Blinking	Objects can be blink in many different ways depending on a process variable (background, inverted, different frequencies, etc.).
Colors	Objects can change their colors depending on a process variable. Background and foreground colors, fill patterns, and line types can be changed.
Rotate	Objects rotate around a definable fixed point depending on a process variable. Picture scaling must take place during animation.
Move	The objects change their position (are moved) in a process graphic depending on a process variable. Picture scaling must take place during animation
Scaling	The objects change their size depending on a process variable, whereby the x and y axes can be defined independently of each other. Picture scaling must take place during animation.
Percent	Objects are trimmed depending on a process variable and no longer displayed in full size. (bar graph display, etc.). Picture scaling must take place during animation.
Text (text)	A process variable can be displayed differently depending on its value. This is the case if the correct mask is set. (+xx.xxx) If this mask is not set, there will be a text display. (Usually used for text output)
Text list	Texts defined in a list can be displayed depending on a process variable.
Image list	This allows project-specific images (SVG, etc.) to be replaced or displayed.
Invisible	This allows objects to be displayed or made invisible.

Details can be found in the 'APROL product documentation'.

To animate an object, it needs to be connected to a process variable. Because this link is made in the function chart, only the place holder can be defined in the following menu. This placeholder is then created as PIN in the corresponding graphic block.

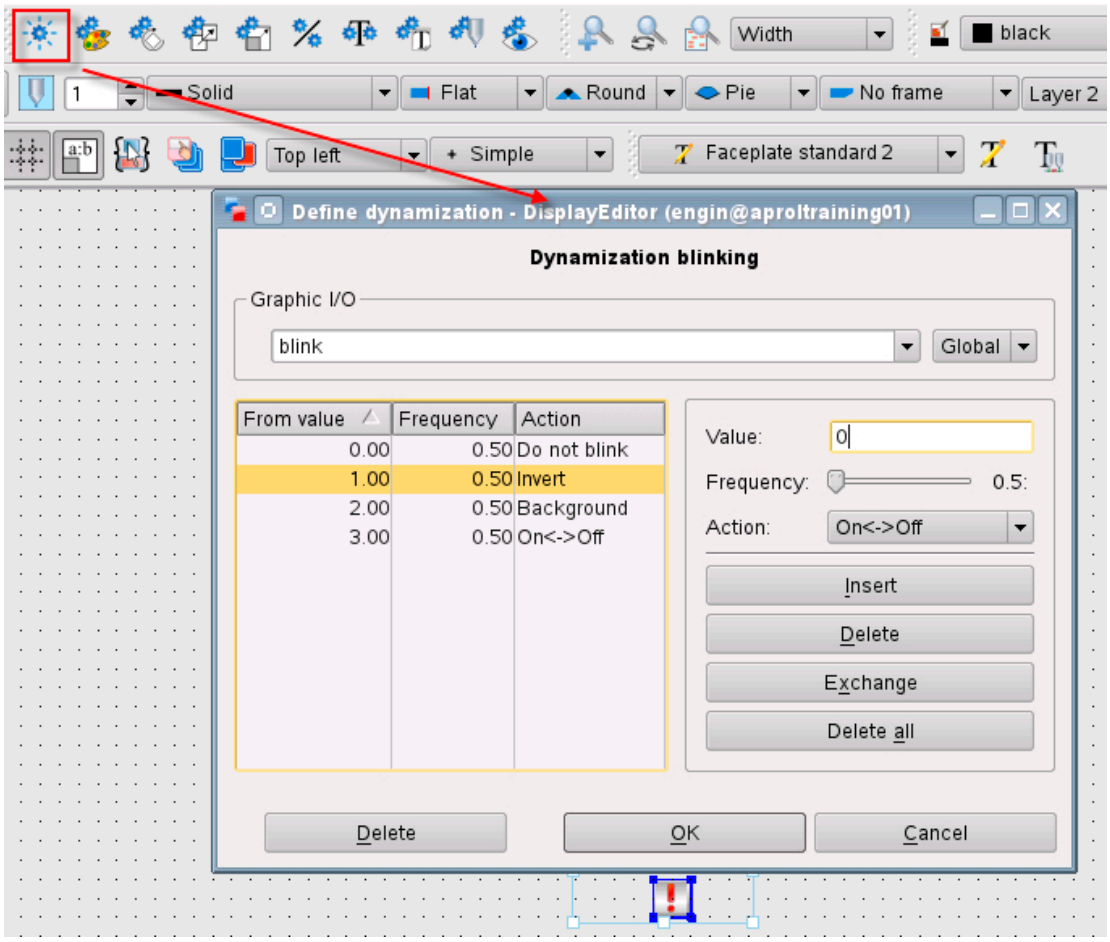

Figure 32: Blinking animation type example

The placeholder is created with the entry ***[Graphic I/O]***. Exactly this variable will be available in the graphic block.

After the variable has been entered, actions resulting from the value of the variable can then be defined. In the case of blinking, this can be **'Not blinking, Inverted, background, On < -> Off'**. Other settings are available for the other types of dynamization.

The object must be saved after the dynamization is configured, and the DisplayEditor closed. You return to the CAE view and can continue working with the graphic block.

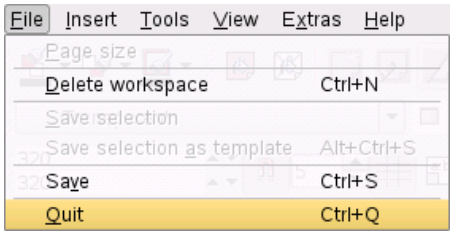

Figure 33: Creating and saving the graphic block

You can further configure the graphic block by clicking on it. Under the **"Block"** tab, the pins for the block can be assigned I/O variables. The placeholder mentioned before is thus converted to a PIN.

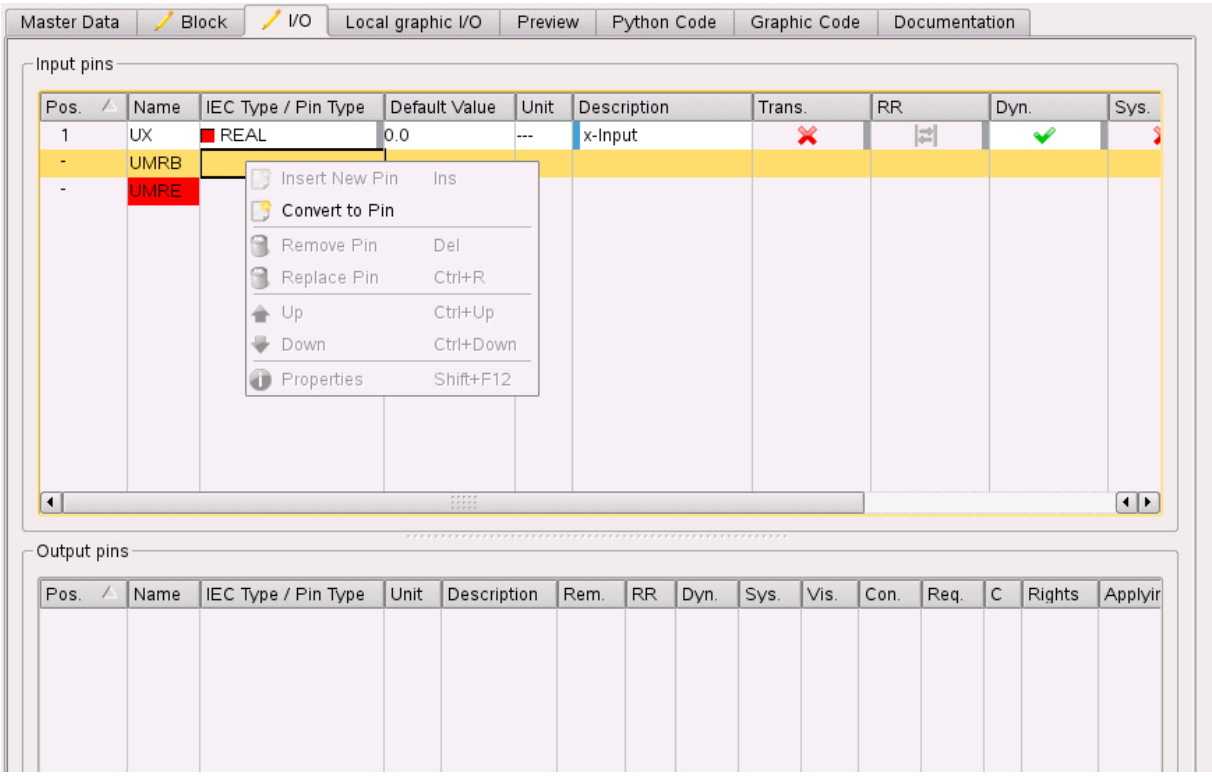

Figure 34: Configuring the graphic block

The best method of assigning so-called **"Graphic I/Os"** is using the **"I/O"** window. Incorrectly configured "Graphic I/Os" are seen here in red. With the right mouse button, these variables can be declared as input or output pins. Finally, the properties are changed by this PIN to make the functionality logical. (Data type, description, default value, operator rights)

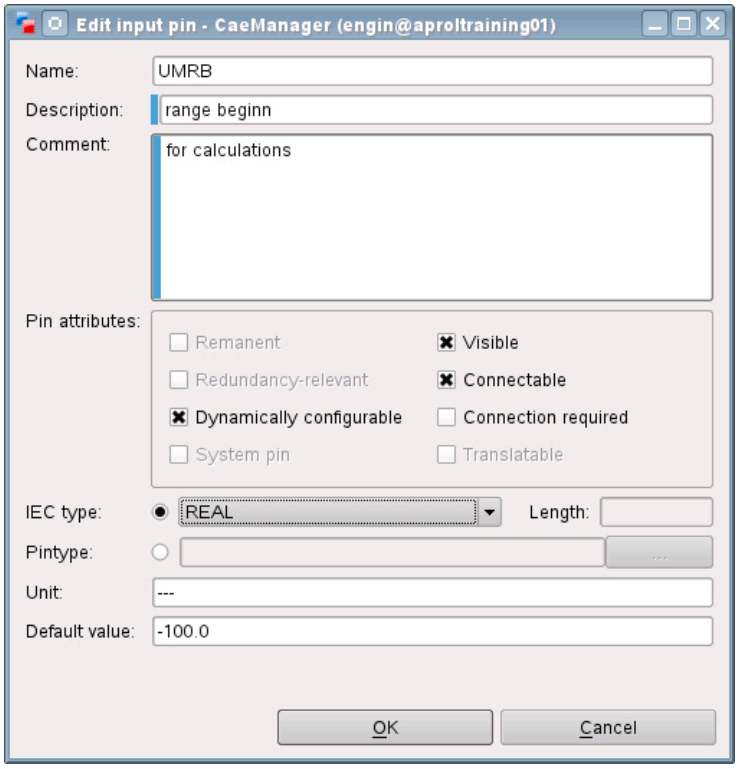

Figure 35: PIN properties

The block appears in the chart and in the process graphic, the same as it is displayed in the function block view and program view. If red entries are visible, then this is because there are animations that are not yet included in the graphic block.

The function block created is then saved, versioned, activated and compiled and is then available in both project engineering and library engineering.

Note:

Of course the function block is only available when the corresponding library for the project is linked (see project properties). Apart from that, the block is not allowed to be stored with the 'Library internal' status.

5.2 Linking Logic and the Process Graphic

After the library is closed and a function chart is selected in CaeManager, the user can use the Context view to see libraries linked to the project.

Picture function blocks can now be integrated like a function block in the function chart (or in a hyper macro (library engineering)) and be linked with the PV whose process value it should illustrate.

Figure 36: Combination of logic and process graphic

After the changes have been saved in the function chart, the graphic element for the graphic block can be placed on the process graphic. The corresponding process graphic is opened for this. It's now possible to navigate to the desired graphic block using the Context view. After selecting the function chart in the process graphic, the desired graphic blocks can be selected and placed on the process graphic using drag-and-drop. Maintaining the picture elements is guaranteed by connecting the block in the function chart one time.

The following image shows the placement of the animated picture function blocks:

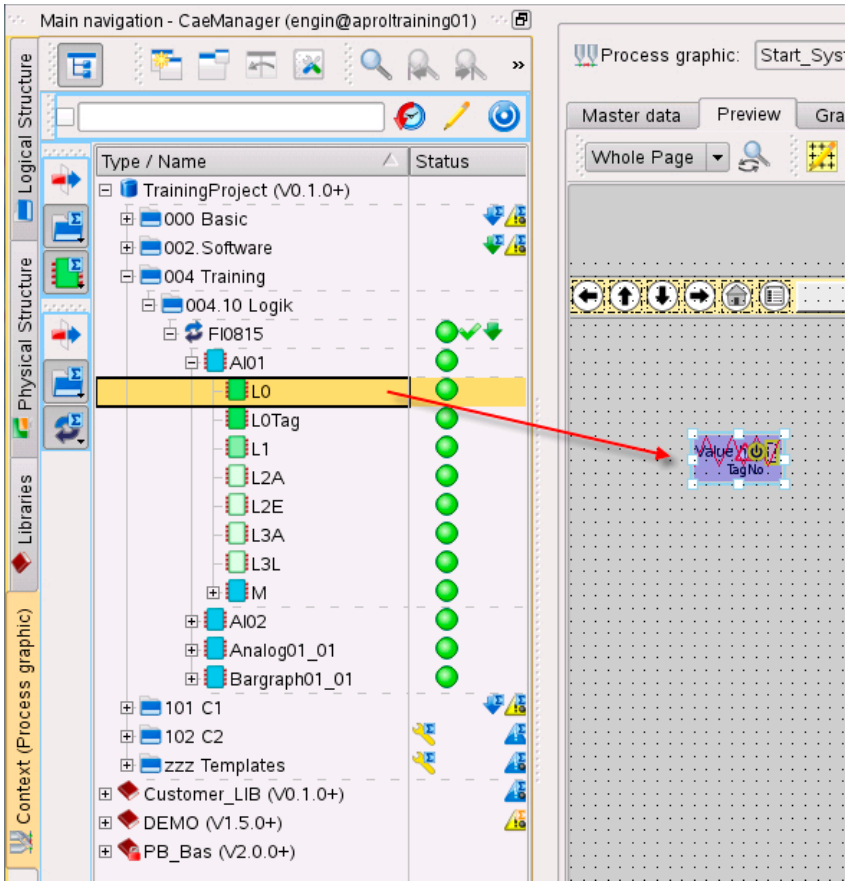

Figure 37: Inserting a display driver function block

A picture element placed in the process graphic can be subsequently moved anywhere on the screen. Additional functions are available with a right mouse click.

For fast orientation, the most important information is also displayed using a tool tip when the mouse pointer is placed over a dynamic picture element.

In this way, the source of the picture macro can be determined at any time, which helps determine how animations should be provided for this graphic block.

Note:

A direct dynamization is possible and must be mentioned at this stage. This means that the source of the dynamization is the background image or direct the library block. It must be ensured that the placeholder variables are linked to process variables in the **"I/O mapping"** tab. This represents a typical SCADA functionality. (Supervisory Control and Data Acquisition) This relinquishes the before mentioned DCS comfort. (Distributed Control System)

5.3 Dynamic Element Ready for Use

By creating a graphic block, we are now able to implement our animations. The complexity of the graphic blocks will increase in later versions. Nevertheless, their fundamental uses will remain the same.

Exercise 4:

Create a picture macro of your choice, the corresponding picture function block, and use it in the project. Check your work in the RUNTIME and operating environments. Ideally, each group creates its own type of dynamization to present to other participants.

6 Rights Management in the Library

Rights management begins in library engineering, and ends with the creation of operators for the runtime and operating environment. This chapter describes the context of operator management in the library engineering

6.1 Graphic Block Outputs

Graphic block outputs are created using controls and widgets.

What is a control?

A **control area** is basically not visible in the DisplayCenter. It's more of a **mouse-sensitive area** that executes an action when clicked. A **change in the mouse pointer** within the DisplayCenter indicates **something that can be operated**.

The "Highlight buttons ..." project option identifies the **control surface** with a **bounded area** when the mouse pointer is placed over it.

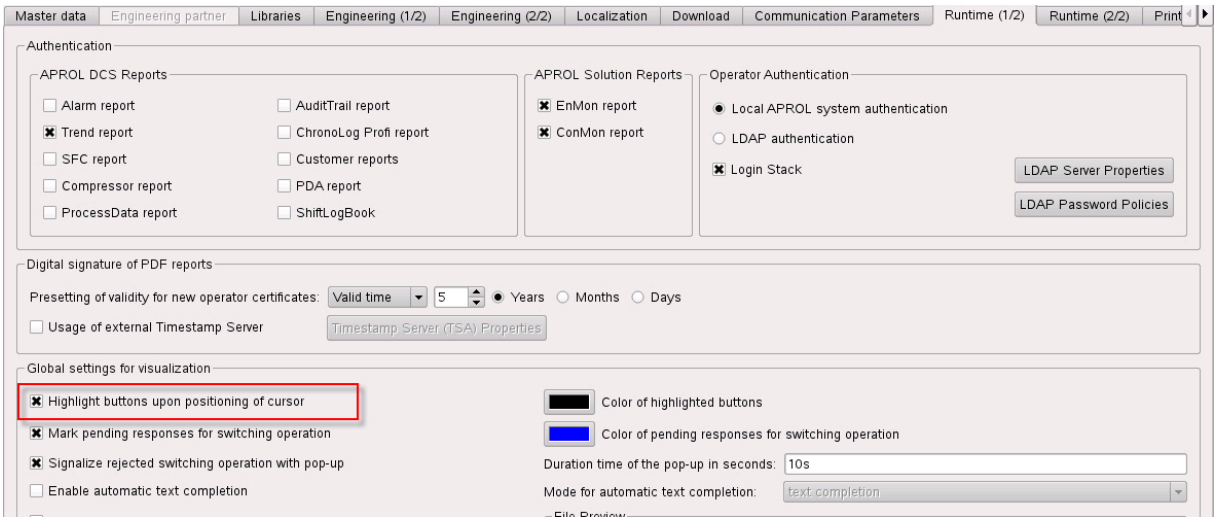

Figure 38: Highlight buttons

The following "controls" are available:

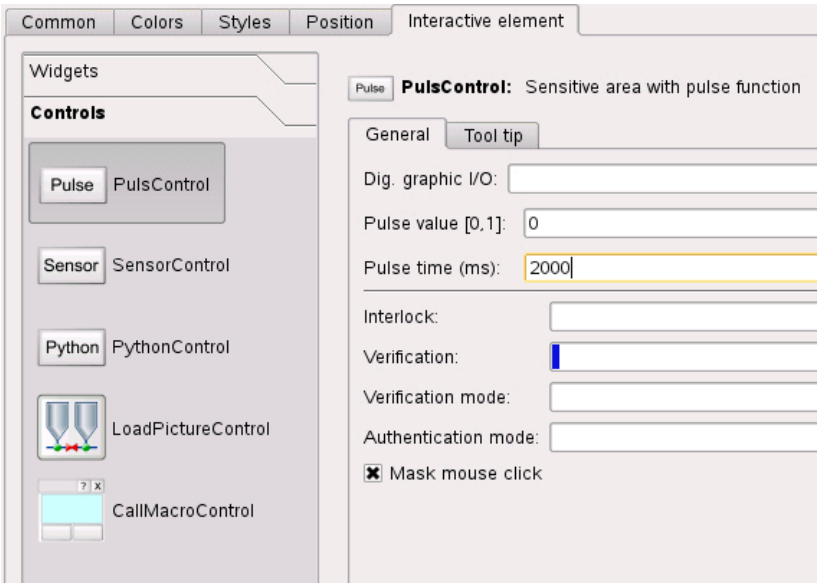

Figure 39: Available controls

Control name	Short description
Pulse control	Provides a mouse-sensitive are without the visualization of a button. Implements the functionality of a pulse generator. The switching operation is triggered for the duration of the set impulse time when pressed. The pulse button can then be used if only a short pulse is needed to control the actuator.
Sensor control	Provides a mouse-sensitive are without the visualization of a button. Implements the functionality of a button. When pressed, the switch is triggered and is only in effect during the time it is pressed. The sensor button can then be used when switching should be controlled by the operator but limited in time.
Python control	Provides a mouse-sensitive are without the visualization of a button. The python button behaves like python code and also provides a dynamically labeled key that provides visualization for switching. The python button can then be used to implement complex visualization applications that require explicit logic definitions using python code.
Load picture control	Provides a mouse-sensitive are without the visualization of a button. When pressed, the load picture button initiates a change to the process graphic instance, at which time the zoom factor and opening position can be transferred. With the load picture button, it's possible to jump to and zoom process graphics.
Call macro control	Provides a mouse-sensitive are without the visualization of a button. The call macro control triggers a Macro type process variable that can be used to switch to other macros. From a graphic block, the call macro control can be used to trigger other graphic blocks that are available in the continuous function chart.

To create a **control**, use the **Tools / Draw / interactive element / Buttons** menu item in the **DisplayEditor** or select the **Interactive element** button in the toolbar on the side.

After you have defined the end of the **element**, a window appears in which you can set the type of control/widget and configure it. In our example, a pulse is sent for 2,000 ms.

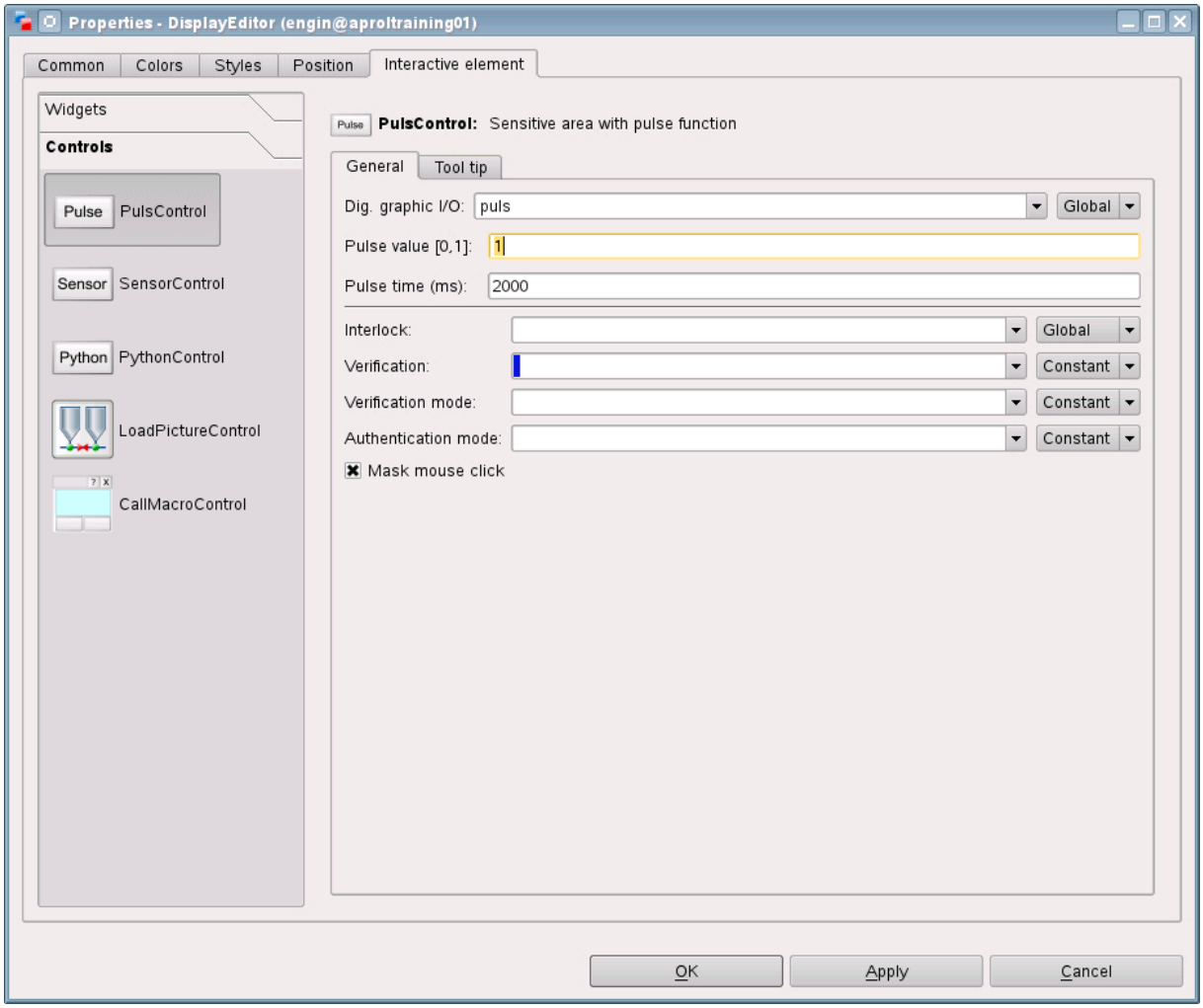

Figure 40: Creating a control

What is a visualization element (widget)?

A visualization element, or widget, is a predefined graphical element whose appearance or form can only be changed using certain properties.

The following visualization elements are available:

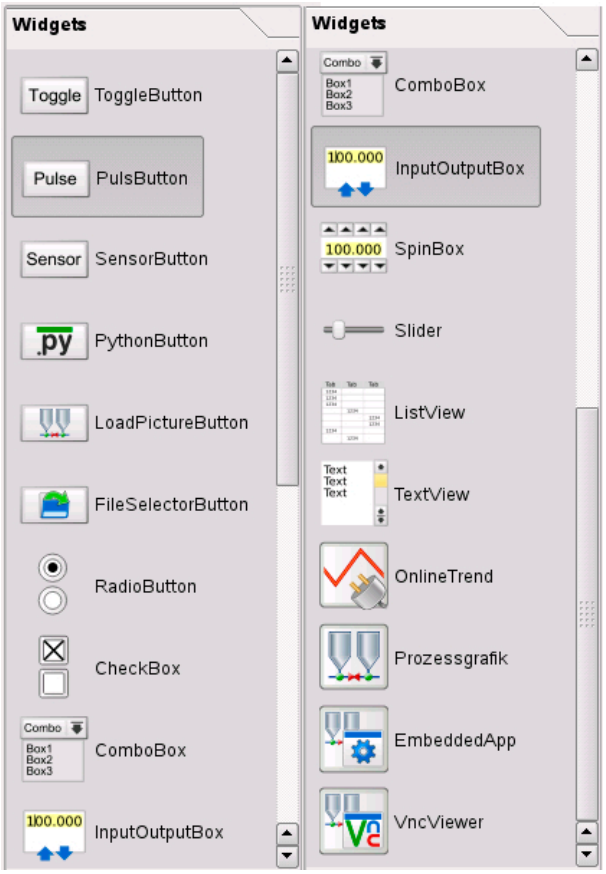

Figure 41: All available widgets at a glance

Widget name	Short widget description
Toggle button	Implements the functionality of a toggle switch. The switch is activated when pressed, and reset the next time it is pressed. The toggle button can be used if an actuator should be turned on or off for an extended period with a single button.
Pulse button	Implements the functionality of a pulse generator. The switching operation is triggered for the duration of the set impulse time when pressed. The pulse button can then be used if only a short pulse is needed to control the actuator.
Sensor button	Implements the functionality of a button. When pressed, the switch is triggered and is only in effect during the time it is pressed. The sensor button can then be used when switching should be controlled by the operator but limited in time.
Python button	The python button behaves like python code and also provides a dynamically labeled key that provides visualization for switching. The python button can then be used to implement complex visualization applications that require explicit logic definitions using python code.
Load picture button	When pressed, the load picture button initiates a change to the process graphic instance, at which time the zoom factor and opening position can be transferred. With the load picture button, it's possible to jump to and zoom process graphics.

Widget name	Short widget description
FileSelectorButton	Opens a file selection dialog and allows the user to select a file or directory. The information is written to an output variable to be able to carry out further actions.
Radio buttons	Implements the functionality of a switch or a group of switches. When the radio button is pressed, switching is triggered and is reset the next time it is pressed or when another radio button in the group is pressed. The radio button can be used if only one option should be selected from a list of n possibilities
Check box	Implements the functionality of a switch or a group of switches. When the check box is selected, switching is triggered and is reset the next time it is selected or when another check box in the group is selected. The checkbox can be used to allow multiple selections.
Combo box	Implements selecting an element from a list. An element is a value/object assignment (tuple). The combo box can be used if only one option should be able to be selected from a list of n possibilities
Input output box	Implements input and/or output of a process variable. The input output box should be used if the actual values used by the system should be displayed in addition to entering a control value.
SpinBox	The input and display of a process variable with the additional functionality of the incremental change. This is used, amongst others, when the specified steps should be incremented or decremented.
Slider	Implements a linear value change to a process variable using a slider. The slider can be used if speed is considered more important that precision when a value changes during interaction.
List view	Implements, similar to a spreadsheet, simultaneous visualization and editing of various process values in a matrix. The process values can be grouped by lines or columns as desired.
Text view	Implements displaying the content of a file or URL. The display is dynamic, i.e. changes to the content are seen immediately. Text view can be used to dynamically display reports, logs or CSV file content in the process graphic.
Online Trend	Implements displaying the values of one or more process variables in the form of a line graph. Value changes are shown continually. The display can be changed using the configuration dialog box in the runtime system.
Process graphic	Implements image in image functionality, i.e. an open process graphic can show another, fully controllable process graphic in any size and in any location.
Embedded app	Embeds another application in the widget as long as the application supports this type of use. The minimum requirement is an "-embed" or "-wid" type of option, so that the application (DisplayCenter) which embeds can tell it in which X window it should embed.
VncViewer	This makes it possible to embed a VNC viewer in the DisplayCenter. For example, a VC (Visual Components) created with AS can be visualized with this. The source may also be another PC or something similar.

These are created just the same as controls.

Detailed information is available in the APROL help and in the APROL Library Engineering (TM860) seminar.

6.2 Graphic Block Outputs

Since controls and widgets were used to create the picture macro, we now need I/O outputs for the picture function block that can be subjected to the rights management system. Rights can be established in the library. This is done – like the project engineering – from the "**Extras / Operator rights"** menu. These rights are then assigned to the picture function block outputs.

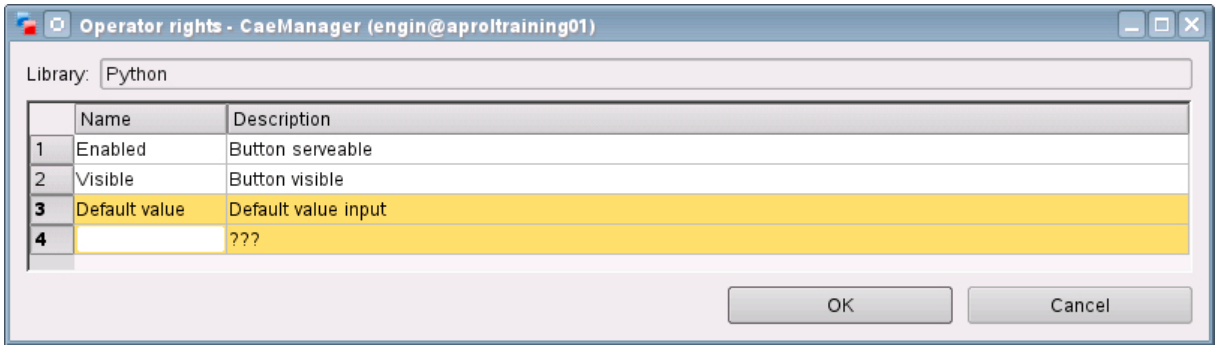

Figure 42: Operator rights in the library

A new block is created with the **"File>>New>>Graphic block"**menu.

When you open up the graphic block, you can configure the **dynamization variables shown in red** in the **I/O view** (input/output, data type, default value, operator rights). The following graphic illustrates this.

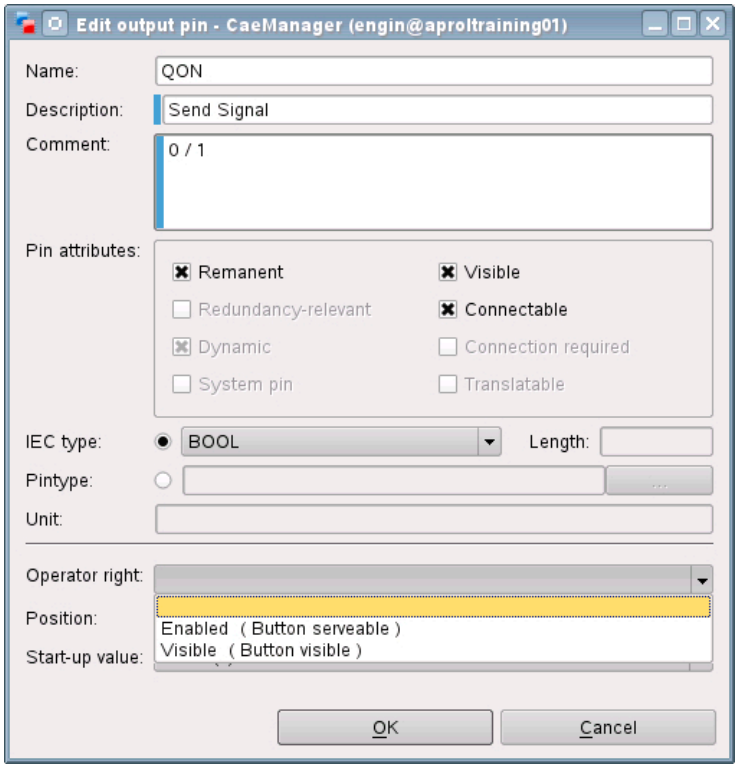

Figure 43: Graphic block editing

The library rights are then assigned as default values to the picture function block outputs. This will help with the engineering from now on. The default rights can be changed during hyper macro engineering and project engineering.

Details can be found in the APROL help documentation at any time.

6.3 Default Library Right Active

This closes the circle of the user management for the runtime and operating environments. The default rights assigned in the libraries are now available in the entire system.

Exercise 5:

Create ON/OFF logic using two buttons and the control "send pulse". Create two library rights and assign them accordingly. Apply these rights to your project profiles and, after a successful build and download, check their functionality in the runtime and operating environments.

7 Summary

What have we achieved so far?

We have reviewed and practiced user management and the necessary rights. Access to operation, alarms, and applications can thus be regulated to fit your needs. The exercises have added depth to your understanding of the processes involved in user management.

This ensures that the system can be correctly and safely operated.

Figure 44: Everything is under control

8 Appendix

8.1 Objectives Checklist (a few On-topic Questions)

Steps involved in user management:

- User rights in the function chart for the hyper macro and the picture function block
- Operator rights in the project
- Operator profile
- OperatorManager and engineering
- Generation
- Operator system download
- Controls in the RUNTIME and operating environments
- Some dynamic elements
- Dynamic elements with operating possibilities (control, widget)
- Operator rights in the library

8.2 Solutions to the Objectives Checklist

Please take note of the necessary additional information.

Seminars and training modules

The Automation Academy provides targeted training courses for our customers as well as our own employees.
At the Automation Academy, you'll develop the skills you need in no time!
Our seminars make it possible for you to improve your knowledge in the field of automation engineering.
Once completed, you will be in a position to implement efficient automation solutions using B&R technology. This will make it possible for you to secure a decisive competitive edge by allowing you and your company to react faster to constantly changing market demands.

Automation Studio seminars and training modules

Programming and configuration	Diagnostics and service
SEM210 – Basics SEM246 – IEC 61131-3 programming language ST* SEM250 – Memory management and data storage SEM410 – Integrated motion control* SEM441 – Motion control: Electronic gears and cams** SEM480 – Hydraulics** SEM1110 – Axis groups and path-controlled movements** SEM510 – Integrated safety technology* SEM540 – Safe motion control*** SEM610 – Integrated visualization*	SEM920 – Diagnostics and service for end users SEM920 – Diagnostics and service with Automation Studio SEM950 – POWERLINK configuration and diagnostics* If you don't happen to find a seminar on our website that suits your needs, keep in mind that we also offer customized seminars that we can set up in coordination with your sales representatives: SEM099 – Individual training day Please visit our website for more information****.****: **www.br-automation.com/academy**

Overview of training modules

TM210 – Working with Automation Studio TM213 – Automation Runtime TM223 – Automation Studio Diagnostics TM230 – Structured Software Development TM240 – Ladder Diagram (LD) TM241 – Function Block Diagram (FBD) TM242 – Sequential Function Chart (SFC) TM246 – Structured Text (ST) TM250 – Memory Management and Data Storage TM400 – Introduction to Motion Control TM410 – Working with Integrated Motion Control TM440 – Motion Control: Basic Functions TM441 – Motion control: Electronic gears and cams TM1110 – Integrated Motion Control (Axis Groups) TM1111 – Integrated Motion Control (Path Controlled Movements) TM450 – Motion Control Concept and Configuration TM460 – Initial Commissioning of Motors TM500 – Introduction to Integrated Safety TM510 – Working with SafeDESIGNER TM540 – Integrated Safe Motion Control	TM600 – Introduction to Visualization TM610 – Working with Integrated Visualization TM630 – Visualization Programming Guide TM640 – Alarm System, Trends and Diagnostics TM670 – Advanced Visual Components TM920 – Diagnostics and service TM923 – Diagnostics and Service with Automation Studio TM950 – POWERLINK Configuration and Diagnostics TM280 – Condition Monitoring for Vibration Measurement TM480 – The Basics of Hydraulics TM481 – Valve-based Hydraulic Drives TM482 – Hydraulic Servo Pump Drives TM490 – Printing Machine Technology In addition to a printed version, our training modules are also available on our website for download as electronic documents (login required): Visit our website for more information: **www.br-automation.com/academy**

Process control seminars and training modules

Process control standard seminars	Process control training modules
SEM841 – Process Control Training: Basic 1 SEM842 – Process Control Training: Basic 2 SEM890 – Advanced Process Control Solutions	TM800 – APROL System Concept TM811 – APROL Runtime System TM812 – APROL Operator Management TM813 – APROL XML Queries and Audit Trail TM830 – APROL Project Engineering TM890 – The Basics of LINUX Visit our website for more information: **www.br-automation.com/academy**

* SEM210 - Basics is a prerequisite for this seminar.
** SEM410 - Integrated motion control is a prerequisite for this seminar.
*** SEM410 - Integrated motion control and SEM510 - Integrated safety technology are prerequisites for this seminar.
**** Our seminars are listed in the Academy\Seminars area of the website.
**** Seminar titles may vary by country. Not all seminars are available in every country.