



**COPADATA**  
do it your way

# zenon manual

## Batch Control

v.7.50





©2016 Ing. Punzenberger COPA-DATA GmbH

All rights reserved.

Distribution and/or reproduction of this document or parts thereof in any form are permitted solely with the written permission of the company COPA-DATA. Technical data is only used for product description and are not guaranteed qualities in the legal sense. Subject to change, technical or otherwise.

# Table of contents

<b>1. Welcome to COPA-DATA help .....</b>	<b>7</b>
<b>2. Batch Control .....</b>	<b>7</b>
<b>3. Introduction .....</b>	<b>9</b>
<b>4. Terminology .....</b>	<b>11</b>
<b>5. Procedure.....</b>	<b>14</b>
<b>6. Function authorizations .....</b>	<b>15</b>
<b>7. Engineering in the Editor.....</b>	<b>15</b>
7.1 Job variables.....	16
7.2 Units .....	17
7.2.1 Toolbar unit tree .....	17
7.2.2 Project tree - Batch Control context menu .....	18
7.2.3 Context menu units .....	19
7.2.4 Context menu unit X.....	20
7.2.5 Detail view units .....	20
7.2.6 Detail view unit X.....	21
7.2.7 Information in Runtime .....	22
7.3 Basic functions .....	24
7.3.1 Context menu phase .....	25
7.3.2 Detail view phase .....	26
7.4 Conditions .....	27
7.4.1 Waiting periods .....	29
7.5 Reactions.....	29
7.5.1 Context menu reactions unit tree .....	31
7.5.2 Detail view reactions .....	31
7.5.3 Events .....	32
7.5.4 Reaction types .....	37
7.6 Parameters.....	39
7.6.1 Detail view tag.....	39

7.6.2	Command TAGs .....	40
7.6.3	Return TAG .....	41
7.6.4	Example for status tag .....	41
7.6.5	Duration of execution .....	43
7.7	Control strategies .....	44
7.7.1	Control strategies node context menu .....	45
7.7.2	Context menu selected control strategy .....	45
7.7.3	Toolbar and control strategy list context menu .....	46
7.7.4	Parameters .....	47
7.8	Keyboards .....	47
7.9	Input lock .....	49
7.10	Create screen of type Batch Control .....	50
7.11	Screen switch Batch Control .....	59
7.11.1	Recipe list settings .....	60
7.11.2	Variable assignment .....	73
7.11.3	Tag list settings .....	91
7.12	zenon functions .....	94
7.12.1	Export Batch recipes .....	95
7.12.2	Import Batch recipes .....	100
7.12.3	Execute recipe command change or mode change .....	102
7.12.4	Create control recipe function .....	107
7.13	Replace links .....	111
<b>8.</b>	<b>Conversion .....</b>	<b>114</b>
<b>9.</b>	<b>Configure and control in the Runtime .....</b>	<b>115</b>
9.1	User interface .....	116
9.1.1	Editor operating elements .....	117
9.2	Commands and actions .....	123
9.2.1	Commands .....	123
9.2.2	Commands and actions .....	127
9.3	Graphical design .....	131
9.4	Engineering rules for recipes .....	132
9.5	Status line .....	134
9.6	Recipe types and recipe states .....	135
9.7	Control strategies .....	136

9.8	Master recipes .....	139
9.8.1	Create master recipe .....	140
9.8.2	Toolbar and context menu for master recipe list view .....	143
9.8.3	PFC recipe .....	146
9.8.4	Matrix recipe .....	182
9.8.5	Master recipe - test mode .....	190
9.8.6	Release master recipe .....	192
9.8.7	Highlight recipe as outdated .....	192
9.8.8	Versioning for master recipes .....	192
9.9	Validate recipe .....	194
9.10	Operations .....	194
9.10.1	Toolbar and context menu operations .....	197
9.10.2	Selection of the template for an operation .....	201
9.10.3	Status operation .....	203
9.10.4	Symbol for execution .....	204
9.11	Control recipe .....	204
9.11.1	Create control recipe .....	205
9.11.2	Toolbar and context menu for control recipe list view .....	206
9.11.3	Control recipe edit mode toolbar .....	207
9.11.4	Execute control recipe .....	209
9.12	Synchronization .....	212
9.13	Manage recipes .....	214
9.13.1	Manage master recipes .....	215
9.13.2	Manage control recipes .....	219
9.13.3	Import recipes .....	221
9.13.4	Saving on the hard disk and backup scenarios .....	222
9.14	Recipe Execution Engine (REE) .....	224
9.14.1	Symbols and Color .....	224
9.14.2	Create recipe image .....	228
9.14.3	Behavior of elements in Runtime .....	229
9.14.4	Mode and mode change .....	231
9.14.5	The execution status .....	231
9.14.6	Step-by-step execution of a recipe and jumps in the recipe .....	233
9.14.7	Process of a phase in detail .....	238
9.14.8	Escape phase .....	243
9.14.9	Restart phase .....	244

9.14.10	Secure writing of the command parameters.....	245
9.14.11	Exit and restart Runtime.....	246
<b>10.</b>	<b>Behavior in the network .....</b>	<b>248</b>
10.1	Redundancy .....	250
<b>11.</b>	<b>Reporting .....</b>	<b>250</b>
11.1	Batch Control recipe filter .....	251
<b>12.</b>	<b>Formula editor .....</b>	<b>255</b>
12.1	Adding parameters .....	260
12.2	List of status bits .....	262
12.3	Logical operators.....	264
12.4	Bit formulas.....	265
12.4.1	Example: ORing bitwise .....	265
12.5	Comparison operators .....	266
12.6	Examples for formulas .....	267
<b>13.</b>	<b>XML export: Units, phases and recipes .....</b>	<b>268</b>
13.1	General recipe properties in the XML file .....	269
13.2	Matrix properties in the XML file .....	272
13.3	PFC properties in the XML file.....	273
13.4	Parameter properties.....	280
<b>14.</b>	<b>CEL.....</b>	<b>289</b>
<b>15.</b>	<b>Failure handling.....</b>	<b>291</b>
15.1	communication errors.....	291
15.2	PLC error .....	293
<b>16.</b>	<b>Error Handling .....</b>	<b>294</b>

# 1. Welcome to COPA-DATA help

## GENERAL HELP

If you cannot find any information you require in this help chapter or can think of anything that you would like added, please send an email to [documentation@copadata.com](mailto:documentation@copadata.com) (<mailto:documentation@copadata.com>).

## PROJECT SUPPORT

You can receive support for any real project you may have from our Support Team, who you can contact via email at [support@copadata.com](mailto:support@copadata.com) (<mailto:support@copadata.com>).

## LICENSES AND MODULES

If you find that you need other modules or licenses, our staff will be happy to help you. Email [sales@copadata.com](mailto:sales@copadata.com) (<mailto:sales@copadata.com>).

# 2. Batch Control

The module Batch Control offers the possibility to automate batch-orientated manufacturing processes for lot products. The module complies to ANSI/ISA-88.01-1995 also known as ANSI/ISA-S88.

For Batch Control there are two different editors available in the Runtime.

- ▶ Matrix editor: (on page 183) For simple, sequentially or parallel running recipes.
- ▶ PFC editor: (on page 147) For complex recipes with branches.

Depending on the license, either both editors or only the matrix editor is available to you.



### License information

*Must be licensed for Editor and Runtime (single-user, Server, Standby and Client).  
Licensing distinguishes after use of:*

- ▶ Matrix editor
- ▶ Matrix Editor and PFC Editor

Editor:

*For the engineering in the zenon Editor one of the Batch editors must be licensed.*

Runtime:

*For execution in Runtime, a Batch Editor must be licensed in order to fill the list of recipes. Creating and editing recipes is only possible for the licensed editor.*

Network:

*In the network the license of the Server counts. Stand-alone licenses of the Clients are ignored in network operation.*



### Attention

*Notes in the help embedded in the Editor on VBA keywords are not functional in the current version.*

## NOTE FOR CHANGE FROM 7.00 TO 7.10 OR HIGHER

Before converting a project to a new zenon version all recipes must be completed. Recipes that are running continue to be executed after a restart. The restart only functions within the same zenon version.

Attention: Projects with recipes that were created in zenon 7.10 or higher cannot run in zenon 7.0.

## EFFECTS OUTSIDE BATCH CONTROL

When using different versions for Editor and Runtime, problems can occur if the Batch Control module is licensed but Batch Control is not used.

**Background:** Some files are compiled in Runtime as soon as the module is licensed. A batch project that was compiled with 7.0 cannot be executed in 7.10.

**Solutions:**

- ▶ Compile the project with an Editor of version 7.10 or higher.
- ▶ Use a Runtime of version 7.0.
- ▶ Use a license that does not include the Batch Control module.



## BATCH CONTROL AND COMMAND SEQUENCER MODULE

If both the Batch Control (on page 7) module and the Command Sequencer module, which both require a license, are licensed at the same time, selection of the module used is carried out by means of the project setting.

To select the preferred module in Runtime:

- ▶ Click on the node of your project in the Editor.
- ▶ Go to the **Runtime settings** project properties group.
- ▶ Select, for **License module in Runtime** of the **Preferred module** property, `command sequence`(default) or `Batch Control`.

The selected model is then available in Runtime for further project configuration.

## 3. Introduction

The module Batch Control consists of three parts:

1. The engineering environment (on page 15) in the zenon Editor:  
There all units (ISA nomenclature, chapter 4.2.5: units) with their phases (ISA - 5.1.2.4: phases) and reactions are created. The phases must have an equivalent in the control (ISA: equipment control) which is called process action according to ISA.  
Batch Control reflects the physical model in accordance with ISA 4.2 as flat hierarchical level based on units.  
The other levels of the model such as process cell, area, plant, etc. were deliberately forgone. When creating the batch recipes too, only the lowest level (phases) of the ISA structure model 5.1 and operations were implemented. Additional levels such as unit procedures and procedures are not available.
2. PFC editor (on page 147) and Matrix editor (on page 183):  
With the help of these editors, master recipes in zenon Runtime (ISA: master recipes) are created. The control recipes that can be executed (ISA: control recipes) are derived from these (see also ISA 5.3.1.). During the process the exact status of the Batch recipe is displayed in the respective editor and you can interfere in the recipe process.
3. Recipe Execution Engine (on page 224) (REE):  
The REE is directly integrated in the zenon Runtime and executes a Batch recipe automatically in the background. Via commands such as **Start**, **Pause**, **Stop** etc. the user can control the REE. There are three possible modes: **Automatic**, **Semi-automatic** and **Manual**.

### SPECIAL FEATURES OF MODULE BATCH CONTROL:

In contrast to most other zenon modules, a large part of the engineering - the recipe creation - is done in the Runtime and not in the Editor. This entails special features which are dealt with in the respective

chapter. So for example changed phases are no longer transferred to an already released master recipe in order to prevent unwanted data changes.

The module is designed in a way which makes it completely independent of the control. This means that the data communication take place via all available zenon drivers with any PLCs or even RTUs. They only execute the process actions. The entire recipe processing is done at the computer in the REE. For changes on the Batch recipe or for new master recipes, no modifications are necessary in the PLC code.

The module follows the strict separation between the procedure of the batch recipe (ISA: Procedural Control Model) and execution of the technological function (ISA: Process Model) as describe in ISA-S88, chapter 5.2.1.

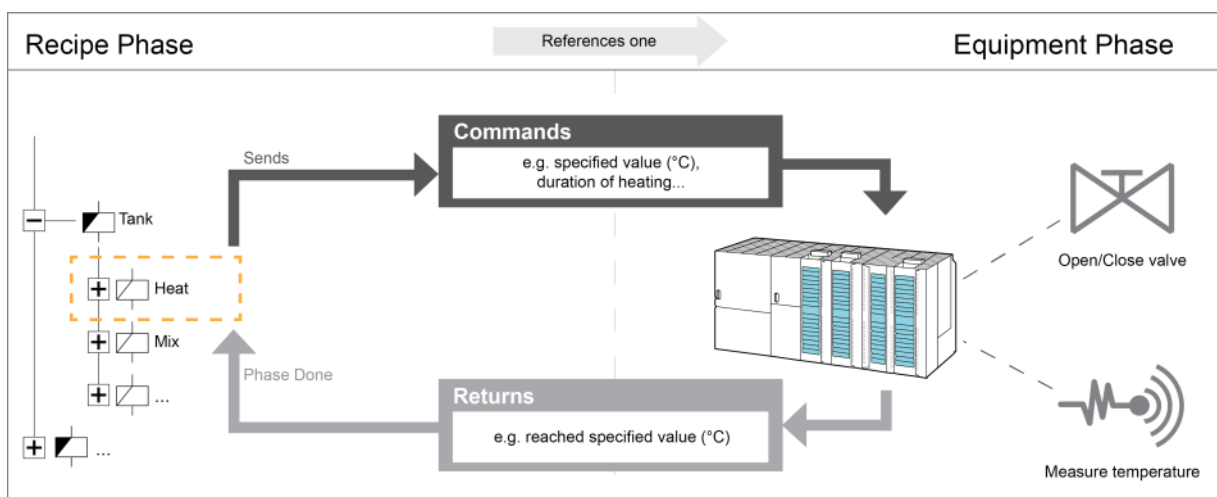
## PRINCIPLE STRUCTURE OF THE COMMUNICATION

For the communication with the process standard zenon variables are used. As variable names often have cryptic label, an additional abstraction level was introduced. It contains the tags which are available in two types:

- ▶ Command tags: They are used to transfer set values to the PLC when a phase is executed.
- ▶ Return tags: They are used to return values from the PLC for evaluation.

With both tag types values can be both written (e.g. in reactions) and also read (e.g. in conditions).

Schema:



The PLC communicates with the zenon driver. The driver communicates with the zenon Runtime. The Runtime sends the values to the REE where they are processed. The REE internally works asynchronously to the zenon Runtime in a 100 ms cycle.

## NETWORK

The module **Batch Control** is fully capable of using a network in terms of Client/Server technology. This means that Batch recipes can be created, duplicated, edited, deleted, etc. on a Client. The whole recipe management remains always on the server. Likewise the whole process control such as **start** recipe, **pause** recipe, **stop** recipe, etc. can be done from the Client. Also mode changes and manual operations such as **jump** are possible.



### Attention

*Module Batch Control does not support redundancy. There is no synchronization between Standby Server. When the Server breaks down, the executed Batch recipes are not continued seamlessly on the Standby!*

## 4. Terminology

In the zenon module Batch Control the following terms are used:

Term	Definition
<b>Unit</b>	Physically available machine or equipment part with which phases can be carried out. (ISA 88 Unit)
<b>Releasing the unit</b>	Element of module Batch Control which cancels the allocation of a unit in the unit manager. With this the unit can be allocated by another recipe again.
<b>Unit allocation</b>	Element of module Batch Control which causes the allocation of a unit in the unit manager. An allocated unit can only be used by phases with the recipe. With this the unit is locked for phases of other recipes which are executed parallel.
<b>Unit manager</b>	Internal management mechanism which manages the unit allocation for all REE's in the Runtime.
<b>Action</b>	Used in Batch Control: all commands which are used for editing a recipe e.g. insert phase, testing recipes etc.
<b>Begin parallel branch</b>	Element that ensures the separation of the recipe process in two or more sequence selections.
<b>Begin branch</b>	Element that makes it possible to separate a recipe in two or more sequence selections of which only one can be active at a time. Each following sequence selection must start with a transition. The transition defines which sequence selection is executed in the recipe process.
<b>Begin element</b>	Element of module Batch Control with which every recipe begins.
<b>Active element</b>	Position in a recipe in the batch control module where the processing is interrupted in a semi-automatic and manual mode and the active elements are put into a pause status. With the "next step" command, the process is resumed from this position.
<b>Batch Control</b>	Tool for creating master recipes and creating and executing control recipes in accordance with ISA-S88.
<b>Batch operation</b>	Automatic and sequential processing of a stack of single operations.
<b>End parallel branch</b>	Element that combines the separation of the recipe process into two or more sequence selections back into one sequence selection.

<b>End branch</b>	Element which brings together a sequence selection started by a begin sequence selection element.
<b>End element</b>	Element of module Batch Control with which every recipe ends.
<b>Phase</b>	Predefined process consisting of input interlocking, command and return tags, a phase done condition, event reactions, etc... (ISA 88: Phase)
<b>Command</b>	Used in Batch Control: a command which intervenes in the recipe process e.g start, stop, mode change etc.
<b>Matrix recipe</b>	Recipe in the Batch Control module which was created with the Matrix editor.
<b>Parallel branch</b>	Area of module Batch Control. A simultaneous sequence starts with a begin simultaneous sequence element and is brought together with an end simultaneous sequence element to one execution branch. Between there are at least two sequence selections which are executed at the same time.
<b>PFC recipe</b>	Recipe in the batch control module, which was created with the PFC Editor
<b>REE - Recipe Execution Engine</b>	Part of module Batch Control for process control of recipes. The engine executes a control recipe and manages the entire process of the recipe.
<b>Recipe</b>	In recipes related data such as machine parameters or format data are summed up. This data can be transferred from the control system to the control and vice versa in one step. We differentiate between standard recipes and RGM recipes. The procedure is defined additionally to data in Batch Control Module in a recipe. It is distinguished between Matrix recipe and PFC Recipe.
<b>Jump target</b>	Element of the batch control module which allows a direct jump to a defined location of a sequence selection.
<b>Control recipe</b>	Part of the batch control module. Contains the process of a production process on basis of the batch process according to standard ISA S88. A control recipe is always derived from a template recipe and can be implemented once only. (ISA 88: Control Recipe)
<b>Operation</b>	Recipes can be divided into individual parts within the batch control module. Operation management takes place via a central library. Instances of operations can be added within the recipe. Tags of the applied phase can be edited, the structure can only be edited in the operation template.
<b>Transition</b>	Element of module Batch Control which contains a condition. The element is used after phases in order to ensure a defined transition from one phase to another.

<b>Connection line</b>	Part of the connector in the Batch Control module: Positions the connection point at the element.
<b>Connection point</b>	Part of the connection element in the Batch Control module. Connects two elements to each other (e.g. phase to phase or phase to line). It changes color when the mouse pointer is on it.
<b>Connection element</b>	A possibility in Batch Control to connect elements to one another. It consists of a connection point and a connection line.
<b>Branch</b>	Area of module Batch Control which ensures a separation in two or more sequence selections, of which only one can be active at the recipe process. It is an either/or sequence selection. A sequence selection always starts with a begin sequence selection element and ends with an end sequence selection element.
<b>Master recipe</b>	Part of the batch control module. Contains the process of a production process on basis of the batch process according to standard ISA S88. A recipe consists of the following components: basic functions, transitions, parallel circuits etc (ISA 88: Master recipe). Template recipes serve as templates for control recipes.
<b>Branch</b>	Execution area for the Batch Control Module Basic functions, transitions and transfer targets can be placed on it.

## 5. Procedure

The engineering and the use of module Batch Control takes place in three main steps:

1. Engineering in the zenon Editor
2. Recipe creation in the Batch editor (PFC or matrix) in the Runtime
3. Recipe execution in the Runtime

### ENGINEERING

The engineer depicts the existing physical world in the Editor. He defines and creates units (on page 17) and assigns phases (on page 24) with tags (on page 39) to them.

### RECIPE CREATION

In the Runtime the recipe creator defines the master recipes on base of the presets from the Editor. They define the process. At this only the units, phases and tags defined in the engineering environment can be used.

**Exception:** If you activate property **Changeable in master recipe** in group **Write set value** in the Editor, you can modify the corresponding tag. This is only possible for command tags.

## RECIPE EXECUTION

The operator executes the recipe in the Runtime. For this he starts a control recipe which is based on the control recipe. Each control recipe can only be started and used once. With this it can be assigned to a lot unambiguously. Recipes run either automatically, semi-automatically or manually.

The operator cannot influence the recipes.

**Exception:** If the recipe creator activates option (on page 163) **Changeable in the control recipe**, the operator can edit tags in the Runtime.

## 6. Function authorizations

For changes in the Editor and in Runtime, the corresponding function authorizations can be issued to users. A warning is displayed if operations are executed for which there are no corresponding rights. **Exception:** No warning is displayed when editing.

Users can also log in temporarily for the execution of operations for which they have no special authorizations. For details on this, see the **Permanent and temporary login** section in the User administration manual.

You can find details on the individual authorizations in the User administration chapter, most of all in the **Function authorizations** section.

**Note:**

- ▶ These function authorizations are only available if Batch Control is licensed on the computer.
- ▶ Operation instances always use the user rights of the recipe in which they are embedded.

## 7. Engineering in the Editor

To use module Batch Control in the Runtime, you must do the following in the Editor:

- ▶ create units (on page 17), phases (on page 24) and reactions (on page 29)
- ▶ create a screen (on page 50) of type `Batch Control`
- ▶ create a screen switch function (on page 59) for the screen of type `Batch Control`

**Note:** In the editor, copying and inserting of elements throughout all levels is possible.

## DETAIL VIEW

In module Batch Control the detail view divided in two:

- ▶ The left part features the unit tree. The entry **Unit** is the starting point of the tree. On the next level the existing units are displayed. The phase belonging to each unit follow. Each phase has the subitem reactions.
- ▶ In the right part a flat list of the units, phases, tags or reactions is displayed depending on what level is chosen on the left side.

## 7.1 Job variables

Job variables can be linked in the Editor. Job variables make it possible to assign job IDs to Batch productions. In order to ensure that the variable value is available immediately, the job variable is inserted into the global connection and registered when Runtime is started. The variable is requested again on reloading.

Job variables defined in the **Individual job variable** property can be allocated using a function (on page 94). Otherwise the global **Job variable** is used.

### GLOBAL JOB VARIABLES

To link a global **Job variable**:

1. navigate to the **General/Job variable** node in the module
  2. Link a variable in the **Job variable** property
  3. Select, from the drop-down list of the **Apply value from job variable** property, the type of transfer to Runtime:
    - When creating the control recipe:

When creating the control recipe, the current value of the variables is transferred into the control recipe. The value is displayed in the control recipe configuration dialog. The value must not be empty, otherwise the recipe cannot be created.
    - When starting the control recipe:

In Runtime, the content of the variables is written to the recipe when the control recipe is started. The value must not be empty.
- Note about value changes:** The value transferred by the variables is always changed into a string, regardless of the type of variables. When changing from real to string, 10 decimal points are taken into account. Zeros at the end are cut off.
4. You can also define a **Individual job variable** if you wish



In Runtime, the allocated job ID can be displayed in the list of control recipes.

## INDIVIDUAL JOB VARIABLE

To link a **Individual job variable**:

1. navigate to the **General/Job variable** node in the module
2. Link one or more variables in the **Individual job variable** property
3. When configuring zenon functions (on page 94), select the desired variable and the type of allocation

## 7.2 Units

To create a new unit:

1. in the project manager go to node `Batch Control`
2. in the detail view select `Unit`
3. in the context menu select menu item **New unit**
4. a new unit is created in the detail view

### 7.2.1 Toolbar unit tree

The toolbar corresponds to the complete unit tree. Depending in the selected element symbols are available or deactivated.



Parameters	Description
<b>Rename</b>	Opens the name field of the selected element for renaming. Not available for main node <b>Units</b> .
<b>Copy</b>	Copies the selected unit to the clipboard. Not available for main node <b>Units</b> .
<b>Paste</b>	Pastes a unit that was copied to the clipboard.
<b>Delete</b>	Deletes selected element after confirmation message. Not available for main node <b>Units</b> .
<b>Expand all</b>	Displays the entire tree structure.  By clicking on the arrow you receive a drop-down list in which you can select one of the following commands: <ul style="list-style-type: none"> <li>▶ Expand all: expands all nodes</li> <li>▶ Collapse all: collapses all nodes</li> <li>▶ Expand selected: expands all selected nodes</li> <li>▶ Collapse selected: collapses all selected nodes</li> </ul> A click on the button always expands all elements.  Via double click on the superordinate entry elements can also be expanded or collapsed.
<b>Export all units to XML...</b>	Exports all units as an XML file.
<b>Import units from XML...</b>	Imports units from an XML file.
<b>Help</b>	Opens online help.

### 7.2.2 Project tree - Batch Control context menu

Parameters	Description
<b>New unit</b>	Creates a new unit.
<b>Export XML all...</b>	Exports all units as an XML file.
<b>Import XML...</b>	Imports units from an XML file.
<b>Editor profile</b>	Opens the drop-down list with predefined editor profiles.
<b>Help</b>	Opens online help.

### 7.2.3 Context menu units

Right click the main entry **Unit** in the unit tree to open a context menu:

Menu item	Action
<b>New unit</b>	Creates a new unit.
<b>Replace linking in phases...</b>	Opens the dialog to replace linking (on page 111) for linking in phases.
<b>Replace linking in units...</b>	Opens the dialog to replace linking (on page 111) for linking in units.
<b>Paste</b>	Pastes a unit that was copied to the clipboard.
<b>Export all units to XML...</b>	Exports all units as an XML file.
<b>Import units from XML...</b>	Imports units from an XML file.
<b>Help</b>	Opens online help.

## 7.2.4 Context menu unit X

Right click on a created unit in order to open the context menu:

Menu item	Action
<b>New phase</b>	Creates a new phase.
<b>Replace linking in phases...</b>	Opens the dialog to replace linking (on page 111) for linking in phases.
<b>Replace linking in unit...</b>	Opens the dialog to replace linking (on page 111) for linking in the selected unit.
<b>Rename</b>	Allows you to rename the selected unit.
<b>Delete</b>	Deletes the selected unit.
<b>Copy</b>	Copies the selected unit to the clipboard.
<b>Paste</b>	Pastes a unit that was copied to the clipboard.
<b>Export selected XML...</b>	Exports selected unit as an XML file.
<b>Import phases from XML...</b>	Imports phases from an XML file.
<b>Help</b>	Opens online help.

## 7.2.5 Detail view units



Menu item	Action
<b>New unit</b>	Creates a new unit in the detail view.
<b>Replace linking in phase</b>	Opens the dialog to replace linking (on page 111) for linking in phases.
<b>Replace linking in units</b>	Opens the dialog to replace linking (on page 111) for linking in units.
<b>Copy</b>	Copies the selected entries to the clipboard.
<b>Paste</b>	Pastes the contents of the clipboard. If an entry with the same name already exists, the content is pasted as " <b>Copy of...</b> ".
<b>Delete</b>	Deletes selected entries after a confirmation from list.
<b>Remove all filters</b>	Removes all filter settings.
<b>Edit selected cell</b>	Opens the selected cell for editing. The binocular symbol in the header shows which cell has been selected in a highlighted line. Only cells that can be edited can be selected.
<b>Replace text in selected column...</b>	Opens the dialog for searching and replacing texts.
<b>Export selected XML...</b>	Exports the selected units as an XML file.
<b>Import units from XML...</b>	Imports units from an XML file.
<b>Properties</b>	Opens the <b>Properties</b> window for the selected entry.
<b>Help</b>	Opens online help.

### 7.2.6 Detail view unit X



Parameters	Description
<b>New phase</b>	Creates a new phase in the detail view.
<b>Replace links</b>	Opens the dialog to replace linking (on page 111) for linking in phases.
<b>Copy</b>	Copies the selected entries to the clipboard.
<b>Paste</b>	Pastes the contents of the clipboard. If an entry with the same name already exists, the content is pasted as " <b>Copy of...</b> ".
<b>Delete</b>	Deletes selected entries after a confirmation from list.
<b>Remove all filters</b>	Removes all filter settings.
<b>Edit selected cell</b>	Opens the selected cell for editing. The binocular symbol in the header shows which cell has been selected in a highlighted line. Only cells that can be edited can be selected.
<b>Replace text in selected cell...</b>	Opens the dialog for searching and replacing texts.
<b>Export selected XML...</b>	Exports the selected phases as an XML file.
<b>Import phases from XML...</b>	Imports phases from an XML file.
<b>Properties</b>	Opens the <b>Properties</b> window for the selected entry.
<b>Help</b>	Opens online help.

## 7.2.7 Information in Runtime

Information on individual units can be called up and displayed using variables. Each element to be allocated initializes the values of the linked variables with 0 or empty text. The information comes from the recipe that allocates the unit when the query takes place. Variables are always only filled in the event of a change.

The variables for the information desired in Runtime is configured in the unit properties in the **Runtime information** group for:

- ▶ **Master recipe:** Information on ID, name and description of the master recipe, as well as version (on page 192) and initial version.
- ▶ **Control recipe:** Information on ID, name, description and job ID of the control recipe.
- ▶ **Execution:** Information on the number of active recipes and execution status and mode, numerically and as a text respectively.

**Note:** The value for **Number of active recipes** is generally 0 or 1. If a higher number is displayed, then the start of other recipes was forced manually.

- ▶ **Phases:** Information on active phases. If the phase is in an operation, the name of the operation is displayed in brackets.

- **Error:** Errors are shown visually in Runtime and saved in logs. A process error in the unit is shown visually this way and the absolute number of historical procedure errors is also displayed. The counter is increased by 1 as soon as a procedure error occurs. If the phase is restarted, the historic error goes from the display; it is no longer signaled visually. The logged information is retained however. The same applies to communication failures: Symbols only display active errors; counters also inform you of historic errors.
- **Matrix information:** Display of the active steps in a matrix recipe. It is always the information from the main recipe that is used, even if the object to be triggered is in an operation. All numerical variables whose data type  $\geq 2$  byte can be selected. PFC recipes always receive empty values or 0.

## EXECUTION STATUS AND EXECUTION MODE OF VALUES

Values for variables in the **Execution status (numeric)** property:

Status	Return	Remark
Idle	0	
Running	1	
Executed	2	Is never displayed, because a completed recipe does not allocate the unit.
Stopping	3	
Stopped	4	Is never displayed, because a stopped recipe does not allocate the unit.
Pausing	5	
Paused	6	
Holding	7	
Held	8	
Aborting	9	
Aborted	10	Is never displayed, because an aborted recipe does not allocate the unit.
Restarting	11	

Values for variables in the **Execution mode (numeric)** property:

Mode	Return	Remark
Ignore	0	No recipe runs
Automatic	1	
Semi-automatic	2	
Manual	3	

## 7.3 Basic functions

The phase is the execution object of a recipe - and therefore its main component. Each phase in module Batch Control must stand facing a **Technological function** in the control.

**Example:** You want to heat up a tank. For this you need:

1. The corresponding equipment: a heating in the tank.
2. A temperature sensor which measures the actual temperature in the tank. Connect this sensor with a control.
3. In the control a program which controls the heating until the set temperature is reached. This control program is the **process action** in the PLC. It:
  - has an input tag: a set temperature - which is implemented via a command tag (on page 40) in Batch Control
  - needs an output tag - also a set temperature - which must be reached and which is implemented via a return tag (on page 41) in Batch Control

To inform the control about the progress of the recipe, you need corresponding status information which is transferred to the control. For this you use reactions (on page 29) and conditions (on page 27) for the response.

A phase therefore consists of:



Type	Description
Command TAGs (on page 40)	The set values which should be transferred to the control
Return TAG (on page 41)	The response values which inform the REE about the status of the <b>process action</b> in the control. They can be evaluated in conditions and transitions.
Reactions (on page 29)	REE events can be used on the one hand to inform the <b>process action</b> in the control about the state of the REE and on the other hand to inform the user about errors (e.g. time outs, invalid tag values). time outs, invalid tag values).
Conditions (on page 27)	Are used for the evaluation of the return tags: The state of the <b>process action</b> in the control is evaluated.
Times (on page 29)	Time critical processes can be monitored with this. If the engineered time is exceeded, an event is triggered on which you can react with a reaction.

## ENGINEERING

To create a new phase:

1. select the unit for the phase or first create the desired unit
2. in the context menu select menu item **New phase**
3. a new phase is shown in the detail view
4. the subitem **Reactions** is automatically added to the phase



### Information

*If several phases are selected in the Editor at once whose formulas are identical but whose tags are different, this is not displayed by color coding the different values.*

### 7.3.1 Context menu phase

#### CONTEXT MENU PHASE X

Right click on a created phase in order to open a context menu for creating the parameters:

Parameters	Description
<b>New initial tag</b>	Creates a new initial tag (on page 40).
<b>New value tag</b>	Creates a new value tag (on page 40).
<b>New return tag</b>	Creates a new return tag (on page 41).
<b>Replace linking in phase</b>	Opens the dialog to replace linking (on page 111) for linking in phases.
<b>Rename</b>	Makes it possible to change the name of the currently selected phase.
<b>Delete</b>	Deletes the currently selected phase after a confirmation message.
<b>Copy</b>	Copies the selected element to the clipboard.
<b>Paste</b>	Pastes an element script that was copied to the clipboard.
<b>Export selected XML</b>	Exports the selected phases as an XML file.
<b>Help</b>	Opens online help.

### 7.3.2 Detail view phase



Parameters	Description
<b>New initial tag</b>	Creates a new initial tag (on page 40).
<b>New value tag</b>	Creates a new value tag (on page 40).
<b>New return tag</b>	Creates a new return tag in the detail view.
<b>Replace links</b>	Opens the dialog to replace links (on page 111).
<b>Copy</b>	Copies the selected entries to the clipboard.
<b>Paste</b>	Pastes the contents of the clipboard. If an entry with the same name already exists, the content is pasted as " <b>Copy of...</b> ".
<b>Delete</b>	Deletes selected entries after a confirmation from list.
<b>Remove all filters</b>	Removes all filter settings.
<b>Edit selected cell</b>	Opens the selected cell for editing. The binocular symbol in the header shows which cell has been selected in a highlighted line. Only cells that can be edited can be selected.
<b>Replace text in selected column...</b>	Opens the dialog for searching and replacing texts.
<b>Properties</b>	Opens the <b>Properties</b> window for the selected entry.
<b>Help</b>	Opens online help.

## 7.4 Conditions

The conditions are used to inform the REE of the status of the **technological function** in the controller. For evaluating the conditions formulas are used which were created with the formula editor (on page 255).

Hint: Use a single status tag (return tag) which takes on different values in order to transmit the status of the **process action** in the control to the phase. You can find an example in chapter Example for status parameter (on page 41).

### ENGINEERING

To create a new condition:

1. click on the corresponding phase
2. in the properties select the desired conditions from node **General** or **Condition transient status**
3. click in the field for the value or on button ...
4. the formula editor is started

5. define the formula (on page 255) for the condition

**Note:** The counterpart to the conditions are the reactions (on page 29). With them, the execution status is transferred to the technological functions of the controller.

## TRANSIENT STATES

As transitions the following properties are available

► **Paused**

Within the phase the process stops at:

- Waiting for `Finished`
- Waiting for allocation
- Waiting for interlocking
- Waiting for `Phase finished`
- Check for parallel execution

► **Held**

Within the phase the process stops at:

- Waiting for `Finished`
- Waiting for allocation
- Waiting for interlocking
- Waiting for `Phase finished`
- Check for parallel execution

► **Stopped**

► **Aborted**

► **Restarted**

Phase is completely restarted.

► **Escape condition**

If this condition is met, the current execution step is stopped and the phase is exited. You can find details in the Exiting a phase (on page 243) chapter.

## VARIABLES

Transition conditions can be linked with a binary variable which defines when the phase changes its status. The status changes is delayed until:

- the value of the variable is defined
- the value of the variable is 1
- the status of the variable is not invalid.

If no variable is defined, the status change is always allowed.

All variables for the status change are requested at the advising of the variables in order to receive a value as soon as possible. The values for a variable which define a status change are read on the change to the transient status. A possible pulse must have value `TRUE` within the waiting period in order to be recognized.

**Note:** When closing the Runtime, it is not waited for the variable for the status change from `stopping` to `stopped` as at this time all variables are already signed off.

### 7.4.1 Waiting periods

The recipe creator can define waiting periods. The configuration of waiting periods (time outs) prevents that time-critical processes take too long because of unforeseen events. If the condition is not fulfilled within the defined waiting period, a corresponding event (on page 32) is triggered. With the reactions (on page 37) you can react on the event and influence the recipe process.

For all waiting periods the following is true:

- ▶ If `0d 00:00:00` is defined as waiting period, the event is not triggered.
- ▶ The waiting periods are independent of the recipe status (e.g. `Recipe paused`) and continue to run even when the Runtime is closed and restarted.
- ▶ If a phase is held and restarted, the waiting periods are also restarted.
- ▶ If a phase is passed through several times, the waiting periods are started again for every pass.
- ▶ Waiting periods themselves do not influence the process. They are simply used to generate an event. The reaction must be defined in the event. After the event is triggered, it is still waited for the fulfillment of the condition.

## 7.5 Reactions

Reactions are the most important piece to influence the recipe process and to communicate with the control. Reactions are always based on events. These can be REE events (e.g.: `Phase started`), as well as general events (e.g. `Exit Runtime initiated`). With the help of reactions you can e.g. tell the control when a phase has been started or finished in the REE and when all command tags have been written.

Likewise you must transfer the status of the phase to the control with the help of the reactions. Otherwise the control has no information about the process of the recipe.

**Example:** If you stop the REE or the phase, the event `Status change: Stop` will be triggered. As reaction you can transfer this status change as set value input to the control. Only then can the control

react and stop the **process action**. You can find an example in chapter Example for status parameter (on page 41).

**Note:** The counterpart to the reactions are the conditions (on page 27). With them the states of the process action in the control is transferred to the REE.

## ENGINEERING

With each phase the node **Reactions** is created automatically. In this node you can create any reactions. To create a new reaction:

1. click on **reactions**
2. in the context menu select menu item **New reaction**
3. a new reaction is shown in the detail view
4. in the detail view click on the entry in column **Event**
5. select the desired event from the drop-down list and define the desired reaction type (on page 37) in the property window.

For each reaction type several reactions are possible. They are sorted at the triggering and are executed in accordance to their priority. At this 1 is the highest priority. Reactions of the same type can only be re-sorted using the toolbar or context menu (on page 31).

Some of the reactions are triggered only once in the process - e.g. `time outs`. If the phase is restarted, these reactions are also retriggered if necessary.

Reactions can only use tags of their own phase. If reactions are copied from other phases, they try to use tags with the same name of the name phase.

Values of the reactions are logged in the CEL.

## PROJECT CONFIGURATION RULES

- ▶ Reactions can appear in each object state.
- ▶ For each reaction type several reactions are possible. They are sorted at the triggering and are executed in accordance to their priority. At this 1 is the highest priority.
- ▶ All variables of all parameters are signed in to the driver for reading. If a value is needed at a reaction but is not yet available or invalid, the alternate value is written. The writing of the value is done without write confirmation.
- ▶ Some of the reactions are triggered only once in the process - e.g. `time outs`. If the phase is restarted, these reactions are also retriggered if necessary.

## RUNTIME: USERS

If zenon functions are executed by a reaction, the reaction is reported as the user `System`.

### 7.5.1 Context menu reactions unit tree

Parameters	Description
<b>New reaction</b>	Creates a new reaction in the detail view.
Replace links	Opens the dialog to replace links (on page 111).
Paste	Pastes the contents of the clipboard. If an entry with the same name already exists, the content is pasted as " <b>Copy of...</b> ".
<b>Help</b>	Opens online help.

### 7.5.2 Detail view reactions



Parameters	Description
<b>New reaction</b>	Creates a new reaction in the detail view.
<b>Execution order: Earlier</b>	For reactions of the same type: Moves the reaction forward in the execution order.
<b>Execution order: Later</b>	For reactions of the same type: Moves the reaction backward in the execution order.
<b>Execution order: Change places</b>	Only active if exactly two reactions are chosen. The two selected reactions change their places in the execution order.
<b>Replace links</b>	Opens the dialog to replace links (on page 111).
<b>Copy</b>	Copies the selected entries to the clipboard.
<b>Paste</b>	Pastes the contents of the clipboard. If an entry with the same name already exists, the content is pasted as " <b>Copy of...</b> ".
<b>Delete</b>	Deletes selected entries after a confirmation from list.
<b>Remove all filters</b>	Removes all filter settings.
<b>Edit selected cell</b>	Opens the selected cell for editing. The binocular symbol in the header shows which cell has been selected in a highlighted line. Only cells that can be edited can be selected.
<b>Replace text in selected column</b>	Opens the dialog for searching and replacing texts.
<b>Properties</b>	Opens the <b>Properties</b> window for the selected entry.
<b>Help</b>	Opens online help.

### 7.5.3 Events

Each `reaction` is a reaction to an event. The event is defined in property **Event**. For each event several reactions can be defined. The execution order can be defined in the detail view.

When validating the recipe in Runtime, the name of the event is displayed in the event of an error.

Syntax: **(event name.x)** whereby **event name** corresponds to the **Event**. **x** is a number that indicates the position in the execution sequence.

From the drop-down list you can select the following events:



Event	Description
<b>Procedure</b>	Events in the procedure.
Phase activated	<p>Is the first event which is triggered.</p> <p>With this event, you tell the PLC that the phase has been activated in the REE and that the phase is expected to be started soon.</p>
Unit allocation not possible	Is triggered if the unit was not allocated successfully at first try.
Phase started	<p>With this event you tell the PLC that the phase has been started in the REE and that it is likely that the command tags will be written soon.</p> <p>Other events can be executed before the event if the <b>Allow execution before start event</b> has been activated for the corresponding event. For details, see the <b>Allow execution started before phase started</b> section.</p>
Input interlocking blocked	<p>Is triggered if the input lock has been successfully checked.</p> <p>Makes only sense if property <b>Interlocking condition</b> was configured.</p>
Input lock checked successfully	<p>Is triggered if the input interlocking was blocked at the first check.</p> <p>Makes only sense if property <b>Interlocking condition</b> was configured.</p>
Finished writing value tags	<p>Is triggered if all command tags have been written. It cannot be guaranteed however that really all tags arrived at the control. It depends on the communication and the respective driver. It can however be assumed.</p> <p><b>Recommendation:</b> Use this event to tell the PLC that the phase has written all command tags and the PLC can start processing the process actions.</p>
Phase done condition completed	<p>Is triggered if the phase is finished. This event is the last reaction of the phase and independent of the reason of the finishing. With this the phase done condition is fulfilled.</p> <p>This event is also triggered at a restart.</p>
Phase deactivated	Is triggered if the phase was started and now is finished.
<b>Timeout</b>	Timeout events.
Phase started multiple times	A phase can only be active once. If it is activated

	several time in parallel, this event is triggered.
Waiting period unit allocation exceeded	Is triggered if the waiting duration for the unit allocation runs. Can also occur during Paused and Held.
Waiting period input interlocking exceeded	Is triggered if the waiting duration (time out) for the input interlocking expired.  Makes only sense if a <b>Interlocking condition</b> was defined.
Command parameters without value	Is triggered if the command tag should be toggled and the variable linked to the tag does not have a valid initial value.
Maximum execution period exceeded	Is triggered if the waiting duration (time out) for waiting for the phase done condition (Finished) was exceeded.
Waiting period following condition exceeded	Is triggered if the phase was not finished within the scheduled waiting duration (time out) although the phase done condition was fulfilled.
Linked variable invalid	<ul style="list-style-type: none"> <li>▶ If the value of a variable with status INVALID should be used, this event is created once per invalid variable and phase.</li> <li>▶ If the variable status changed from INVALID to not INVALID and back to INVALID, the reaction is again triggered when the variable is used.</li> <li>▶ If the phase is restarted, the event is triggered again when an invalid variable is used.</li> </ul> <p>At the following activities it is checked for invalid variable:</p> <ul style="list-style-type: none"> <li>▶ Source variable in reaction</li> <li>▶ Variable for phase done condition</li> <li>▶ Variable for input interlocking</li> <li>▶ Write command tag inversely</li> <li>▶ Variables for status change at transient states allowed</li> <li>▶ <b>Note:</b> In the event of an INVALID, the events are not necessarily processed in the order in which they are received. If an INVALID occurs whilst another event is being processed, this event can overtake the one that is currently being executed.</li> </ul>

<b>Status change</b>	<p>Status change events.</p> <p>If the phase changes its status, the corresponding reaction is activated.</p>
Status change: Running	The phase is executed.
Status change: Pausing	The phase is switched to <code>Paused</code> at the moment.
Status change: Paused	The phase is paused.
Status change: Continue	<p>The phase is resuming after a break.</p> <p>A status change in the object from <code>Paused</code> to <code>Running</code> triggers the events <code>Resuming</code> and <code>Running</code>.</p>
Status change: Holding	The phase is held at the moment.
Status change: Held	The phase was stopped.
Status change: Restarting	The phase is restarting at the moment.
Status change: Stopping	The phase is stopping at the moment.
Status change: Stopped	The phase was stopped.
Status change: Aborting	The phase is aborted at the moment.
Status change: Aborted	The phase was aborted.
Status change: Executed	The phase is finished.
Escape condition started	Is triggered if the <b>Escape condition</b> for exiting from a phase is started.
Escape condition fulfilled	Is triggered if the <b>Escape condition</b> for exiting from a phase is met.
<b>Mode change</b>	Events in relation to mode change in the REE
Mode change: Automatic	The REE switched to mode <code>Automatic</code> .
Mode change: Semi-automatic	The REE was switched to <code>semi-automatic mode</code> .
Mode change: Manual	The REE switched to mode <code>Manual</code> .
Close and restart Runtime	Events in relation to closing and restarting Runtime.
Exit Runtime initiated	<p>Is triggered if the Runtime is exited. This is an especially critical state for the Batch Control module as the recipe process does not stop in the control immediately. Therefore exiting the Runtime is prevented as long as module Batch Control saved all data. A process image is created which can later be used as starting point.</p> <p>Likewise it is made sure that the tags of action <b>Write set value</b> safely arrive at the control. Internally the</p>

	<p>phase is paused only when the writing confirmation from the driver ensued.</p> <p>For more details about existing the Runtime see chapter: Exit and restart Runtime (on page 246).</p> <p>At this event no reaction types of group <b>Influencing the recipe</b> are possible.</p>
Runtime restart	The Runtime was restarted.
<b>Interruptions and errors</b>	Events in relation to interruptions and errors in communication and on the PLC.
Loss of communication	This event reports that communication has been interrupted.
Loss of communication fixed	This event reports that the communication failure has been rectified.
Loss of communication acknowledged	This event reports that a displayed communication failure has been acknowledged.
PLC error	Is triggered if there is a PLC error.
PLC error rectified	Is triggered if a PLC error has been rectified.
PLC error rectified by deactivating the phase	Is triggered if there was a PLC error when a phase was ended. This was changed to <i>rectified</i> when it ended. Does not apply for a restart of the phase.

## EVENTS ON RESTART

The reactions `phase activated`, `phase started` and `phase deactivated` are always only executed once. These reactions are not triggered again after the phase has been restarted. The phase starts to run again, however it was not executed in full beforehand.

Along the same lines, the reaction `phase deactivated` is only triggered once the phase has been ended and not during a restart.

The `phase started` reaction is triggered if the unit allocation and the parallel execution detection has been executed. If the procedure has not exceeded this detection on restart, the reaction is triggered for the `Restart` command. If the process is already in an advanced state, the reaction is not carried out again.

## ALLOW EXECUTION BEFORE "PHASE STARTED"

Events can be also be approved before the `phase started` event. To do this, the **Allow execution before start event** property must be activated for the corresponding event. This property can only be configured

for events that are possible both before and after `phase started`. The value is automatically set according to type for all other events.

The following events are approved before and after "phase started":

- ▶ Mode change
- ▶ Status change
- ▶ Exit from a phase
- ▶ Exit Runtime initiated
- ▶ Restart Runtime
- ▶ Linked variable invalid
- ▶ Phase deactivated (can occur before if the phase was exited before `phase started`)
- ▶ Waiting period for subsequent conditions (can occur before if the phase was exited before `phase started`)
- ▶ Communication failures (however only from the point at which the values for the **Loss of communication** property are waited for)

The following events are only approved before "phase started":

- ▶ Phase activated
- ▶ Unit allocation
- ▶ Phase started more than once (exclusive execution)

All other events are only approved after the `Phase started` event.

## 7.5.4 Reaction types

In the properties of the reactions the reaction types more precisely defined and engineered. In group **Reactions** the following reaction types are available:

Reaction type	Description
<b>Tag set value</b>	<p>Influences command and return tag directly. All tag data types can be used.</p> <p>Attention: The value must be within of the set value limits of the variables which are linked at the tag. If this is not the case, an error message is created during the validation.</p>
<b>CEL entry</b>	<p>Creates entries in the CEL and log files. With this the reaction can be documented and the recipe process can be tracked later. For this property <b>Create CEL entry</b> must be activated. The text for the CEL is defined in property <b>CEL message text</b>.</p>
<b>Function</b>	<p>Makes it possible to link any zenon function.</p> <p>With this you can e.g. call up a pop-up in order to inform the user about a certain status or start a data backup.</p> <p>Note: In the network the function is always executed at the server.</p>
<b>Allocate tag</b>	<p>Makes it possible to perform a value assignment from <b>Source tag</b> to another <b>Target tag</b>. You can use both command tags and return tags. The data type of source and target tag must be identical otherwise an error is displayed at the validation of the recipe.</p>
<b>Recipe influence</b>	<p>Make it possible to:</p> <ul style="list-style-type: none"> <li>▶ Change the execution mode</li> <li>▶ Execute commands for the REE</li> <li>▶ execute phase commands</li> </ul> <p>With this you can react on serious events such as <code>Waiting duration exceeded</code> or <code>Linked variable invalid</code>.</p> <p>Note: Use this reaction type carefully as this reaction type influences the entire recipe process.</p> <p>For each event you can only once:</p> <ul style="list-style-type: none"> <li>▶ set the <b>Mode</b> and</li> <li>▶ write a single <b>Command</b></li> </ul> <p>Because e.g. it does not make sense to pause and hold the recipe at the same time with the same event.</p> <p>You can read more about commands in the commands and actions (on page 127) chapter.</p>

## 7.6 Parameters

TAGs are the communication interface to the control. With them all values are transferred to the control and also read back. To not have to work with complex and for the user incomprehensible variable names in module Batch Control, the abstract level is used. Each tag consists - for each phase - of a unique name and a description. In this, the engineer can give the recipe creator or user a description of what the tag is used for or which effects it has.

You can add any number of tags to a phase. We distinguish between command tags (on page 40) and return tags (on page 41). Command parameters are further subdivided into initial parameters and value parameters. Each tag can be switched between command and return tag at any time.

### 7.6.1 Detail view tag

Toolbar and context menu provide commands to create and administer variables of parameters.



Parameters	Description
<b>New initial tag</b>	Creates a new initial parameter in the detail view.
<b>New value tag</b>	Creates a value parameter in the detail view.
<b>New return tag</b>	Creates a new return tag in the detail view.
<b>Replace links</b>	Opens the dialog to replace links (on page 111).
<b>Copy</b>	Copies the selected entries to the clipboard.
<b>Paste</b>	Pastes the contents of the clipboard. If an entry with the same name already exists, the content is pasted as " <b>Copy of...</b> ".
<b>Delete</b>	Deletes selected entries after a confirmation from list.
<b>Remove all filters</b>	Removes all filter settings.
<b>Edit selected cell</b>	Opens the selected cell for editing. The binocular symbol in the header shows which cell has been selected in a highlighted line. Only cells that can be edited can be selected.
<b>Replace text in selected column</b>	Opens the dialog for searching and replacing texts.
<b>Properties</b>	Opens the <b>Properties</b> window for the selected entry.
<b>Help</b>	Opens online help.

## 7.6.2 Command TAGs

Command parameters transfer information and values to the controller. They can be subdivided into:

- ▶ **Initial parameters:** Command parameters that are set before the start event. They transfer information that must be stored before setting the input lock in the controller, for example, which control strategy (on page 44) is executed.
- ▶ **Value parameter:** Command parameters that are sent after input locking when the phase is executed.

Command tags contain the set values which should be transferred to the control. Initial parameters and value parameter are backed up (on page 245) and written to the controller. You can find the exact description in chapter: Process of a phase in detail (on page 238).

Command tags can also be used in transitions (on page 27), conditions (on page 27) and reactions (on page 29). Initial parameters and value parameters can have the same variable linked. This is taken into account when validating for multiple use of a variable.

Command parameters have a number of properties which can be defined via the property window. For this, the following applies:

- ▶ Each tag must be linked with a variable.
- ▶ The data type of the variable must correspond to the data type of the parameter.
- ▶ The set value limits of the parameter must be within the set value limits of the variable.

If this is not the case, error messages are created during the validation.

**Hint to property *Changeable in master recipe*:** With this you define whether the value of the command TAG may be modified by the creator of the master recipe. If e.g. machine tags should not be changeable in the recipe but defined fixedly, you must deactivate this property.

### ENGINEERING

To create a new command tag:

1. select the desired phase
2. Select, in the context menu, the command **New initial parameter** or **New value parameter**
3. a new command tag is created in the detail view

### NOTE ON COMPATIBILITY

If Runtime files are created for zenon 7.10 or older versions, then the initial parameters and value parameters are treated the same as command parameters again. Command parameters from zenon 7.10 or earlier are all converted to value parameters.



### 7.6.3 Return TAG

The return tags contain the return values with which the process action of the REE communicate its status. Normally the value is set by the control and evaluated by the REE. Return tags can be evaluated in transitions (on page 27) and conditions (on page 27).

Likewise they can be used in reactions (on page 29) and can also be written there. For this they are listed as target tags at **Allocate tag** and **Tag set value**.

## ENGINEERING

To create a new return tag:

1. select the desired phase for which you want to create a new return tag
2. in the context menu select menu item **New return tag**
3. a new command tag is created in the detail view

### 7.6.4 Example for status tag

To be able to communicate with the control, you normally need two status tags:

- ▶ one in write direction and
- ▶ one in read direction

The variable behind these parameters should have a numeric data type such as USINT or UINT. We recommend that you execute both parameters as return parameters. This may seem illogic for commands in write direction at first glance but has the following background: At the execution of the phase, all command tags are written. With this they are visible in the list of command tags and can therefore be deleted accidentally. This makes no sense for a command for the control. The goal is not just to communicate a single value to the control as command but to transmit the status of the phase in the recipe.

Especially at the writing of a command tag it makes sense to not simply inform the PLC about the writing but the status when all values have been written and the PLC therefore can start to process the process action.

For this it is best to use the reaction to event `Finished writing command tags`. At reactions to an event you can also write values to an return tag. Therefore it is better to use return tags for both status tags.

Here is an example about which values the tags can take on:

## STATUS TAG IN WRITE DIRECTION (TO THE PLC): COMMANDS

Value	Name of the event
0	not defined
1	Phase started
2	Finished writing command tags
3	Phase finished: Phase done condition fulfilled and <b>Minimum execution duration</b> reached (if engineered)
4	Phase deactivated
5–9	Reserve
10	Status change: Pausing
11	Status change: Resuming
12	Status change: Holding
13	Status change: Restarting
14	Status change: Stopping
15	Status change: Aborting
16–19	Reserve
20	Mode change: Automatic
21	Mode change: Semi-automatic
22	Mode change: Manual
23–29	Reserve
30	Exit Runtime initiated
31	Runtime restart
32	Unit allocation not possible
33	Waiting period unit allocation exceeded
34	Input interlocking blocked
35	Waiting period input interlocking exceeded
36	Maximum execution period exceeded
37	Waiting period following condition exceeded
38	Phase started multiple times

You can find the exact meaning of the events in the Event type (on page 32) chapter.

For each entry in the table you define a corresponding reaction for writing the status value at the phase.

**Hint:** Use the same tag label for all phases; e.g. `StatusPhase`. Then you only have to engineer the

reaction at one phase and can then transfer it to all phases via copy & paste. You can of course also copy the tags. Do not forget to correct the variable. They must match the respective phase.

#### TAG IN READ DIRECTION (FROM THE PLC): RETURN VALUES

Value	Description	Linked in property
0 - 1	Not defined	
2	Process action finished	<b>Phase done condition</b>
3 - 9	Reserve	
10	Process action paused	<b>Paused</b>
11	Reserve	
12	Process action held	<b>Held</b>
13	Process action restarted	<b>Restarted</b>
14	Process action stopped	<b>Stopped</b>
15	Process action aborted	<b>Aborted</b>

Link the values with a formula in the respective property.

**Hint:** You can copy the formula and just change the respective value. If you set this setting at the start of setting the parameters, you can copy the entire phase and with that have these settings for all phases.

### 7.6.5 Duration of execution

The execution duration is controlled via two independent properties. Their values must not complement one another.

#### MAXIMUM EXECUTION DURATION

The **Maximum execution duration** refers to `Phase deactivated` and therefore to the process. It is not connected to the **Minimum execution duration**.

#### MINIMUM EXECUTION DURATION

Property **Minimum execution duration** defines how long zenon waits after writing the command tag independent of the check of the phase done condition. During the execution the maximum execution duration is checked. An event is triggered if this is exceeded. This can be linked to a reaction. This happens regardless of whether the phase still checks its **Phase done condition** or only waits for the **Minimum execution duration**.

The length of the **minimum execution** duration can exceed the **maximum execution** duration.

### EXAMPLE

- ▶ There is a phase: **Start mixing**. The confirmation that the mixer runs must not take longer than 5 seconds before a warning of an error is displayed.  
Engineering: Property **Maximum execution duration** gets value 5 seconds with corresponding reaction.
- ▶ The mixer however should run 15 minutes before the next phase is executed.  
Engineering: Property **Minimum execution duration** gets value 15 minutes.

With this the minimum execution duration is 15 minutes and the maximum execution duration 5 seconds.

## 7.7 Control strategies

Control strategies make it possible to set parameters for different versions of a phase. Only the command parameters allocated to the control strategy are sent for each control strategy.

### CONFIGURING CONTROL STRATEGIES

To use control strategies, these must be activated in the phase. To do this:

1. Highlight the desired phase
2. Go to property group **Control strategies**\
3. Activate the checkbox in front of the **Active control strategies** property.
4. Select a parameter in the **control strategy tag** property. This parameter defines the control strategy that is active in Runtime.
5. The phase is thus displayed in the Editor with the **Control strategies** node.
6. Right click on the node and select the entry **New control strategy** in the context menu.
7. A new control strategy is created.
8. Configure the properties of the control strategy.  
In doing so, please note:
  - **Name** and **control strategy number** of the control strategy must be unique within the phase.
  - The **Name** must not be empty, contain a dot, consist of only spaces and must be within a maximum of 256 characters long.
9. Add the desired command parameters.

**Note:** Clicking on a parameter add its properties.

Only the following properties in the **Write set value** group can be edited:

- **Tag value**
- **Min. value**
- **Max. value**
- **Changeable in master recipe**

All other properties cannot be edited. To edit these, switch to the parameter list of the phase.

## COPYING CONTROL STRATEGIES

Control strategies can be copied using commands in the context menu and the toolbar and inserted into the same or other phases.

If control strategies are copied throughout phases, units or a project, only the parameter linkings that are also to be triggered in the new phase are inserted. In doing so, the conditions are the same as for inserting parameter linking (on page 47).

### 7.7.1 Control strategies node context menu

Right-clicking on the **Control strategies** node opens a context menu with the following entries:

Parameters	Description
<b>New Control strategy</b>	Creates a new control strategy
<b>Paste control strategy</b>	Pastes a control strategy from the clipboard.  Copied control strategies are adapted when pasted into a phase so that <b>Name</b> and <b>control strategy number</b> are made unique, if this is not already the case in the phase.
<b>Help</b>	Opens online help.

### 7.7.2 Context menu selected control strategy

Right-clicking on the control strategy opens a context menu with the following entries:

Parameters	Description
<b>Add command parameters</b>	Opens the dialog to select command parameters (on page 260).
<b>Insert parameter linking</b>	Inserts parameters that have been copied into the control strategy list of a different control strategy to the selected control strategy.
<b>Rename</b>	Highlights the name to be renamed.
<b>Delete</b>	Deletes the selected control strategy after a confirmation message.
<b>Copy</b>	<p>Copies the selected control strategy. This can be pasted using <b>Paste</b> in the context menu of the <b>Control strategies</b> node.</p> <p>Copied control strategies are adapted when pasted into a phase so that <b>Name</b> and <b>control strategy number</b> are made unique, if this is not already the case in the phase.</p>
<b>Help</b>	Opens online help.

### 7.7.3 Toolbar and control strategy list context menu

Entries in the control strategy list can be edited using symbols or entries in the context menu.



The following are available in the context menu and the toolbar:

Parameters	Description
<b>Add command parameters</b>	Opens the dialog to select command parameters (on page 260).
<b>Copy</b>	Copies the selected entries to the clipboard.
<b>Paste</b>	Pastes the contents of the clipboard. If an entry with the same name already exists, the content is pasted as " <b>Copy of...</b> ".
<b>Delete</b>	Deletes selected entries after a confirmation from list.
<b>Remove all filters</b>	Removes all filter settings.
<b>Edit selected cell</b>	Opens the selected cell for editing. The binocular symbol in the header shows which cell has been selected in a highlighted line. Only cells that can be edited can be selected.
<b>Properties</b>	Opens the <b>Properties</b> window for the selected entry.
<b>Help</b>	Opens online help.

## 7.7.4 Parameters

### LINKING PARAMETERS

There are different methods available to link parameters to a control strategy:

- ▶ To do this, select the **Add command parameter** command in the context menu of the control strategy.
- ▶ Select, in the detail view of the control strategies in the toolbar or in the context menu of a parameter, the **Add command parameter** command.
- ▶ Drag the parameter from the parameter list of the phase by dragging & dropping it onto the control strategy. Only command parameters (on page 40) are linked.

**Note:** If a parameter is deleted for a phase, the attendant parameter linking is also deleted for all control strategies.

### DELETING OR RESTORING LINKED VALUES

Parameters of the control strategies take on the values of the parameters with which they are linked. This linking can be deleted by:

- ▶ Overwriting the value
- ▶ Separating the linking via the context menu

The context menu can also be used to restore the link to the source parameter again.

You can read details on linked values in the **Linked properties** chapter in the **Editor** manual.

### COPYING PARAMETERS

Existing parameter linking can be copied between control strategies. Copying is possible throughout via phases, units and projects. When inserting parameter linkings, an attempt is made to link the names accordingly. No new parameters are created.

Insertion is possible if there is no parameter with the copied name in the target phase or the corresponding parameter is already part of the control strategy.

## 7.8 Keyboards

Parameters can be amended in Runtime. Adapted keyboards are available for this.

## KEYBOARDS FOR BATCH CONTROL

You define keyboards for use in the Batch Control module in general in the properties for the module in the **Edit tag/Keyboards** group. Define the desired keyboards for:

- ▶ **Binary tags**
- ▶ **Numeric tags**
- ▶ **String tags**
- ▶ **Time period tags**

With the **Reason for value change necessary** property, you can also stipulate that each value change must have a reason. If this property is active, a dialog to enter the reason is opened before the change is made.

## KEYBOARD FOR WRITING SET VALUE FOR PARAMETER

You define keyboards for writing set values for individual parameters in the properties of the respective parameter in the **Write set value/Keyboards** group. To do this:

1. Navigate to the **Write set value** group in the parameter properties.
2. Activate, in the **Keyboards** subgroup, the **Use screen Keyboard** property
3. In the **Screen Keyboard** property, define which keyboard screen is to be called up in Runtime

## USING A KEYBOARD

Values can be edited in the master recipe and in the control recipe if the recipe status and the parameter settings allow this. The minimum and maximum can only be changed in the master recipe and only for numerical parameters and duration parameters.

The following applies for the use of keyboards Batch Control screens:

- ▶ Only one keyboard can be active at a time. If a new one is called up, the previous one is closed.
- ▶ If the keyboard is active and the mouse is double-clicked with the pointer in a column in which nothing can be changed, nothing happens.
- ▶ The following is applicable to value, minimum and maximum of a parameter:
  - If a parameter itself is linked to a keyboard, this is used.
  - If no keyboard is linked to the parameter, the keyboard that is generally linked to Batch Control in the **Edit tag** group is used.
  - If a keyboard is linked to the parameter, but this is no longer available, the keyboard that is also generally linked to Batch Control in the **Edit tag** group is used.
  - No keyboard is opened if this also does not exist.
- ▶ Keyboards for binary inputs can be provided using the **On**, **Off** and **Toggle** keys.



- ▶ When switching the units of measurement, the min/max values and the unit name are sent to the system variables again.
- ▶ For the **Changeable in control recipe property**, the keyboard for **Binary tags** defined in the **Edit tag** group is searched for. If none is linked, then a search is carried out for a keyboard with the standard name **SETBOOKBD**. No keyboard is opened if this also does not exist.
- ▶ Keyboards that are directly linked to the parameter can no longer be changed after a recipe has been approved. If the keyboard linked to the parameter is deleted, only the keyboard defined in the **Edit tag** group can be used.

## 7.9 Input lock

With the help of an input interlocking the phase is only executed in the Runtime when the condition for the input interlocking is fulfilled.

The input interlocking is configured via property **Interlocking condition**. Via the formula editor (on page 255) the condition is defined which the input interlocking must fulfill. The formula can consist of one or more command tags and return tags of the phase. Value and status of the variables can also be used. The formula returns **TRUE** or **FALSE** as result. The condition can be displayed in the Runtime but cannot be changed there. The waiting period for the input interlocking is configured with the help of property **Waiting period input interlocking**.

If an input interlocking was defined, the phase is executed as soon as the following conditions are fulfilled:

- ▶ the phase is active
- ▶ the unit is allocated
- ▶ the phase is not active twice
- ▶ the phase is not already executed
- ▶ the input interlocking is fulfilled

If no input interlocking is linked, the phase is executed when:

- ▶ the phase becomes active
- ▶ the unit is allocated
- ▶ the phase is not executed twice
- ▶ the needed variables all have a value



### Information

*You can find more information about input interlockings in chapter Processing a phase in detail (on page 238).*

## 7.10 Create screen of type Batch Control

### CREATE SCREEN OF TYPE BATCH CONTROL

#### ENGINEERING

Steps to create the screen:

1. Create a new screen:  
In the tool bar or the context menu of the **Screens** node, select the **New screen** command.  
An empty `Standard` screen is created.
2. Change the properties of the screen:
  - a) Name the screen in the **Name** property.
  - b) Select `Batch Control` in the **Screen type** property.
  - c) Select the desired frame in the **Frame** property.
3. Configure the content of the screen:
  - a) select menu item **Control elements** from the menu bar
  - b) Select `Insert template` in the drop-down list.  
The dialog to select pre-defined layouts is opened. Certain control elements are inserted into the screen at predefined positions.
  - c) Remove elements that are not required from the screen.
  - d) If necessary, select additional elements in the **Elements** drop-down list. Place these at the desired position in the screen.
4. Create a screen switch function.



CONTROL ELEMENTS

PFC Editor  
Recipe Editor  
Typ: STATIC  
ID: 53507

Master recipes

Column selection...Format Columns...

Master recipes list  
Typ: STATIC  
ID: 53504

New...  
Open  
Duplicate  
Rename  
New version  
Highlight as outdated  
Delete  
Release

Display control recipes in list

currently executed control recipes  
Typ: BUTTON

finished control recipes  
Typ: BUTTON

prepared control recipes  
Typ: BUTTON

outdated control recipe  
Typ: BUTTON

Dynamically update control recipe list  
Typ: BUTTON

Display associated control recipes in list

Control recipe

Column selection...Format Columns...

Control recipe list  
Typ: STATIC  
ID: 53513

New...  
Open  
Duplicate  
Delete  
Rename  
Starting

51

Control element	Description
<b>Insert template</b>	Inserts control elements for master recipes or control recipes on predefined locations on the screen. These control elements can be supplemented, reduced and positioned newly.
<b>Default (master recipes)</b>	Inserts control elements for master recipes on predefined locations on the screen. These control elements can be supplemented, reduced and positioned newly.
<b>Default (control recipes)</b>	Inserts control elements for control recipes on predefined locations on the screen. These control elements can be supplemented, reduced and positioned newly.
<b>Recipe Editor</b>	Adds the licenses editor for creating master and control recipes.
<b>Recipe control</b>	Control elements for the recipe control.
<b>Execution commands of the recipe</b>	Control elements for recipe commands: <ul style="list-style-type: none"> <li>▶ Start recipe</li> <li>▶ Recipe pausing</li> <li>▶ Recipe resuming</li> <li>▶ Recipe holding</li> <li>▶ Restart recipe</li> <li>▶ Recipe stopping</li> <li>▶ Recipe aborting</li> </ul>
<b>Execution commands of the phases</b>	Control elements for phase commands: <ul style="list-style-type: none"> <li>▶ Phase pausing</li> <li>▶ Phase resuming</li> <li>▶ Phase holding</li> <li>▶ Restart phase</li> <li>▶ Escape phase</li> </ul>
<b>Switch execution mode</b>	Control elements for execution modes: <ul style="list-style-type: none"> <li>▶ Switch to automatic mode</li> <li>▶ Switch to semi-automatic mode</li> <li>▶ Switch to manual mode</li> </ul>
<b>Execution navigation</b>	Control elements for navigation in recipes: <ul style="list-style-type: none"> <li>▶ Continue recipe only on selected active elements</li> <li>▶ Continue recipe at all execution positions</li> </ul>



	► Skip active condition
--	-------------------------

<b>General</b>	<p>General control elements:</p> <ul style="list-style-type: none"> <li>▶ Check recipe for errors</li> <li>▶ Edit element</li> <li>▶ Display grid</li> <li>▶ Colors for background and grid</li> </ul>
<b>Operation navigation</b>	<p>Control elements for navigation in operations:</p> <ul style="list-style-type: none"> <li>▶ Change to operation template</li> <li>▶ Switch to main recipe</li> </ul>
<b>Master recipes</b>	Control elements for master recipes.
<b>Master recipes list</b>	<p>In this list all master recipes can be displayed. The display can be limited by filters to an individual selection.</p> <p>The filtering can be preset in the zenon Editor in the screen switch function (on page 59). Online filtering is also possible. These filters are discarded when the screen is called up again. A permanent definition is only possible in the zenon Editor.</p> <p>All commands are also possible in the context menu of the list. The commands for list management can be called from the header of the list. The commands for recipe management can be called at editing one or more recipes.</p> <p>The recipes in the list cannot be edited directly in the list. Renaming, changing the description or changing the recipe status is only possible with the corresponding commands.</p>
<b>Column selection master recipe...</b>	<p>Opens a dialog in order to determine which columns should be displayed (on page 65).</p> <p><b>Attention:</b> These changes are discarded when the screen is called up again. A permanent definition is only possible in the zenon Editor.</p>
<b>Format columns master recipe...</b>	<p>Opens a dialog to edit the column settings (on page 67).</p> <p><b>Attention:</b> These changes are discarded when the screen is called up again. A permanent definition is only possible in the zenon Editor.</p>
<b>New master recipe...</b>	Opens dialog for creating a new master recipe (on page 140).
<b>Create new version of master recipe</b>	Creates a new version (on page 192) of the selected master recipe. This must be approved or marked as obsolete.

<b>Rename master recipe</b>	<p>Only active if exactly one master recipe was selected. The dialog for the input of a unique name and the description is opened.</p> <p>Recipes can only be renamed if they are in status <code>Editable</code>.</p> <p>Also use this function in order to changed the description of the control recipe.</p> <p>When renaming a recipe, a CEL entry is created.</p>
<b>Duplicate master recipe</b>	<p>Only active if precisely one recipe was selected. Creates a copy of the selected recipe. At the creation of the copy, the version of the recipe saved on the hard disk is used. If the recipe is just edited in another computer and the changes have not yet been saved, the changes are not applied. The dialog for the input of a unique name and the description is opened.</p> <p>The copy of the recipe automatically receives status <code>Editable</code> and can be edited further.</p> <p>When duplicating a recipe, a CEL entry is created.</p>
<b>Delete master recipe</b>	<p>Deletes the selected recipes irrevocably. If the recipe is opened on another computer for editing, it is automatically closed there.</p> <p>Deleting is only possible if there are no control recipes which are based on the master recipe. First you must delete all control recipes.</p> <p>Recipes which are currently executed in test mode (master recipe status: <code>Test in execution</code>) cannot be deleted. First they must be <code>finished</code>, <code>stopped</code> or <code>canceled</code>.</p> <p>If recipes must not be deleted - e.g. in an FDA-regulated environment - it is recommended that this button is not configured or that it is given an appropriate <b>Authorization level</b>.</p> <p>A CEL entry is created when a recipe is deleted.</p>
<b>Open master recipe</b>	<p>Opens the selected master recipe in the recipe editor if screen element <code>Recipe editor</code> exists in the screen. Each selected master recipe is opened in a separate tab of the recipe editor.</p>
<b>Switch master recipe to edit mode</b>	<p>Changes the master recipe status of the selected recipes to <code>Editable</code>. In this status, recipes can again be edited completely.</p> <p>Only recipes in <code>Test mode</code> can be set back to <code>Editable</code>.</p>
<b>Switch master recipe to test mode</b>	<p>Changes the master recipe status of the selected recipe to <code>Test mode</code>. Only faultless recipes can be switched to test mode. If error occur during the validation (on page</p>

	<p>194), you must first fix them.</p> <p>Recipes in the test mode can be executed but no longer reengineered. For details about the states see chapter Recipe types and recipe states (on page 135).</p>
<b>Release master recipe</b>	<p>Changes the master recipe status of the selected recipes to <b>Released</b>. Only recipes without errors can be released. If error occur during the validation (on page 194), you must first fix them.</p> <p>Only recipes in status <b>Test mode</b> and <b>Editable</b> can be released.</p> <p>Released recipes can no longer be edited. Control recipes can only be created from released recipes. For details about the states see chapter Recipe types and recipe states (on page 135).</p> <p>When releasing a recipe, a CEL entry is created.</p>
<b>Highlight master recipe as outdated</b>	<p>Changes the status of the recipe to <b>outdated</b>. The recipe can no longer be edited or approved. No control recipe can be created on the basis of this recipe.</p>
<b>Display associated control recipes in list</b>	<p>Displays all control recipes that are based on the selected master recipe and that comply with the set filter criteria.</p>
<b>Dynamically update control recipe list.</b>	<p>Deactivates the <b>Display associated control recipes in list</b> button. When selecting a master recipe, all attendant control recipes are displayed automatically.</p>
<b>Filter for displaying the control recipe</b>	<p>Makes it possible to filter control recipes for the following criteria:</p> <ul style="list-style-type: none"> <li>▶ <b>Currently executed control recipes:</b> Displays only control recipes which are currently executed. Only takes effect as soon as you click on <b>Show associated control recipes in list</b>.</li> <li>▶ <b>Prepared control recipes:</b> Display only control recipes which are prepared for execution. Only takes effect as soon as you click on <b>Show associated control recipes in list</b>.</li> <li>▶ <b>Completed control recipes:</b> Displays only control recipes which have already been executed. Only takes effect as soon as you click on <b>Show associated control recipes in list</b>.</li> <li>▶ Outdated control recipe</li> </ul>
<b>Control recipe</b>	<p>Control elements for control recipes.</p>
<b>List control recipes...</b>	<p>In this list all control recipes can be displayed. The display can be limited by filters to an individual</p>



	<p>selection.</p> <p>Per default the list is empty. To fill the list, you must:</p> <ul style="list-style-type: none"> <li>▶ select master recipes</li> <li>▶ Set the <b>currently-executed control recipes</b>, <b>prepared control recipes</b> and <b>completed control recipes</b> filters</li> <li>▶ click button <b>display associated control recipes in list</b></li> </ul> <p>In addition to the filters mentioned above, you can filter the list itself. The filtering can be preset in the zenon Editor in the screen switch function (on page 59). Online filtering is also possible. These filters are discarded when the screen is called up again. A permanent definition is only possible in the zenon Editor.</p> <p>All commands are also possible in the context menu of the list. The commands for list management can be called from the header of the list. The commands for recipe management can be called at editing one or more recipes.</p> <p>The recipes in the list cannot be edited directly in the list. Renaming, changing the description or starting the recipes is only possible with the corresponding commands.</p>
<b>Column selection control recipe...</b>	<p>Opens a dialog in order to determine which columns should be displayed (on page 65).</p> <p><b>Attention:</b> These changes are discarded when the screen is called up again. A permanent definition is only possible in the zenon Editor.</p>
<b>Format columns control recipe...</b>	<p>Opens a dialog to edit the column settings (on page 67).</p> <p><b>Attention:</b> These changes are discarded when the screen is called up again. A permanent definition is only possible in the zenon Editor.</p>
<b>New control recipe...</b>	<p>Opens the dialog (on page 205) for entering a unique name and a description for the control recipe. The uniqueness of the name is also checked in the zenon network. The name must only be unique within the master recipes. Control recipes which are based on other master recipes may have the same name. The uniqueness within module Batch Control is achieved by always referencing the master recipe name and the control recipe name.</p>

	When creating a control recipe, a CEL entry is created.
<b>Rename control recipe</b>	<p>Only active if exactly one control recipe was selected. The dialog for the input of a unique name and the description is opened.</p> <p>Recipes can only be renamed if they are in status <i>Prepared</i>.</p> <p>Also use this function in order to changed the description of the control recipe.</p>
<b>Duplicate control recipe</b>	<p>Only active if precisely one recipe was selected. Creates a copy of the selected recipe. At the creation of the copy, the version of the recipe saved on the hard disk is used. If the recipe is just edited in another computer and the changes have not yet been saved, the changes are not applied. The dialog for the input of a unique name and the description is opened.</p> <p>The copy of the recipe automatically gets the status <i>Prepared</i> and can therefore be edited and started. The execution status (on page 231) of the duplicate is set to <i>automatic</i>.</p> <p>When duplicating a recipe, a CEL entry is created.</p>
<b>Delete control recipe</b>	<p>Deletes the selected recipes irrevocably. If the recipe is opened on another computer for editing, it is automatically closed there.</p> <p>Deleting is only possible if all selected recipes are not executed (control recipe status: In execution). In execution: First they must be finished, stopped or canceled.</p> <p>If recipes must not be deleted - e.g. in an FDA-regulated environment - it is recommended that this button is not configured or that it is given an appropriate <b>Authorization level</b>.</p> <p>A CEL entry is created when a recipe is deleted.</p>
<b>Open control recipe</b>	<p>Opens the selected control recipe in the recipe editor if screen element <i>Recipe editor</i> exists in the screen. Each selected control recipe is opened in a separate tab of the recipe editor.</p>
<b>Start control recipe</b>	<p>Starts the selected control recipe in the set execution mode. The recipes are executed invisibly at the Server. It is not necessary that the recipe is opened in the recipe editor.</p>
<b>Parameter lists</b>	<p>List box for the display of parameters.</p> <p>Two list boxes can be created. These are configured in the screen switching (on page 91).</p>

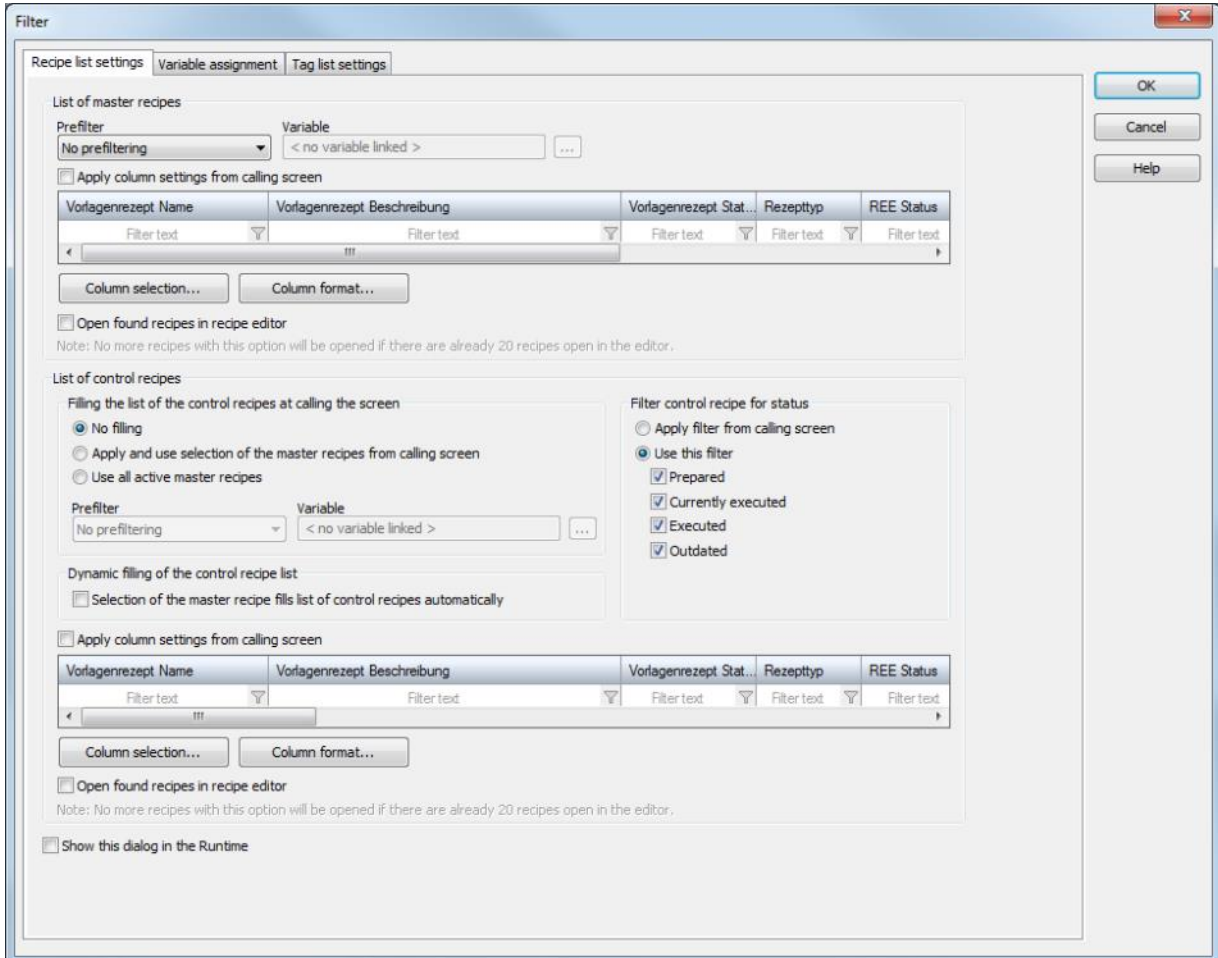
<b>XML import/export</b>	Control element for XML import/export.
<b>Export master recipes...</b>	Exports the selected master recipe as an XML file.
<b>Export control recipes...</b>	Exports the selected control recipe as an XML file.
<b>XML Import...</b>	Imports the selected XML file.

## 7.11 Screen switch Batch Control

To use Batch Control in the Runtime, engineer a screen switch function to a screen of type `Batch Control`:

1. Select the **New function** command in the **Functions** node
2. select the **Screen switching** function
3. select the screen of type `Batch Control`
4. the filter dialog (on page 60) is displayed
5. configure the
  - a) Settings (on page 60) for the list of the master recipes/control recipes including prefiltering (on page 71)
  - b) Variable allocations (on page 73)
  - c) Tag list settings (on page 91)

6. link the function with a button on the screen in order to switch in the Runtime



### 7.11.1 Recipe list settings

The settings are configured for:

- List of master recipes
- Control Recipes List

► Prefilter (on page 71)

**Filter**

Recipe list settings | Variable assignment | Tag list settings

List of master recipes

Prefilter:  Variable:

☐ Apply column settings from calling screen

Vorlagenrezept Name	Vorlagenrezept Beschreibung	Vorlagenrezept Stat...	Rezepttyp	REE Status
Filter text	Filter text	Filter text	Filter text	Filter text

Column selection... Column format...

☐ Open found recipes in recipe editor

Note: No more recipes with this option will be opened if there are already 20 recipes open in the editor.

List of control recipes

Filling the list of the control recipes at calling the screen

☒ No filling

☐ Apply and use selection of the master recipes from calling screen

☐ Use all active master recipes

Prefilter:  Variable:

Dynamic filling of the control recipe list

☐ Selection of the master recipe fills list of control recipes automatically

Filter control recipe for status

☐ Apply filter from calling screen

☒ Use this filter

☒ Prepared

☒ Currently executed

☒ Executed

☒ Outdated

☐ Apply column settings from calling screen

Vorlagenrezept Name	Vorlagenrezept Beschreibung	Vorlagenrezept Stat...	Rezepttyp	REE Status
Filter text	Filter text	Filter text	Filter text	Filter text

Column selection... Column format...

☐ Open found recipes in recipe editor

Note: No more recipes with this option will be opened if there are already 20 recipes open in the editor.

☐ Show this dialog in the Runtime

OK Cancel Help

## LIST OF MASTER RECIPES

Parameters	Description
<b>List of master recipes</b>	Configuration for master recipes.
<b>Prefilter</b>	<p>Select from a drop-down list whether master recipes should be pre-filtered when called up. Possible settings:</p> <ul style="list-style-type: none"> <li>▶ No prefiltering: Recipes are not pre-filtered.</li> <li>▶ ID from variable: Recipes are filtered according to ID. Filter condition is defined in the <b>Variable</b> property.</li> <li>▶ Name from variable: Recipes are filtered according to recipe name. Filter condition is defined in the <b>Variable</b> property.</li> </ul>
<b>Variable</b>	Definition of the variables that provide the values for the prefiltering. Click the button ... and a dialog opens to configure the variables.
<b>Apply column settings from calling screen</b>	Active: The column settings are accepted by the screen that is calling them up in Runtime. The corresponding properties can no longer be configured in the Editor.
<b>List field</b>	Display of the configured columns.
<b>Column selection</b>	Opens the dialog for selecting the columns
<b>Column Format</b>	Opens a dialog (on page 174) to format the columns.
<b>Open found recipes in recipe editor</b>	<p>Active: The recipes found are opened in the recipe editor.</p> <p>Note: The first 20 recipes found is the maximum that can be opened.</p>

## CONTROL RECIPES LIST

Parameters	Description
<b>Control Recipes List</b>	Configuration for control recipes.

## FILTER CONTROL RECIPE FOR STATUS

Parameters	Description
<b>Filter control recipe for status</b>	Settings for the filtering of the control recipes according to their status.
<b>Apply filter from calling screen</b>	Active: The filter is accepted from the calling screen.
<b>Use this filter</b>	<p>Selection of criteria for the status of a recipe that is to be called up. Several statuses can be selected by selecting the corresponding checkbox:</p> <ul style="list-style-type: none"> <li>▶ Prepared</li> <li>▶ Currently executed</li> <li>▶ Executed</li> </ul>

	<ul style="list-style-type: none"> <li>▶ Outdated</li> </ul>
<b>Filling the list of the control recipes at calling the screen</b>	<p>Settings for the filling of the list when called up. Select an option.</p> <ul style="list-style-type: none"> <li>▶ No filling</li> <li>▶ Apply and use selection of the master recipes from calling screen</li> <li>▶ Use all active master recipes</li> </ul>
<b>Prefilter</b>	<p>Select from a drop-down list whether control recipes should be prefiltered when called up. Possible settings:</p> <ul style="list-style-type: none"> <li>▶ No prefiltering: Recipes are not pre-filtered.</li> <li>▶ ID from variable: Recipes are filtered according to ID. Filter condition is defined in the <b>Variable</b> property.</li> <li>▶ Name from variable: Recipes are filtered according to recipe name. Filter condition is defined in the <b>Variable</b> property.</li> <li>▶ Job ID from variable: Recipes are filtered according to job ID. Filter condition is defined in the <b>Variable</b> property.</li> </ul>
<b>Variable</b>	<p>Definition of the variables that provide the values for the prefiltering. Click the button ... and a dialog opens to configure the variables.</p>

#### DYNAMIC FILLING OF THE CONTROL RECIPE LIST

Parameters	Description
<b>Dynamic filling of the control recipe list</b>	Settings for dynamic filling of the list.
<b>Selection of the master recipe fills list of control recipes automatically</b>	Active: When switching in Runtime, the list of control recipes always displays the master recipes selected at this point in time.
<b>Apply column settings from calling screen</b>	Active: The column settings are accepted by the screen that is calling them up in Runtime. The corresponding properties can no longer be configured in the Editor.
<b>List field</b>	Display of the configured columns.
<b>Column selection</b>	Opens the dialog for selecting the columns
<b>Column Format</b>	Opens a dialog (on page 174) to format the columns.
<b>Open found recipes in recipe editor</b>	<p>Active: The recipes found are opened in the recipe editor.</p> <p>Note: The first 20 recipes found is the maximum that can be opened.</p>
<b>Show this dialog in the Runtime</b>	<p>Active: When calling up the function in Runtime, this dialog is opened and the user can adjust the configuration before execution.</p> <p>The dialog is displayed on the current computer in Runtime. During network operation when activating the client the dialog is also displayed on the client</p>

**CLOSE DIALOG**

Parameters	Description
<b>OK</b>	Applies all changes in all tabs and closes the dialog.
<b>Cancel</b>	Discards all changes in all tabs and closes the dialog.
<b>Help</b>	Opens online help.

Note for variable selection using name or ID: For the selection of variables according to name or ID, numerical variables and string variables can be selected respectively. The data types are converted to the respective correct form.



## Column selection

### OPTIONS

Button	Function
<b>Available columns</b>	List of columns that can be displayed in the table.
<b>Selected columns</b>	Columns that are displayed in the table.
<b>Add -&gt;</b>	Moves the selected column from the available ones to the selected items. After you confirm the dialog with OK, they are shown in the detail view.
<b>Add all -&gt;</b>	Moves all available columns to the selected columns.
<b>&lt;- Remove</b>	Removes the marked columns from the selected items and shows them in the list of available columns. After you confirm the dialog with OK, they are removed from the detail view.
<b>&lt;- Remove all</b>	All columns are removed from the list of the selected columns.
<b>Up</b>	Moves the selected entry upward. This function is only available for unique entries, multiple selection is not possible.
<b>Down</b>	Moves the selected entry downward. This function is only available for unique entries, multiple selection is not possible.

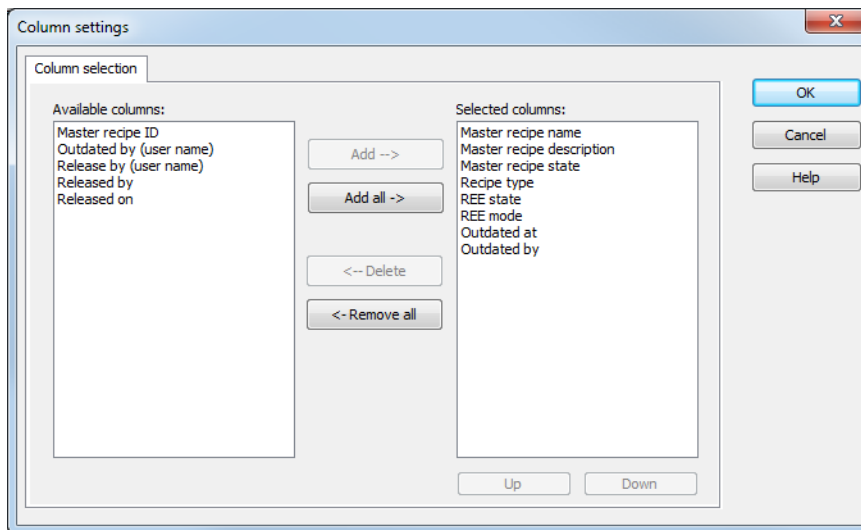
### CLOSE DIALOG

Parameters	Description
<b>OK</b>	Applies settings and closes the dialog.
<b>Cancel</b>	Discards all changes and closes the dialog.
<b>Help</b>	Opens online help.

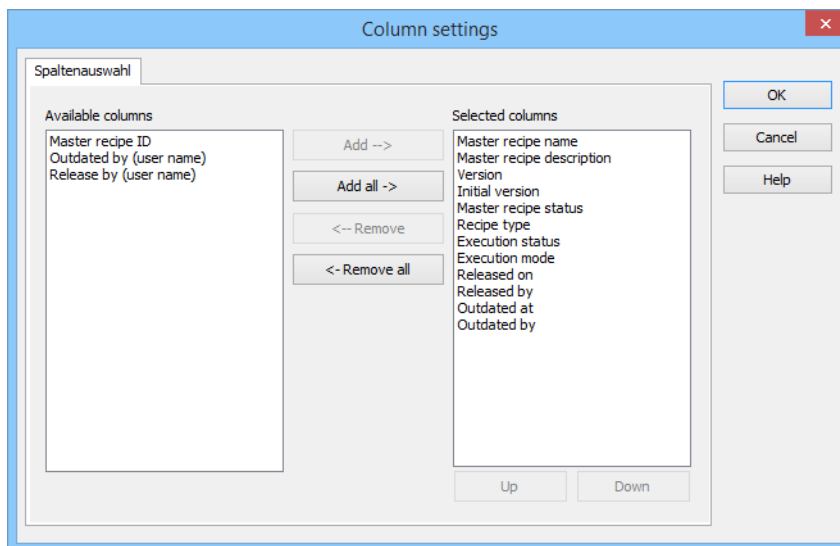
**Note:** These settings are only used in Runtime for dockable windows (on page 118) if there is no Runtime profile available for the user who is logged in.

### EXAMPLES OF COLUMN SELECTION

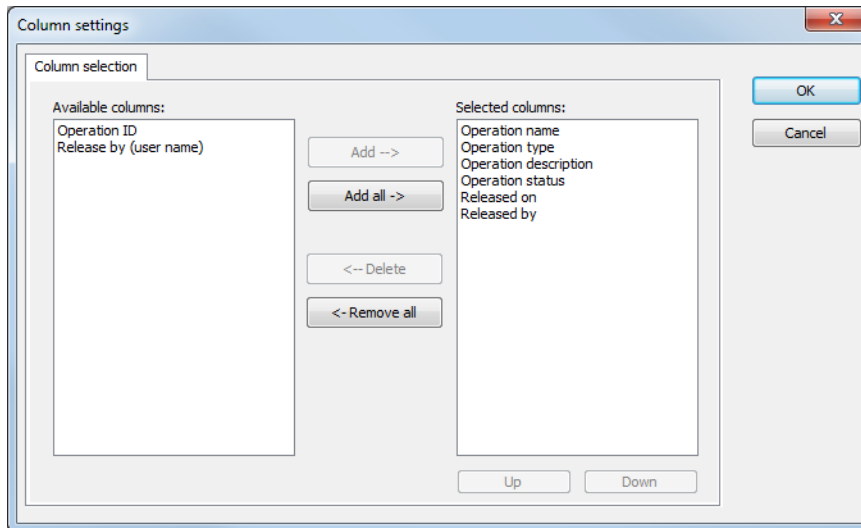
Column selection for list of the master recipes (on page 139):



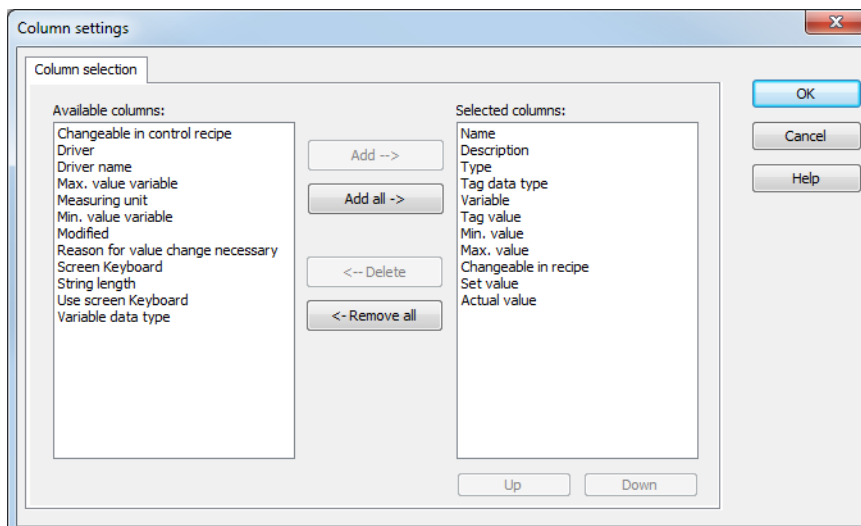
Column selection for list of the control recipes (on page 204):



Column selection for list of the operations (on page 194) (only available in Runtime):



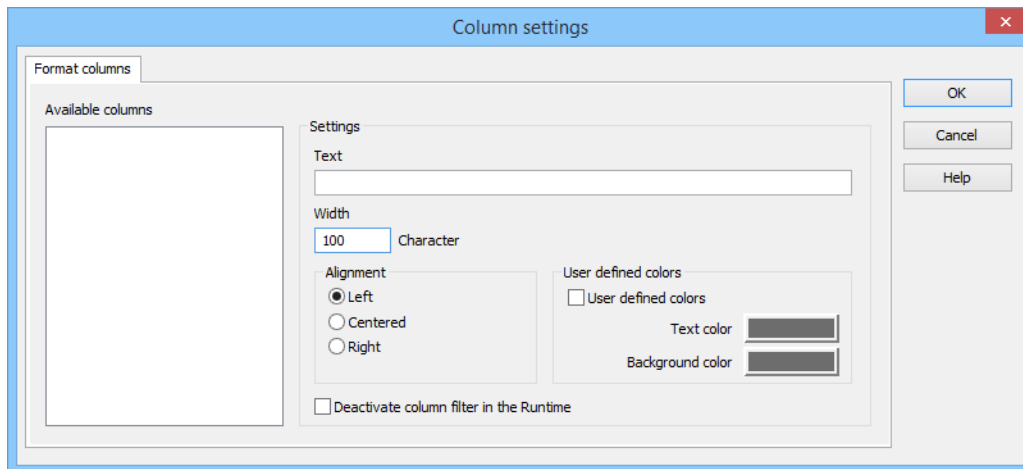
Column selection for Tag lists (on page 91)



## Column Format

In this dialog you define the column format:

Configuration of the properties of the columns for configurable lists. The settings have an effect on the respective list in the Editor or - when configuring screen switching - in Runtime.



## AVAILABLE COLUMNS

Parameters	Description
<b>Available columns</b>	List of the available columns via <b>Column selection</b> . The highlighted column is configured via the options in the <b>Settings</b> area.

## SETTINGS

Parameters	Description
<b>Settings</b>	Settings for selected column.
<b>Text</b>	Name for column title. The column title is online language switchable. To do this, the @ character must be entered in front of the name.
<b>Width</b>	Width of the column in characters. Calculation: Number time average character width of the selected font.
<b>Alignment</b>	Alignment. Selection by means of radio buttons. Possible settings: <ul style="list-style-type: none"> <li>▶ <b>Left-justified</b>: Text is justified on the left edge of the column.</li> <li>▶ <b>Centered</b>: Text is displayed centered in the column.</li> <li>▶ <b>Right-justified</b>: Text is justified on the right edge of the column.</li> </ul>
<b>Deactivate column filter in the Runtime</b>	Active: The filter for this column cannot be changed in Runtime. Note: Only available for: <ul style="list-style-type: none"> <li>▶ Batch Control</li> <li>▶ Extended Trend</li> <li>▶ Filter screens</li> <li>▶ Message Control</li> <li>▶ Recipegroup Manager</li> </ul>
<b>User defined colors</b>	Properties in order to define user-defined colors for text and background. The settings have an effect on the Editor and Runtime. Note: <ul style="list-style-type: none"> <li>▶ These settings are only available for configurable lists.</li> <li>▶ In addition, the respective focus in the list can be signaled in Runtime by means of different text and background colors. These are configured using the project properties.</li> </ul>
<b>User defined colors</b>	Active: User-defined colors are used.
<b>Text color</b>	Color for text display. Clicking on the color opens the color palette to select a color.

<b>Background color</b>	Color for the display of the cell background. Clicking on the color opens the color palette to select a color.
-------------------------	--

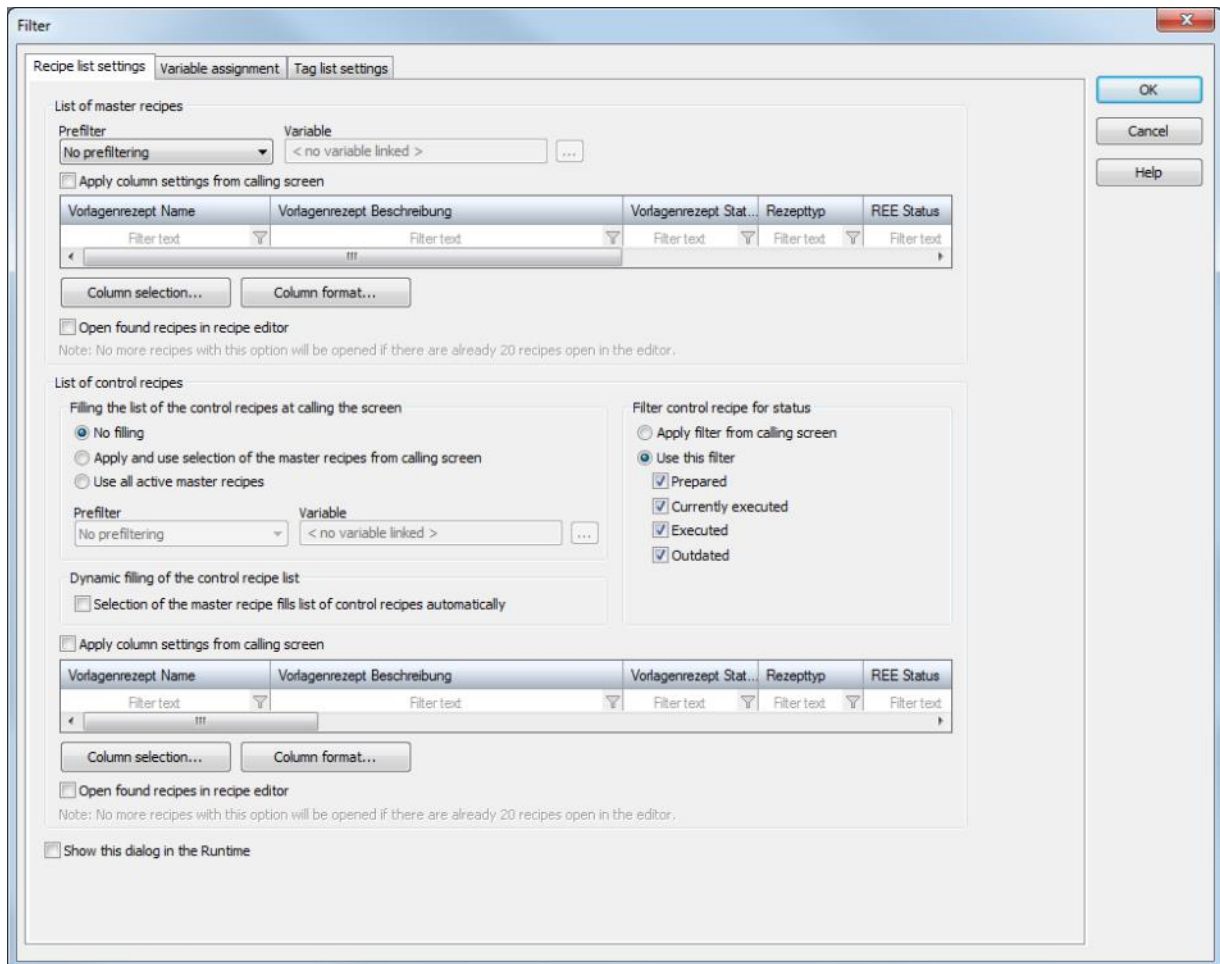
#### CLOSE DIALOG

Parameters	Description
<b>OK</b>	Applies all changes in all tabs and closes the dialog.
<b>Cancel</b>	Discards all changes in all tabs and closes the dialog.
<b>Help</b>	Opens online help.

**Note:** These settings are only used in Runtime for dockable windows (on page 118) if there is no Runtime profile available for the user who is logged in.

## Prefilter

To eliminate the need for all recipes to always be loaded in the recipe list in Runtime, you can define filters for master recipes and control recipes in the screen switch function. Then, the only recipes that appear in the list of master recipes and the control recipes are those that correspond to the configured filter conditions. If activated, these recipes are also opened in the recipe editor.



If you want it to be impossible for users to remove the filters in the runtime environment:

1. Deactivate the **Show this dialog in Runtime** option.
2. Block the column filter: Open the **Column format...** (on page 67) dialog and activate the **Block column filter in the process screen option**. As a result of this, the user cannot modify the filter in Runtime and therefore they do not get the recipes that they cannot modify displayed.

## FILTERING FOR MASTER RECIPES

Configure:

1. Prefilter

Stipulate if recipes are to be prefiltered. You can filter according to name or ID. The filter condition is queried in Runtime using a variable.

2. Column settings

- a) in the screen switch function click on column filters for the **List of the master recipes** or the **List of control recipes**
- b) enter the desired filter text; wildcards (\*) are allowed
- c) confirm the filter text with `Return` for it to be applied.

3. Automatic recipe switching

Stipulate if the recipes found when switching are also to be opened in the Recipe Editor straight away.

**Note:** The first 20 recipes found is the maximum that can be opened automatically.



### Information

When reloading Runtime, the filter settings of the prefiltering for master recipes are applied again. This also applies if the value of the filter variables changes or new recipes are added to the list that do not correspond to the filter. The list is always recreated exactly after reloading.

## FILTERING FOR CONTROL RECIPES

Configure:

1. Recipe state

Filter the control recipes according to their status. You can select several states using checkboxes or accept the filter from the calling screen.

2. Filling the recipe list

Stipulate if and how the list of control recipes is to be filled when called up.

3. Prefilter

Stipulate if recipes are to be prefiltered. You can filter according to ID, name or job ID. The filter condition is queried in Runtime using a variable

4. Column settings

- a) in the screen switch function click on column filters for the **List of the master recipes** or the **List of control recipes**
- b) enter the desired filter text; wildcards (\*) are allowed
- c) confirm the filter text with `Return` for it to be applied.

5. Automatic recipe switching



Stipulate if the recipes found when switching are also to be opened in the Recipe Editor straight away.

**Note:** The first 20 recipes found is the maximum that can be opened automatically.



### Information

*When reloading Runtime, the filter settings of the prefiltering for control recipes are applied again. Instead, all control recipes that correspond to the current filter in Runtime are displayed (master recipes, status, column filter).*

## 7.11.2 Variable assignment

You link variables to elements in the recipe in this tab. This way you can display the statuses of a phase or an operation in another screen and react to these. The execution status of the recipe and the selected object can be displayed by means of string variables or numerical variables. Numerical variables are suitable, for instance, for linking to a combined element. For details on the status, see the Coding of the execution status (on page 76) section.

To display statuses:

1. Create a new screen with the desired elements.
2. Link the elements to variables
3. Link these variables in the screen switching filter to the corresponding objects

### LINK VARIABLES

To link a variable:

1. Click on the ... button
2. The dialog for selecting a variable is opened
3. Select the desired variable

### DISPLAY

The following are displayed:

- For the recipe that is in focus:
  - Name
  - Description
  - Status

- Execution mode
- Execution status
- Recipe type
- For the object in the recipe that is in focus (phase or operation):
  - Name
  - Description
  - Unit
  - Type
  - Status
  - Internal state
  - Start time
  - Time of the end
  - Execution counter
  - Duration of execution

Filter...

Recipe list settings Variable assignment Tag list settings

Master recipe		Control recipe	
Name	< no variable linked > ...	Name	< no variable linked > ...
Version	< no variable linked > ...	Description	< no variable linked > ...
Initial version	< no variable linked > ...	Status (text)	< no variable linked > ...
Description	< no variable linked > ...	Status (numerical)	< no variable linked > ...
Status (text)	< no variable linked > ...	Job ID (text)	< no variable linked > ...
Status (numerical)	< no variable linked > ...		

Operation		General	
Name	< no variable linked > ...	Execution mode (text)	< no variable linked > ...
Description	< no variable linked > ...	Execution mode (numerical)	< no variable linked > ...
		Execution status (text)	< no variable linked > ...
		Execution status (numerical)	< no variable linked > ...
		Recipe type (text)	< no variable linked > ...
		Recipe type (numerical)	< no variable linked > ...

Selected object			
Name	< no variable linked > ...	Status (text)	< no variable linked > ...
Description	< no variable linked > ...	Status (numerical)	< no variable linked > ...
Unit	< no variable linked > ...	Internal status (text)	< no variable linked > ...
Type (text)	< no variable linked > ...	Internal status (numerical)	< no variable linked > ...
Type (numerical)	< no variable linked > ...	Start time	< no variable linked > ...
control strategy	< no variable linked > ...	Time of the end	< no variable linked > ...
Description of control strategy	< no variable linked > ...	Execution counter	< no variable linked > ...
control strategy number	< no variable linked > ...	Duration of execution	< no variable linked > ...

OK Cancel Help

Parameters	Description
<b>Master recipe</b>	<p>Variable linkings for the master recipe (on page 78).</p> <p>Status is displayed if the recipe has the focus or a phase or operation is highlighted.</p> <p>The following can be linked:</p> <ul style="list-style-type: none"> <li>▶ Name</li> <li>▶ Version</li> <li>▶ Initial version</li> <li>▶ Description</li> <li>▶ Status (text)</li> <li>▶ Status (numeric)</li> </ul>
<b>Control recipe</b>	<p>Variable linkings for the control recipe. (on page 79)</p> <p>Status is displayed if the recipe has the focus or a phase or operation is highlighted.</p> <p>The following can be linked:</p> <ul style="list-style-type: none"> <li>▶ Name</li> <li>▶ Description</li> <li>▶ Status (text)</li> <li>▶ Status (numeric)</li> <li>▶ Job ID (text)</li> </ul>
<b>Operation</b>	<p>Variable linkings for the operation. (on page 80)</p> <p>Status is displayed if the recipe has the focus or a phase or operation is highlighted.</p> <p>The following can be linked:</p> <ul style="list-style-type: none"> <li>▶ Name</li> <li>▶ Description</li> </ul>
<b>General</b>	<p>Variable linkings for general information. (on page 80)</p> <p>Status is displayed if the recipe has the focus or a phase or operation is highlighted.</p> <p>The following can be linked:</p> <ul style="list-style-type: none"> <li>▶ Execution mode (text)</li> <li>▶ Execution mode (numerical)</li> <li>▶ Execution status (text)</li> <li>▶ Execution status (numerical)</li> <li>▶ Recipe type (text)</li> </ul>

	<ul style="list-style-type: none"> <li>▶ Recipe type (numeric)</li> </ul>
<b>Selected object</b>	<p>Variable linkings for the selected object (phase or operation) (on page 83). Status is displayed if a phase or an operation in the recipe is highlighted.</p> <p>The following can be linked:</p> <ul style="list-style-type: none"> <li>▶ Name</li> <li>▶ Description</li> <li>▶ Unit</li> <li>▶ Type (text)</li> <li>▶ Type (numeric)</li> <li>▶ Control strategy</li> <li>▶ Description of control strategy</li> <li>▶ control strategy number</li> <li>▶ Status (text)</li> <li>▶ Status (numeric)</li> <li>▶ Internal state (text)</li> <li>▶ Internal state (numeric)</li> <li>▶ Start time</li> <li>▶ Time of the end</li> <li>▶ Execution counter</li> <li>▶ Duration of execution</li> </ul>

#### CLOSE DIALOG

Parameters	Description
<b>OK</b>	Applies all changes in all tabs and closes the dialog.
<b>Cancel</b>	Discards all changes in all tabs and closes the dialog.
<b>Help</b>	Opens online help.

**Note:** No information is displayed if several objects are highlighted.

#### Coding of the execution status

#### EXECUTION STATE

The execution state (both for the recipe and the selected object) is coded with different information:

- ▶ Byte 0: Execution state

- ▶ Byte 1: Status bits for the status
- ▶ Byte 2: Type of object that is part of the status

If no recipe is opened then the string variables are empty and the numerical variables have the value 0.

## BYTE 0

Execution states.

The values that are possible are determined by the object type.

String	Decimal	Binary
Idle	1	1
Running	2	10
Executed	3	11
Pause (starting from state: running)	4	100
Paused	5	101
Hold (starting from state: running)	6	110
Hold (starting from state: Paused)	7	111
Hold (starting from state: Restart)	8	1000
Held	9	1001
Restarting (starting from state: Held)	10	1010
Stopping (starting from state: running)	11	1011
Stopping (starting from state: Paused)	12	1100
Stopping (starting from state: Held)	13	1101
Stopped	14	1110
Aborting (starting from state: running)	15	1111
Aborting (starting from state: Paused)	16	10000
Aborting (starting from state: Held)	17	10001
Aborted	18	10010

## BYTE 1

Status bits for the status.

The text is added to the string variable. The corresponding bits are set in the numerical variables

Value for string variable	Value for numeric variable
n elements in different states	0b00000001
Waiting for n element(s)	0b00000010

**BYTE 2**

Type of object that is part of the status.  
For numerical variables only.

Decimal	Binary	Meaning
1	1	Recipe
2	10	Phase
3	11	Operation object in the recipe

**Master recipe****NAME**

Name of the currently-opened master recipe or the master recipe that belongs to the currently-opened control recipe or operation.

**DESCRIPTION**

Description of the currently-opened master recipe or the master recipe that belongs to the currently-opened control recipe or operation.

**STATUS (TEXT AND NUMERIC)**

Status of the currently-opened master recipe or the master recipe that belongs to the currently-opened control recipe or operation.

Variable values:

String	Decimal	Binary
No recipe in active view	0	0
Creation (not visible)	1	1
Edit mode	2	10
Released	3	11
Test mode	4	100
Test in execution	5	101
Terminated with error	6	110

## Control recipe

### NAME

Name or selection of the control recipe. The recipe must be open and have the focus. Is filled in the control recipe and in the operation instance in the control recipe.

### DESCRIPTION

Description of the currently-selected control recipe. The recipe must be open and have the focus. Is filled in the control recipe and in the operation instance in the control recipe.

### STATUS (TEXT AND NUMERIC)

Status of the currently-selected control recipe or the recipe must be open and have the focus. Is filled in the control recipe and in the operation instance in the control recipe.

Variable values:

String	Decimal	Binary
no control recipe active	0	0
Creation (not visible)	1	1
Prepared	2	10
Running	3	11
Executed	4	100
Terminated with error	5	101

## Operation

### NAME

Name of the operation that is currently open. Is filled in the operation template and in the operation instance.

### DESCRIPTION

Description of the operation currently open. Is filled in the operation template and in the operation instance.

## General

### EXECUTION MODE (TEXT AND NUMERIC)

Currently-set execution mode for the currently-selected recipe. Is filled for all recipe cycles.

Variable values:



String	Decimal	Binary
Automatic	1	1
Semi-automatic	2	10
Manual	3	11

## EXECUTION STATUS (TEXT AND NUMERIC)

Status of the recipe

### BYTE 0

Variable values:

String	Decimal	Binary
Idle	1	1
Running	2	10
Executed	3	11
Pausing (starting from state: running)	4	100
Paused	5	101
Holding (starting from state: running)	6	110
Held	9	1001
Restarting (starting from state: Held)	10	1010
Stopping (starting from state: running)	11	1011
Stopped	14	1110
Aborting (starting from state: running)	15	1111
Aborted	18	10010

**BYTE 1**

Value for string variable	Value for numeric variable
n elements in different states	0b00000001
Waiting for n element(s)	0b00000010

**BYTE 2**

Object type. For the numerical value only

Decimal	Binary	Meaning
1	1	Recipe

**RECIPE TYPE (TEXT AND NUMERIC)**

Recipe type of the current recipe. Is filled for all recipe cycles.

Variable values:

String	Decimal	Binary
Master recipe	1	1
Control recipe	2	10
Operation template	4	100
Operation instance in the master recipe	9	1001
Operation instance in the control recipe	10	1010

### Selected object

The variables for the selected object always contain data if a single phase or a single operation was selected in the currently-selected operation. It is filled for all recipe types. If no object or several objects are selected, then the string variables are empty and the numerical values are 0.

### NAME

Is filled with the name of the phase or the operation.

### DESCRIPTION

Is filled with the description of the phase of the operation.

### UNIT

Is filled with the name of the unit of the selected phase.

### TYPE (TEXT AND NUMERIC)

Variable values:

String	Decimal	Binary
Phase	3	11
Operation	13	1101

**STATUS (TEXT AND NUMERIC)**

Is filled with the current execution status of the element.

**VALUE FOR A PHASE****BYTE 0**

Variable values:

String	Decimal	Binary
Idle	1	1
Running	2	10
Executed	3	11
Pausing (Starting from state: running)	4	100
Paused	5	101
Holding (Starting from state: running)	6	110
Holding (Starting from state: Paused)	7	111
Holding (Starting from state: Restarting)	8	1000
Held	9	1001
Restarting (Starting from state: Held)	10	1010
Stopping (Starting from state: running)	11	1011
Stopping (Starting from state: Paused)	12	1100
Stopping (Starting from state: Held)	13	1101
Stopped	14	1110
Aborting (Starting from state: running)	15	1111
Aborting (Starting from state: Paused)	16	10000
Aborting (Starting from state: Held)	17	10001
Aborted	18	10010

**BYTE 1:**

Always empty.

**BYTE 2:**

Object type. For the numerical value only

Variable values:

Decimal	Binary	Meaning
2	10	Phase

**VALUE FOR AN OPERATION INSTANCE**

**BYTE 0**

Variable values:

String	Decimal	Binary
Idle	1	1
Running	2	10
Executed	3	11
Pausing (Starting from state: running)	4	100
Paused	5	101
Holding (Starting from state: running)	6	110
Held	9	1001
Restarting (Starting from state: Held)	10	1010
Stopping (Starting from state: running)	11	1011
Stopping (Starting from state: Paused)	12	1100
Stopping (Starting from state: Held)	13	1101
Stopped	14	1110
Aborting (Starting from state: running)	15	1111
Aborting (Starting from state: Paused)	16	10000
Aborting (Starting from state: Held)	17	10001
Aborted	18	10010

**BYTE 1**

Value for string variable	Value for numeric variable
n elements in different states	0b00000001

The bit is always set if there are objects in the operation with a different status to that of the operation. `Idle` and `finished` are not included in this.

#### BYTE 2:

Object type. For the numerical value only.

Variable values:

Decimal	Binary	Meaning
3	11	Operation in the recipe.

#### INTERNAL STATUS (TEXT AND NUMERIC)

Is filled with the internal execution status of the element.

Composition of the internal status:

- ▶ Byte 0: Status
- ▶ Byte 1: Status Bits
- ▶ Byte 2: Object type

#### VALUE FOR A PHASE

##### BYTE 0

Internal status. The text can be different for other objects.

Variable values:



String	Decimal	Binary
Idle	1	1
Waiting for phase to be ready for starting	2	10
Waiting for unit allocation	3	11
Waiting for the unit allocation - timeout	4	100
Waiting for exclusive execution	5	101
Waiting for input interlocking	6	110
Waiting for the input interlocking - timeout	7	111
Waiting for phase done condition	8	1000
Waiting for phase done condition - timeout	9	1001
Waiting for phase done condition - error writing value	10	1010
Waiting for minimum execution period	11	1011
Wait until recipe has status "running"	12	1100
Waiting for following conditions	13	1101
Waiting for following condition - timeout	14	1110

## BYTE 1

Status. For numerical variables only.

Variable values:

Decimal	Binary	Meaning
00000001	1	There is an execution error.
00001000	8	There is a communication error.
00010000	16	Loss of communication fixed.
01000000	64	Communication reestablished.

With the phase, only one of the bits can be active for the communication error. With an operation a bit is always set if it is relevant to at least one internal phase.

**BYTE 2**

Object type. For the numerical value only.

Variable values:

Decimal	Binary	Meaning
3	11	Operation in the recipe.

**POSSIBILITIES FOR PHASE****BYTE 0**

All possibilities.

**BYTE 1**

All possibilities.

**BYTE 2**

Value for numeric variable		Meaning
Decimal	Binary	
2	10	Phase

**POSSIBILITIES FOR OPERATION INSTANCES****BYTE 0**

Variable values:

String	Decimal	Binary
Idle	1	1
Execution of the internal objects	8	1000
Waiting for following conditions	13	1101

**BYTE 1**

Always empty.

**BYTE 2**

Variable values:

Decimal	Binary	Meaning
3	11	Operation in the recipe.

**START TIME**

Time at which the execution of the selected object has started.

**TIME OF THE END**

Time at which the execution of the selected object has finished.

**DURATION OF EXECUTION**

Time period that has expired during the execution.

**EXECUTION COUNTER**

Number denoting how often the element was executed

### 7.11.3 Tag list settings

Two parameter lists with identical options are available. These can be configured and used individually.

The parameter lists are created in the screen and configured in screen switching.

Filter...

Recipe list settings | Variable assignment | Tag list settings

Tag list 1

Name	Description	Type	Tag data type	Variable	Tag
Filter text	Filter text	Filter text	Filter text	Filter text	Filter text

< >

Column selection... Column format...

Display command TAG

☒ Changeable in master recipe ☒ Cannot be changed in the master recipe ☒ Changeable in the current recipe type

☒ Changeable in the control recipe ☒ Cannot be changed in the control recipe ☒ Cannot be changed in the current recipe type

☐ Display return TAG

Variable for displaying the number of entries

<no variable linked> ...

Tag list 2

Name	Description	Type	Tag data type	Variable	Tag
Filter text	Filter text	Filter text	Filter text	Filter text	Filter text

< >

Column selection... Column format...

Display command TAG

☒ Changeable in master recipe ☒ Changeable in the control recipe ☒ Changeable in the current recipe type

☒ Cannot be changed in the master recipe ☒ Cannot be changed in the control recipe ☒ Cannot be changed in the current recipe type

☐ Display return TAG

Variable for displaying the number of entries

<no variable linked> ...

OK Cancel Help

## TAG LIST 1

Parameters	Description
<b>Tag List 1</b>	<p>Display of the configured lists. The columns can:</p> <ul style="list-style-type: none"> <li>▶ be filtered</li> <li>▶ have their width changed by dragging the column title with the mouse</li> <li>▶ arranged by dragging &amp; dropping with the mouse</li> </ul>
<b>Column selection</b>	A dialog for choosing the columns which shall be displayed is opened
<b>Column Format</b>	Opens a dialog to format the columns.
<b>Display command tag</b>	<p>Selection of the command parameters which should be displayed. Possible selection by activating the checkboxes:</p> <ul style="list-style-type: none"> <li>▶ <b>Changeable in the master recipe:</b> Only command parameters for which the <b>changeable in the recipe</b> option has been set in the Editor are displayed.</li> <li>▶ <b>Changeable in the control recipe:</b> Only command parameters for which the <b>changeable in the recipe</b> option has been set in the master recipe are displayed.</li> <li>▶ <b>Changeable in the current recipe type:</b> Only command parameters that can be changed in the current recipe type are displayed.</li> <li>▶ <b>Cannot be changed in the master recipe:</b> Only command parameters for which the <b>Changeable in the recipe</b> option has not been set in the Editor are displayed.</li> <li>▶ <b>Cannot be changed in the control recipe:</b> Only command parameters for which the <b>Changeable in the recipe option</b> has not been set in the master recipe are displayed.</li> <li>▶ <b>Cannot be changed in the current recipe type:</b> Only command parameters that cannot be changed in the current recipe type are displayed.</li> </ul> <p>As many connections as desired can be configured. Activating all checkboxes leads to all command parameters being displayed.</p>
<b>Display return tags</b>	Active: Return parameters are displayed:
<b>Variable for displaying the number of entries</b>	<p>Displays the maximum number of entries that is possible with the current pre-filtering of the screen filter. It is independent of the filtering in the columns and groupings.</p> <p>A click on button ... opens the dialog for selecting variables.</p>

## TAG LIST 2

Parameters	Description
<b>Tag List 2</b>	Parameter list 2 with identical configuration possibilities to <b>Parameter Tag 1</b> .

#### CLOSE DIALOG

Parameters	Description
<b>OK</b>	Applies all changes in all tabs and closes the dialog.
<b>Cancel</b>	Discards all changes in all tabs and closes the dialog.
<b>Help</b>	Opens online help.

The tag lists are recreated in Runtime if:

- ▶ The phase was edited
- ▶ The recipe was saved (even if the client has saved it)



#### Information

*Backward compatibility for **displaying command parameters**:*

*The option was expanded with version 7.11 and changed from radio buttons to checkboxes. For backward compatibility, this means:*

*Backward compatible writing is possible if:*

*Only one checkbox is set for the changeable parameters*

*or the combination of all checkboxes lets all parameters through*

*If the combination of the checkboxes results in a setting that was not previously configurable with zenon 7.10 or earlier, no parameters are displayed in the list.*

## 7.12 zenon functions

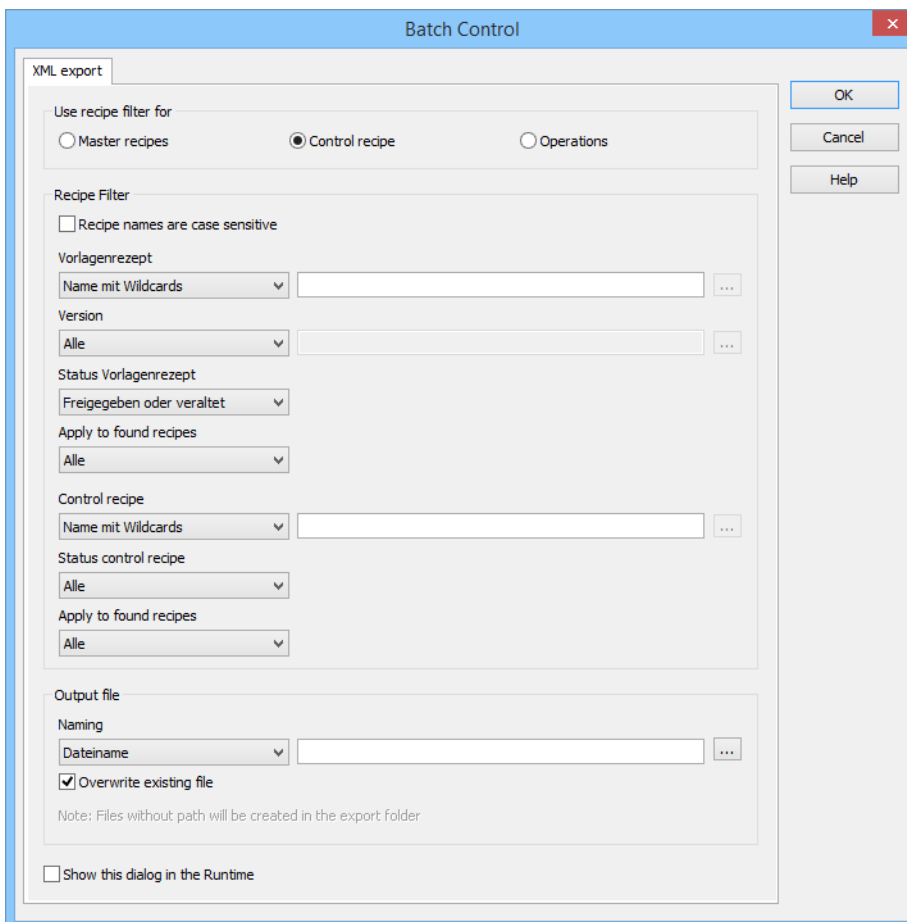
With zenon functions, control commands can be sent to the batch execution and pre-defined control recipes can be created:

- ▶ Execute recipe command or mode switch (on page 102): sends control commands to batch execution
- ▶ Create control recipe function (on page 107): creates, in Runtime, a recipe that has been pre-defined in the Editor
- ▶ Export batch recipes (on page 95): exports recipes as an XML file.
- ▶ Import batch recipes (on page 100): imports recipes as an XML file.

## 7.12.1 Export Batch recipes

Recipes can be exported to an XML file with the **Export Batch recipes** function. To create the function:

1. In the zenon Editor, navigate to the **Functions** node
2. Select **New function**
3. Go to the **Batch Control** in the function selections
4. select **Export Batch recipes**
5. the dialog for configuring functions is opened



The screenshot shows the 'Batch Control' dialog box with the 'XML export' tab selected. The dialog is titled 'Batch Control' and has a close button (X) in the top right corner. It contains several sections for configuring the export process:

- Use recipe filter for:** Three radio buttons are present: 'Master recipes' (unselected), 'Control recipe' (selected), and 'Operations' (unselected).
- Recipe Filter:** A section containing several options:
  - ☐ Recipe names are case sensitive
  - Vorlagenrezept:** A dropdown menu set to 'Name mit Wildcards' and an empty text field with a browse button (...).
  - Version:** A dropdown menu set to 'Alle' and an empty text field with a browse button (...).
  - Status Vorlagenrezept:** A dropdown menu set to 'Freigegeben oder veraltet'.
  - Apply to found recipes:** A dropdown menu set to 'Alle'.
  - Control recipe:** A dropdown menu set to 'Name mit Wildcards' and an empty text field with a browse button (...).
  - Status control recipe:** A dropdown menu set to 'Alle'.
  - Apply to found recipes:** A dropdown menu set to 'Alle'.
- Output file:** A section containing:
  - Naming:** A dropdown menu set to 'Dateiname' and an empty text field with a browse button (...).
  - ☒ Overwrite existing file
  - Note: Files without path will be created in the export folder
- ☐ Show this dialog in the Runtime

On the right side of the dialog, there are three buttons: 'OK', 'Cancel', and 'Help'.

## USE RECIPE FILTER FOR

Parameters	Description
<b>Use recipe filter for</b>	<p>Selection of what the recipe filter is applied to:</p> <ul style="list-style-type: none"> <li>▶ Master recipe</li> <li>▶ Control recipe</li> </ul> <p>The filter is processed from top to bottom For example, <b>version</b> is only applied to the recipes found in the <b>master recipe</b> filter.</p>
<b>Master recipe</b>	Active: It is filtered on Master recipes.
<b>Control recipe</b>	<p>Active: It is filtered on control recipes.</p> <p><b>Note:</b> The attendant master recipes must also be selected. If no master recipe has been selected for the control recipe, the filter cannot find the recipe being searched for in Runtime.</p> <p><b>Hint:</b> If the master recipe is not known, filtering of all master recipes with a placeholder is recommended.</p>

## RECIPE FILTER

Parameters	Description
<b>Recipe Filter</b>	Configuration of the recipe filter
<b>Master recipe</b>	<p>Parameters for the selection of the master recipe. Selection from drop-down list:</p> <ul style="list-style-type: none"> <li>▶ <b>Name with wildcards:</b> A name with placeholder can be entered into the input field. Filtering according to this name is carried out.</li> <li>▶ <b>Name from variable:</b> The name of the master recipe is defined by a variable in Runtime. A click on button ... opens the dialog for selecting variables.</li> </ul> <p><b>ID from variable:</b> The ID of the master recipe is defined by a variable in Runtime. A click on button ... opens the dialog for selecting variables.</p>
<b>Version</b>	<p>Selection of the version (on page 192) from the drop-down list:</p> <ul style="list-style-type: none"> <li>▶ All:</li> </ul>



	<p>The version stated is ignored and each version found is used.</p> <ul style="list-style-type: none"><li>▶ <code>Fixed version:</code> This filters for versions that are entered in this field. Highest possible version: 4294967295</li><li>▶ <code>Version from variable:</code> The recipe that was in the linked variables at the time of execution is filtered for. Click on button ... in order to open the dialog for selecting a variable.</li><li>▶ <code>Only oldest version:</code> Only the recipe with the oldest version number is used.</li></ul> <p><code>Only newest version:</code> Only the recipe with the newest version number is used.</p>
--	--

<b>State master recipe</b>	<p>Status of the master recipe. Select from drop-down list.</p> <p><u>When selecting master recipes for recipe filters:</u></p> <ul style="list-style-type: none"> <li>‣ All</li> <li>‣ Edit mode</li> <li>‣ Released</li> <li>‣ Test mode</li> <li>‣ Test running</li> <li>‣ Terminated with error</li> <li>‣ Outdated</li> </ul> <p><u>When selecting control recipes for recipe filters:</u></p> <ul style="list-style-type: none"> <li>‣ Released or outdated</li> <li>‣ Released</li> <li>‣ Outdated</li> </ul>
<b>Apply to found recipes</b>	<p>Definition of which master recipes the filter is applied to. Select from drop-down list:</p> <ul style="list-style-type: none"> <li>‣ All</li> <li>‣ Only oldest ID</li> <li>‣ Only newest ID</li> </ul>
<b>Control recipe</b>	<p>Parameters for the selection of the control recipe. Selection from drop-down list:</p> <ul style="list-style-type: none"> <li>‣ Name with wildcards: A name with placeholder can be entered into the input field. Filtering according to this name is carried out.</li> <li>‣ Name from variable:_ The name of the control recipe is defined by a variable in Runtime. A click on button ... opens the dialog for selecting variables.</li> <li>‣ ID from variable: The ID of the master recipe is defined by a variable in Runtime. A click on button ... opens the dialog for selecting variables. Precisely one recipe can be found if the variable value at the time of execution is a valid ID of a control recipe.</li> </ul> <p>Job ID from variable: Finds control recipes that belong to the master recipes already found and which have the given job ID. Any type of variable can be linked. The value is automatically converted into STRING.</p>

<b>State control recipe</b>	<p>Definition of which recipe status the filter is applied to. Select from drop-down list:</p> <ul style="list-style-type: none"> <li>▶ All</li> <li>▶ Prepared</li> <li>▶ Running</li> <li>▶ Executed</li> <li>▶ Terminated with error</li> <li>▶ Outdated</li> </ul>
<b>Apply to found recipes</b>	<p>Definition of which control recipes the filter is applied to. Select from drop-down list:</p> <ul style="list-style-type: none"> <li>▶ All</li> <li>▶ Only oldest ID</li> <li>▶ Only newest ID</li> </ul> <p><b>Note:</b> Only the respective IDs are taken into account for master recipes and control recipes. The search for control recipes can find several recipes with this filter. This filter must also be activated for the master recipes for a unique result.</p>
<b>Operation</b>	<p>Parameters for the selection of the operation. Select from drop-down list:</p> <ul style="list-style-type: none"> <li>▶ Name with wildcards: A name with placeholder can be entered into the input field. Filtering according to this name is carried out.</li> <li>▶ Name from variable:_ The name of the operation is defined by a variable in Runtime. Click on button ... opens the dialog for selecting variables.</li> <li>▶ ID from variable: The ID of the operation is defined by a variable in Runtime. Click on button ... opens the dialog for selecting variables. Precisely one recipe can be found if the variable value at the time of execution is a valid ID of an operation.</li> </ul>
<b>Status operation</b>	<p>Definition of which recipe status the filter is applied to. Select from drop-down list:</p> <ul style="list-style-type: none"> <li>▶ All</li> <li>▶ Edit mode</li> <li>▶ Released</li> <li>▶ Outdated</li> </ul>

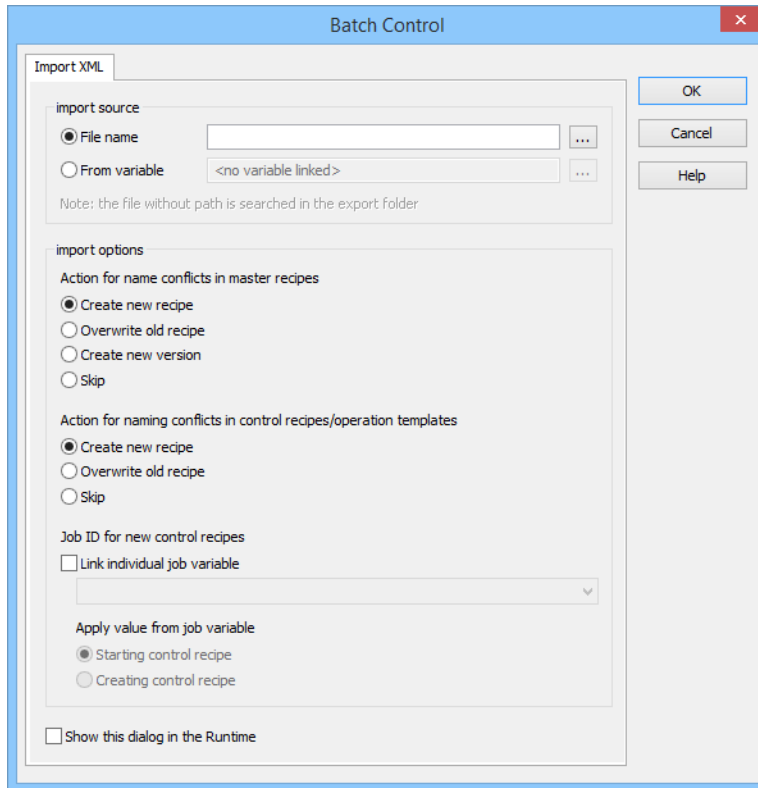
<b>Apply to found recipes</b>	Definition of which operations the filter is applied to. Select from drop-down list: <ul style="list-style-type: none"> <li>▶ All</li> <li>▶ Only oldest ID</li> <li>▶ Only newest ID</li> </ul>
<b>Output file</b>	Selection of the file name and the save location.
<b>Naming</b>	<ul style="list-style-type: none"> <li>▶ <b>File name:</b> The name of the XML file can be entered manually or when selecting a file from the respective save path.</li> <li>▶ <b>File name from variable:</b> The name of the XML file is taken from the linked variable.</li> <li>▶ <b>Create file name from recipe name:</b> The name of the XML file is made up of the recipe type, recipe name and recipe version.</li> <li>▶ <b>Create file name from recipe ID:</b> The name of the XML file is made up of the recipe type and recipe ID.</li> </ul>
<b>Overwrite existing file</b>	<b>Active:</b> If there are naming conflicts, the pre-existing XML file is overwritten with the names.
<b>Note: Files without a path will be created in the export folder</b>	Files without the path stated will be automatically created in the export folder.
<b>Show this dialog in the Runtime</b>	<b>Active:</b> The dialog is shown in Runtime and can be operated.
<b>OK</b>	Applies settings and closes the dialog.
<b>Cancel</b>	Discards all changes and closes the dialog.
<b>Help</b>	Opens online help.

## 7.12.2 Import Batch recipes

Recipes can be imported to an XML file with the **Import Batch recipes** function. To create the function:

1. In the zenon Editor, navigate to the **Functions** node
2. Select **New function**
3. Go to the **Batch Control** in the function selections

4. select **Import Batch recipes**
5. the dialog for configuring functions is opened



## IMPORT SOURCE

Parameters	Description
<b>File name</b>	Imports the XML file with the selected file name.
<b>From variable</b>	Imports the XML file from the linked variable.

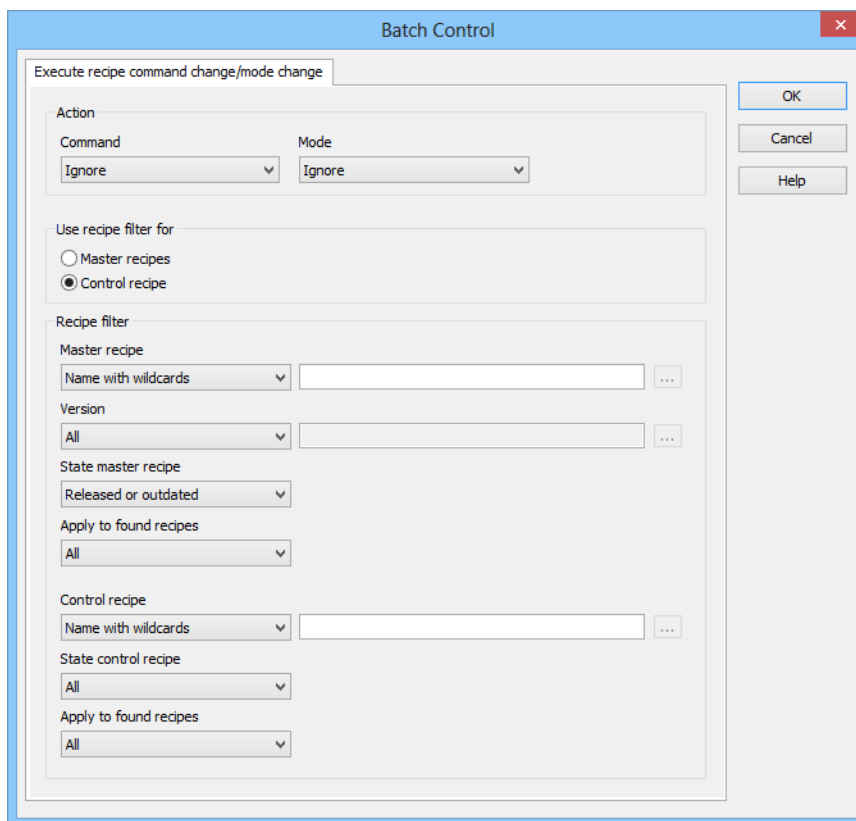
## IMPORT OPTIONS

Parameters	Description
<b>Action for name conflicts in master recipes</b>	Selection of a master recipe name that already exists.
Create new recipe	Creates a new recipe with the name.
Overwrite old recipe	Overwrites an existing recipe with the name.
Create new version	Creates a new version of the recipe.
Skip	Skips this step.
<b>Action in the event of naming conflicts in control recipes/operations</b>	Selection when issuing a control recipe or operation that already exists.
CREATE NEW RECIPE	Creates a new recipe with the name.
Overwrite old recipe	Overwrites an existing recipe with the name.
Create new version	Creates a new version of the recipe.
Skip	Skips this step.
<b>Job ID for new control recipes</b>	Job ID for newly-created control recipe
<b>Link individual job variable</b>	Links an individual order variable for each control recipe
<b>Apply value from job variable in the event of</b>	The value from the job variable is taken with <ul style="list-style-type: none"> <li>▶ <b>Starting control recipe</b></li> <li>▶ <b>Creating control recipe</b></li> </ul>
<b>Show this dialog in the Runtime</b>	This dialog can also be operated in Runtime.

### 7.12.3 Execute recipe command change or mode change

You can send control commands to the batch execution with this function. To create the function:

1. In the zenon Editor, navigate to the **Functions** node
2. Select **New function**
3. Go to the **Batch Control** in the function selections
4. select **Execute recipe command/change mode**
5. the dialog for configuring functions is opened



**ACTION**

Parameters	Description
<b>Action</b>	Selection of the action to be executed: <ul style="list-style-type: none"> <li>▸ Command</li> <li>▸ Mode</li> </ul>
<b>Command</b>	Selection of the command to be executed from drop-down list: <ul style="list-style-type: none"> <li>▸ Ignore</li> <li>▸ Start recipe</li> <li>▸ Recipe pausing</li> <li>▸ Recipe resuming</li> <li>▸ Recipe holding</li> <li>▸ Restart recipe</li> <li>▸ Recipe stopping</li> <li>▸ Recipe aborting</li> </ul>
<b>Mode</b>	Selection of the mode to which the recipe is to be switched: <ul style="list-style-type: none"> <li>▸ Ignore</li> <li>▸ Automatic</li> <li>▸ Semi-automatic</li> <li>▸ Manual</li> </ul>

**USE RECIPE FILTER FOR**

Parameters	Description
<b>Use recipe filter for</b>	Selection of what the recipe filter is applied to: <ul style="list-style-type: none"> <li>▸ Master recipe</li> <li>▸ Control recipe</li> </ul> <p>The filter is processed from top to bottom For example, <b>version</b> is only applied to the recipes found in the <b>master recipe</b> filter.</p>
<b>Master recipe</b>	Active: It is filtered on Master recipes.
<b>Control recipe</b>	Active: It is filtered on control recipes.  <b>Note:</b> The attendant master recipes must also be selected. If no master recipe has been selected for the control recipe, the filter cannot find the recipe being



	<p>searched for in Runtime.</p> <p>Hint: If the master recipe is not known, filtering of all master recipes with a placeholder is recommended.</p>
--	--

## RECIPE FILTER

Parameters	Description
<b>Recipe Filter</b>	Configuration of the recipe filter
<b>Master recipe</b>	<p>Parameters for the selection of the master recipe. Selection from drop-down list:</p> <ul style="list-style-type: none"> <li>▶ <code>Name with wildcards:</code> A name with placeholder can be entered into the input field. Filtering according to this name is carried out.</li> <li>▶ <code>Name from variable:</code> The name of the master recipe is defined by a variable in Runtime. A click on button ... opens the dialog for selecting variables.</li> <li>▶ <code>ID from variable:</code> The ID of the master recipe is defined by a variable in Runtime. A click on button ... opens the dialog for selecting variables.</li> </ul>
<b>Version</b>	<p>Selection of the version (on page 192) from the drop-down list:</p> <ul style="list-style-type: none"> <li>▶ <code>All:</code> The version stated is ignored and each version found is used.</li> <li>▶ <code>Fixed version:</code> This filters for versions that are entered in this field. Highest possible version: 4294967295</li> <li>▶ <code>Version from variable:</code> The recipe that was in the linked variables at the time of execution is filtered for. Click on button ... in order to open the dialog for selecting a variable.</li> <li>▶ <code>Only oldest version:</code> Only the recipe with the oldest version number is used.</li> <li>▶ <code>Only newest version:</code> Only the recipe with the newest version number is used.</li> </ul>

<b>State master recipe</b>	<p>Status of the master recipe. Select from drop-down list.</p> <p><u>When selecting master recipes for recipe filters:</u></p> <ul style="list-style-type: none"> <li>‣ All</li> <li>‣ Test mode</li> <li>‣ Test running</li> </ul> <p><u>When selecting control recipes for recipe filters:</u></p> <ul style="list-style-type: none"> <li>‣ Released or outdated</li> <li>‣ Released</li> <li>‣ Outdated</li> </ul>
<b>Apply to found recipes</b>	<p>Definition of which master recipes the filter is applied to. Select from drop-down list:</p> <ul style="list-style-type: none"> <li>‣ All</li> <li>‣ Only oldest ID</li> <li>‣ Only newest ID</li> </ul>
<b>Control recipe</b>	<p>Parameters for the selection of the control recipe. Selection from drop-down list:</p> <ul style="list-style-type: none"> <li>‣ Name with wildcards: A name with placeholder can be entered into the input field. Filtering according to this name is carried out.</li> <li>‣ Name from variable:_ The name of the control recipe is defined by a variable in Runtime. A click on button ... opens the dialog for selecting variables.</li> <li>‣ ID from variable: The ID of the master recipe is defined by a variable in Runtime. A click on button ... opens the dialog for selecting variables. Precisely one recipe can be found if the variable value at the time of execution is a valid ID of a control recipe.</li> <li>‣ Job ID from variable: Finds control recipes that belong to the master recipes already found and which have the given job ID. Any type of variable can be linked. The value is automatically converted into STRING.</li> </ul>
<b>State control recipe</b>	<p>Definition of which recipe status the filter is applied to. Select from drop-down list:</p> <ul style="list-style-type: none"> <li>‣ All</li> <li>‣ Prepared</li> <li>‣ Running</li> </ul>

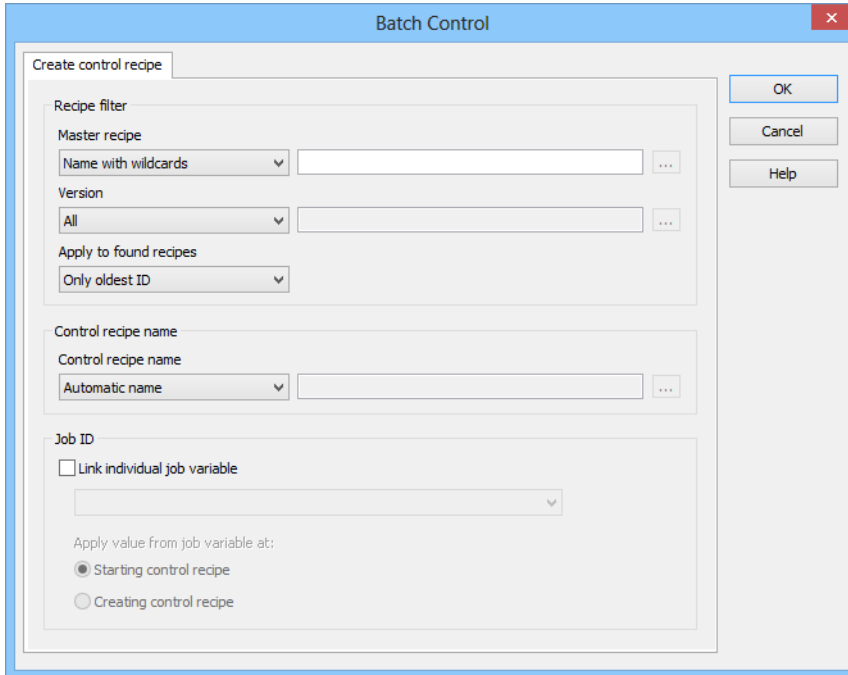
<b>Apply to found recipes</b>	<p>Definition of which control recipes the filter is applied to. Select from drop-down list:</p> <ul style="list-style-type: none"> <li>‣ All</li> <li>‣ Only oldest ID</li> <li>‣ Only newest ID</li> </ul> <p><b>Note:</b> Only the respective IDs are taken into account for master recipes and control recipes. The search for control recipes can find several recipes with this filter. This filter must also be activated for the master recipes for a unique result.</p>
<b>OK</b>	Applies settings and closes the dialog.
<b>Cancel</b>	Discards all changes and closes the dialog.
<b>Help</b>	Opens online help.

#### 7.12.4 Create control recipe function

With the help of the function **Create control recipe**, a pre-defined control recipe can be created in the Editor by means of a button in Runtime. To create the function:

1. In the zenon Editor, navigate to the **Functions** node
2. Select **New function**
3. Go to the **Batch Control** in the function selections
4. select **Create control recipe**

5. the dialog for configuring functions is opened



The image shows a screenshot of the "Batch Control" dialog box, which is used for configuring functions. The dialog has a blue title bar with the text "Batch Control" and a close button (X) in the top right corner. The main content area is divided into several sections:

- Create control recipe**: This section contains a "Recipe filter" group box with three sub-sections:
  - Master recipe**: A dropdown menu set to "Name with wildcards" and an empty text field with a browse button (...).
  - Version**: A dropdown menu set to "All" and an empty text field with a browse button (...).
  - Apply to found recipes**: A dropdown menu set to "Only oldest ID".
- Control recipe name**: A group box containing a "Control recipe name" dropdown menu set to "Automatic name" and an empty text field with a browse button (...).
- Job ID**: A group box containing a checkbox labeled "Link individual job variable". If checked, it would reveal a dropdown menu. Below this is a section labeled "Apply value from job variable at:" with two radio buttons: "Starting control recipe" (which is selected) and "Creating control recipe".

On the right side of the dialog, there are three buttons: "OK", "Cancel", and "Help".

## RECIPE FILTER

Parameters	Description
<b>Recipe Filter</b>	<p>Configuration of the recipe filter</p> <p>The filter is processed from top to bottom For example, <b>version</b> is only applied to the recipes found in the <b>master recipe</b> filter.</p>
<b>Master recipe</b>	<p>Parameters for the selection of the master recipe. Selection from drop-down list:</p> <ul style="list-style-type: none"> <li>▶ <code>Name with wildcards:</code> A name with placeholder can be entered into the input field. Filtering according to this name is carried out.</li> <li>▶ <code>Name from variable:</code> The name of the master recipe is defined by a variable in Runtime. A click on button ... opens the dialog for selecting variables.</li> <li>▶ <code>ID from variable:</code> The ID of the master recipe is defined by a variable in Runtime. A click on button ... opens the dialog for selecting variables.</li> </ul>
<b>Version</b>	<p>Selection of the version (on page 192) from the drop-down list:</p> <ul style="list-style-type: none"> <li>▶ <code>All:</code> The version stated is ignored and each version found is used.</li> <li>▶ <code>Fixed version:</code> This filters for versions that are entered in this field. Highest possible version: 4294967295</li> <li>▶ <code>Version from variable:</code> The recipe that was in the linked variables at the time of execution is filtered for. Click on button ... in order to open the dialog for selecting a variable.</li> <li>▶ <code>Only oldest version:</code> Only the recipe with the oldest version number is used.</li> <li>▶ <code>Only newest version:</code> Only the recipe with the newest version number is used.</li> </ul>

<b>Apply to found recipes</b>	<p>Definition of which recipes the filter is applied to. Selection of ID from drop-down list:</p> <ul style="list-style-type: none"> <li>‣ All</li> <li>‣ Only oldest ID</li> <li>‣ Only newest ID</li> </ul>
-------------------------------	---

## CONTROL RECIPE NAME

Parameters	Description
<b>Control recipe name</b>	Configuration of the name of the control recipe.
<b>Control recipe name</b>	<p>Selection of the naming from the drop-down list:</p> <ul style="list-style-type: none"> <li>‣ Automatic name: Name is automatically issued on creation</li> <li>‣ Name from variable: Name is taken from a variable. Click on button ... in order to open the dialog for selecting a variable. If there is already a recipe with the name that has been transferred from the variable, no new control recipe is created.</li> </ul>

## JOB ID

Parameters	Description
<b>Job ID</b>	Configuration of the Job ID.
<b>Link individual job variable</b>	Active: A job variable (on page 16) can be linked. The variable must already be configured. Selection of the variable from the drop-down list:
<b>Apply value from job variable</b>	<p>Definition of the time at which the job ID is transferred. During:</p> <ul style="list-style-type: none"> <li>‣ Starting control recipe</li> <li>‣ Creating control recipe</li> </ul>
<b>OK</b>	Applies settings and closes the dialog.
<b>Cancel</b>	<p>Discards all changes and closes the dialog.</p> <p>The function is nevertheless created, however without a defined target.</p>
<b>Help</b>	Opens online help.

All filters always have an effect on a group of recipes with the same name. Depending on the configuration, more than one recipe can remain left over.

For example: Recipes with the **\*Test** filter are searched for. The result is 5 versions of **Test\_1** and 3

versions of **Test\_2**. If filtering for the latest version is continued, then two control recipes are created, one each for the recipe with the highest version number per group.

**Note for variable selection using name or ID:** For the selection of variables according to name or ID, numerical variables and string variables can be selected respectively. The data types are converted to the respective correct form.

## 7.13 Replace links

Linking of variables and functions can be replaced automatically in units, phases and reactions. This process corresponds to the process for **replacing linking for screen switching** and **replacing linking in the Editor screen**. The replacement can

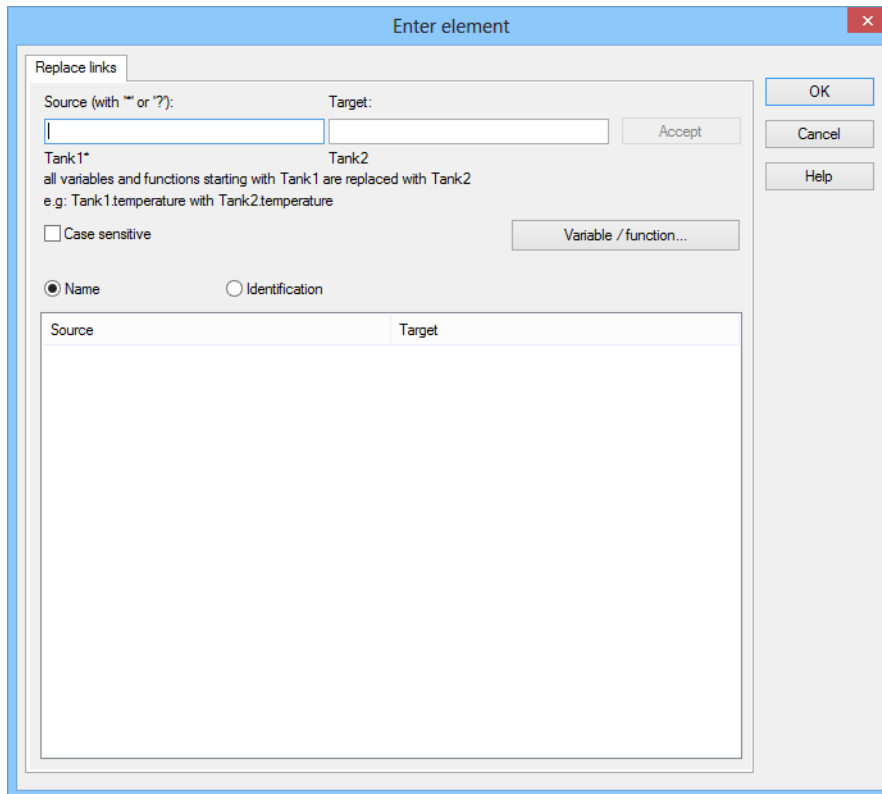
The following can be replaced:

- ▶ Units: linked variables for Runtime information
- ▶ Reactions: linked functions
- ▶ Parameter: linked variables

To replace elements:

1. Select the **Replace linking** command in the context menu or the toolbar

2. The dialog for the replacement of links opens.





Property	Description
Source	Enter the partial string to be searched for.  Place holder * and ? can be used. Placeholders are only permitted as prefix or suffix; e.g. *xxx or xxx*.
Target	Entry of the partial string
Name	Swaps information in process variable names.
Identification	Exchanges information in the identification
Note capitalization	When swapping, be sure that any capitalization is an exact match.
Accept	Swaps target strings from the <b>source</b> for those defined in the <b>target</b> .
Variable/Function	Opens the selection list for variables/functions in relation to the selected line in the list. Clicking on the variable in the list defines new target variables. Alternative: Double-click on the source variable in question.

## REPLACE

### A) REPLACE BY MANUAL SELECTION

- ▶ select the element from the list that you would like as the source
- ▶ select a target element via the **variable/function** button
- ▶ the previous element is replaced by the new one

### B) AUTOMATED REPLACEMENT WITH RULES

- ▶ In the source input field, define the parameters for the element that you wish to replace
- ▶ define the parameter for the new variable/function in the target input field
- ▶ specify what is to be replaced via Name/Identification
- ▶ Click on **Accept**

**Information**

*The target variable or target function can also be in a different project as the source variable or source function. In doing so, all projects concerned must be started and available on the same computer in Runtime.*

**Information**

*Internal variable IDs are used for replacement. This means that if variables are used or functions are renamed, the replacement remains.*

**Attention**

*When replacing variables, be aware of the type and signal resolution. If you replace a variable with one of an incompatible type, this can lead to errors during execution. You will be warned when making the substitution; the substitution will however be carried out.*

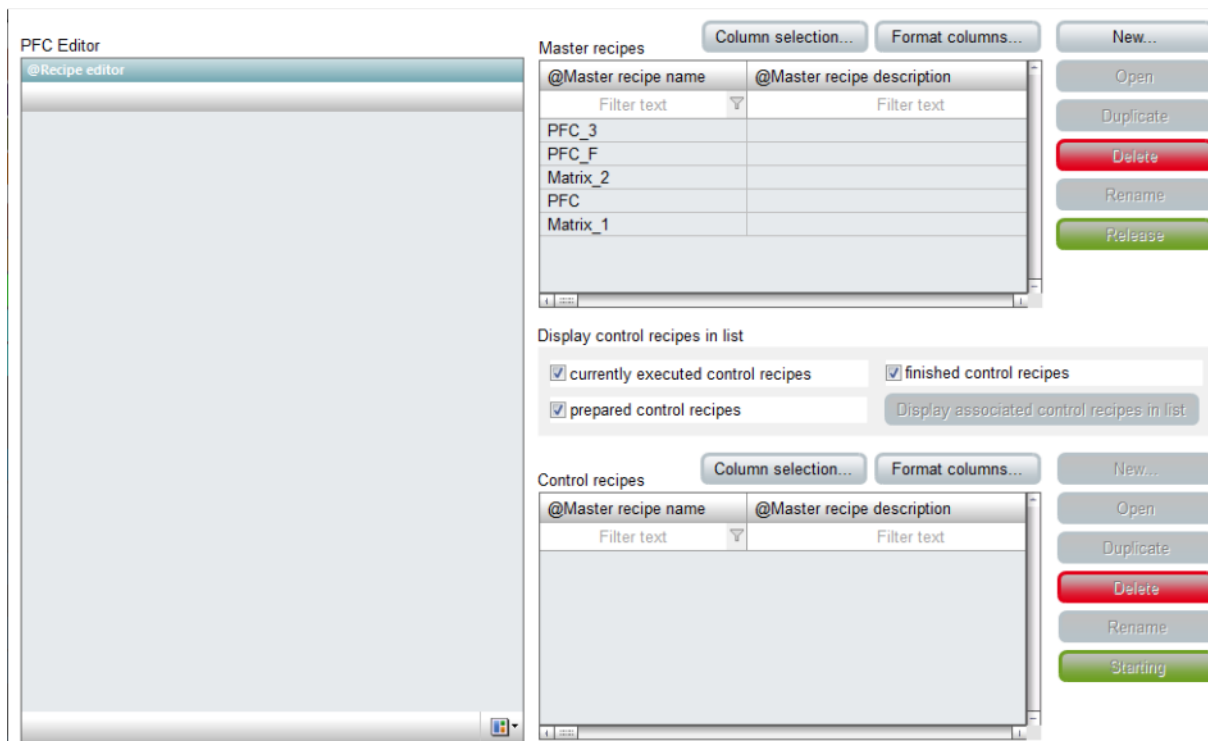
## 8. Conversion

If the version of zenon is changed, all recipes must be ended before converting the project.

Recipes that are running continue to be executed after a restart. The restart only functions within the same zenon version.

## 9. Configure and control in the Runtime

The entire management, creation and processing of the recipes is done in the Runtime. Editing in the zenon Editor is not possible.



**Note:** Runtime files up to and including version 7.00 SP0 are not compatible with subsequent versions. Versions from version 7.10 and later are compatible.

### SYNCHRONIZATION

When loading, opening, duplicating and approving a recipe or operation, a check is made to see if the configuration of units, phases etc. has been changed in the superordinate instance, such as the Editor. For details, see the Synchronization (on page 212) chapter.

### SELECTION PROCEDURE IN LISTS:

- ▶ **Ctrl+A:** selects all elements
- ▶ **Ctrl+mouse click:** adds master recipe to the existing selection
- ▶ **Key Shift+mouse click:** Extends selection from the currently selected master recipe to the clicked master recipe

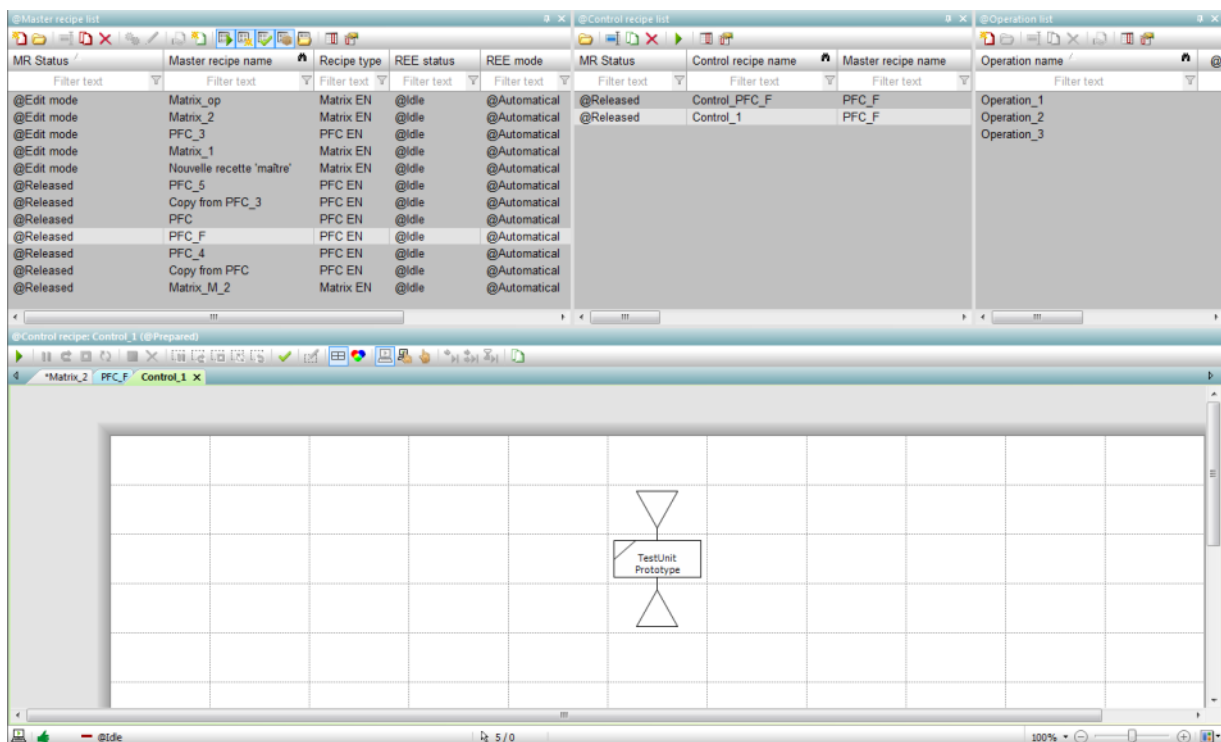
## SAVING OF RECIPES WHEN CLOSING RUNTIME

If Runtime is closed and there are still recipes that have not been saved, you are asked if these recipes are to be saved. In order for this query to not prevent Runtime closing, Runtime is automatically closed after 15 seconds if nothing is entered. Unsaved recipes are then discarded.

### 9.1 User interface

The user interface of the editor in Runtime can be configured with toolbars and dockable windows.

Example: User interface with list of the master recipes, list of the control recipe, list of the operation and PFC editor:



## SWITCHING POSITIONS

The position in the recipe is adapted depending on the recipe and status of the recipe:

- When opening a currently-running recipe:

It is centered on the first active element. The first active element is the one that is at the top. If there are several active elements in the same line, the element that is furthest to the left is

selected. Elements with execution positions before or after this are handled the same as the active ones in this case.

- ▶ When a recipe is first opened

Centering is on the start of the recipe. In a PFC recipe, centering is on the start element. A matrix recipe is opened in such a way that the upper left corner is visible.

### 9.1.1 Editor operating elements

The following are available for the editor:

- ▶ Toolbars (can be hidden)
- ▶ Tabs (on page 117)
- ▶ Dockable windows (on page 118)

#### TOOLBARS

There are independent toolbars available in the Editor for each type of recipe and the different status. For details, see the Matrix recipe toolbars (on page 184) and Toolbars and PFC recipe context menu (on page 148) chapters. All actions of the individual symbols can also be engineered using their own buttons in the screen. If the toolbars are thus not needed, they can be displayed or hidden using the settings in the zenon Editor.

To show/hide toolbars:

1. In the zenon Editor, highlight the **Recipe editor** comment area in the **Batch Control** screen
2. go to property group **Representation\Display editor control elements**
3. Activate or deactivate the **Toolbars** checkbox

#### Tabs in the Editor

If several recipes are open in the editor, these are represented with tabs. Recipes can be displayed and opened in two groups next to each other or underneath each other. To open a recipe in a new group:

1. Select, in the context menu of the recipe, the **horizontal neighboring group** or **vertical neighboring group** command
2. Select the type of display:
  - Move
  - Open in parallel

The control elements are always only applicable to the active recipe of the active group.

- ▶ The active group is emphasized in color.
- ▶ The active tab is emphasized with bold font. Information on the active tab is shown in the title bar of the editor.
- ▶ Tabs can be moved and arranged by dragging & dropping, including between groups.

## TAB CONTEXT MENU

Parameters	Description
<b>Save</b>	Saves changes in the recipe.
<b>Close</b>	Closes the recipe.
<b>Close all others</b>	Closes all other open recipes. Only the recipe in which the context menu was activated remains open.
<b>Group horizontally</b>	<p>Opens the recipe in a new group below the other recipes.</p> <ul style="list-style-type: none"> <li>▶ Move display: The recipe is moved and the upper group is removed.</li> <li>▶ Move display parallel. The recipe is displayed in parallel in both groups.</li> </ul>
<b>Group vertically</b>	<p>Opens the recipe in a new group to the right of the active group.</p> <ul style="list-style-type: none"> <li>▶ Move display: The recipe is moved and removed from the left group.</li> <li>▶ Move display parallel. The recipe is displayed in parallel in both groups.</li> </ul>

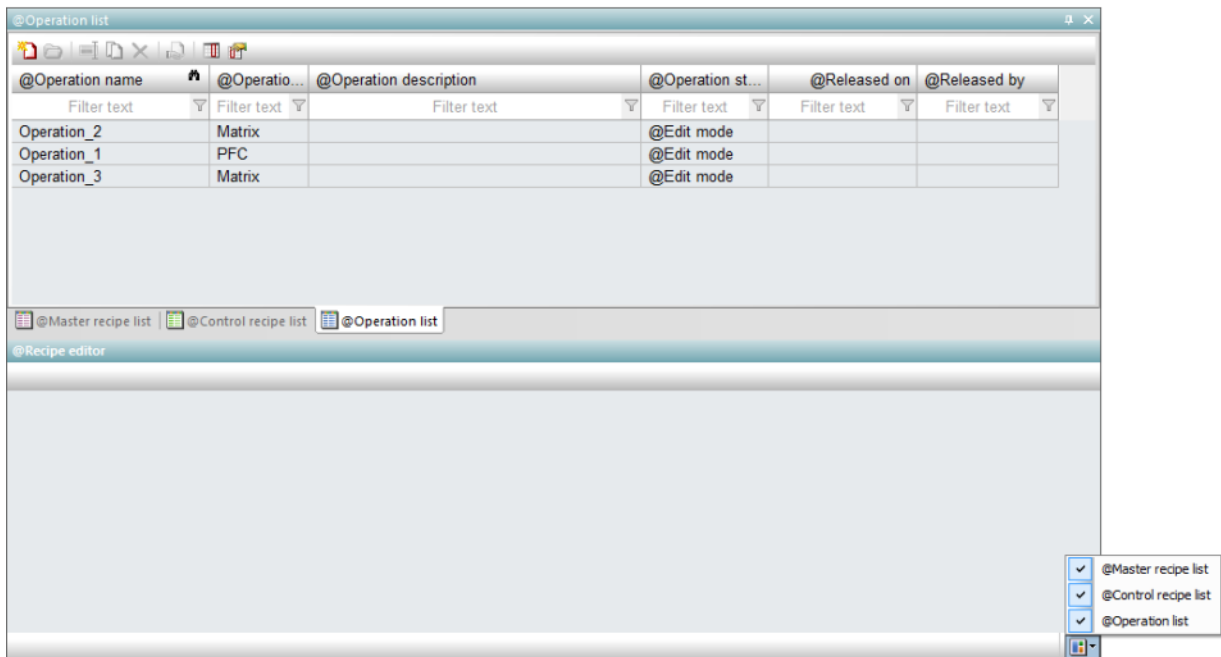
## Dockable windows

In the recipe editor, three windows can be shown, positioned and docked as desired:

- ▶ List of master recipes (on page 139)
- ▶ List of control recipes (on page 204)
- ▶ List of operations (on page 194)

For details on the selection and positioning, see the Selection and positioning (on page 121) chapter.

The settings are saved individually for each computer and user.



## SHOW/HIDE LIST

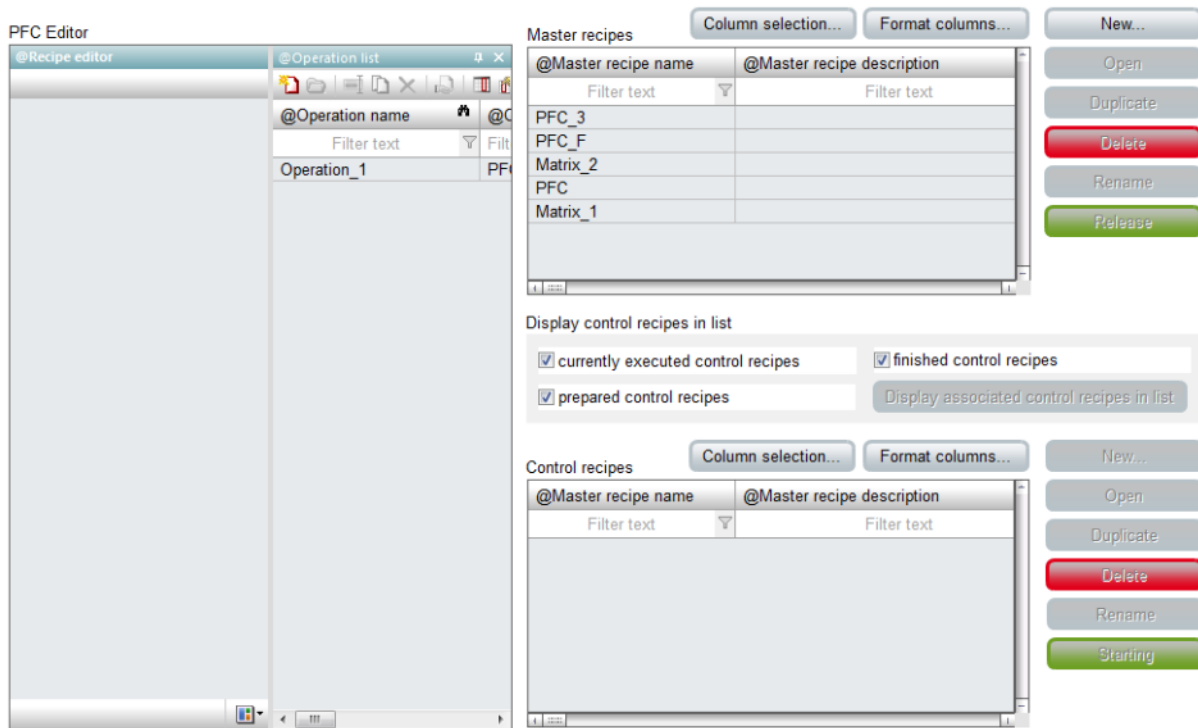
The list of dockable windows can be displayed or hidden by means of settings in the zenon Editor.

To show/hide the list:

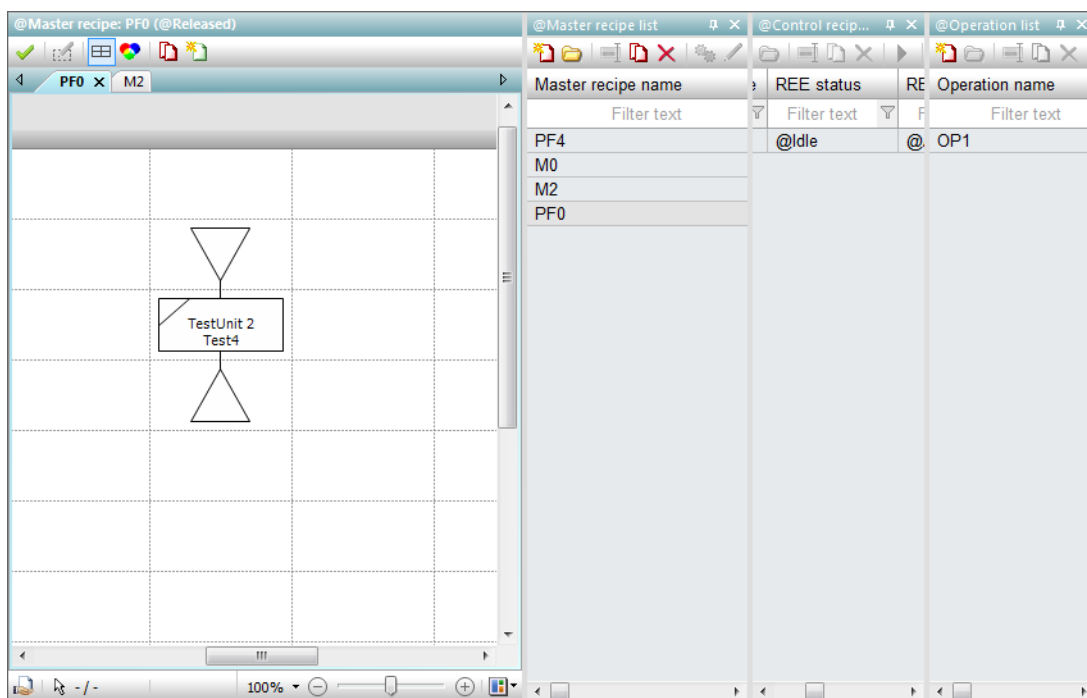
1. In the zenon Editor, highlight the **Recipe editor** comment area in the **Batch Control** screen
2. go to property group **Representation\Display editor control elements**
3. Activate or deactivate the **Dockable windows** checkbox

## EXAMPLES

Additional list of the operations shown in the Recipe Editor:

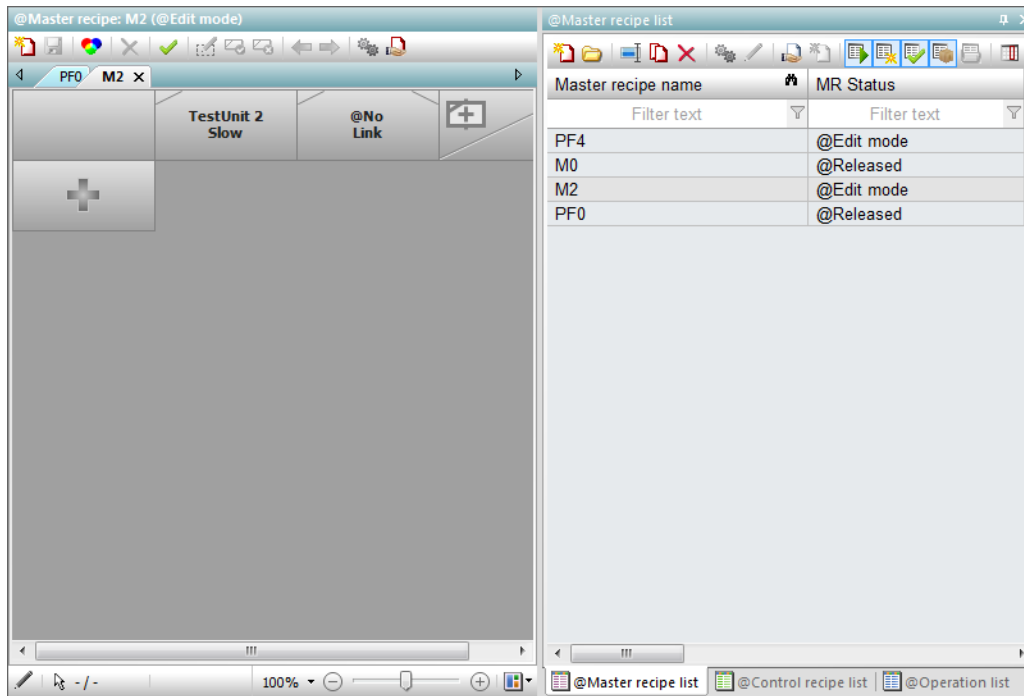


Additional list of master recipes, control recipes and operations shown in the Recipe Editor:





Lists of master recipes, control recipes and operations shown in the Recipe Editor:

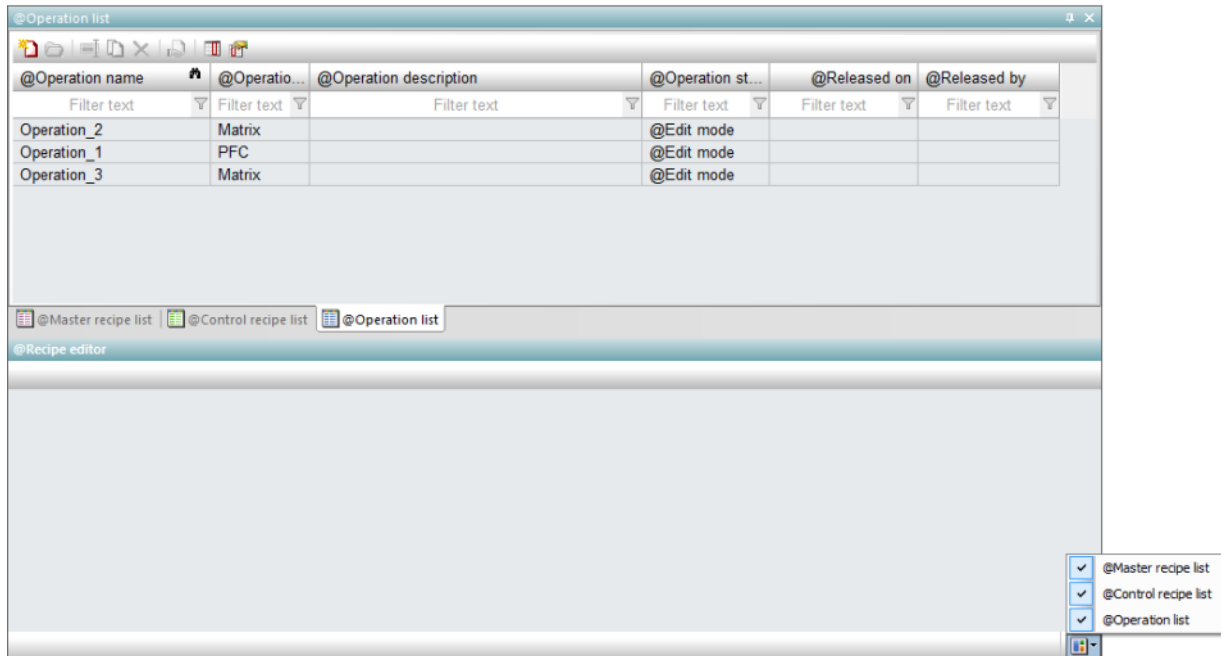


## Selection and positioning

To show and dock the window in the Editor:

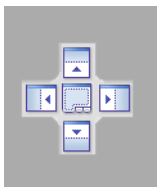
- ▶ Select the desired window using the drop-down list of the **Selection of dockable windows**
- ▶ move the window to the desired position
- ▶ Locate this with the positioning aid

The **Selection of dockable windows** is located at the right edge of the status bar of the Editor. Clicking on the symbol opens the list of dockable windows. Selection is carried out by activating/deactivating the checkboxes.



## POSITIONING AIDS

When moving windows from the Editor interface, positioning aids are displayed. These represent windows or their borders.



This element represents a window area in the Editor.



This element represents the border area of the Editor.

## POSITION WINDOW

To position an element as docked:

1. Move the element with the mouse into the desired area
2. The positioning aid is displayed

3. This represents a window and its areas:
  - a) Center: whole window
  - b) Top: upper half
  - c) Bottom: lower half
  - d) Right: right half
  - e) Left: left halfor the border of the Editor
4. Move the mouse to the central positioning aid or to a positioning aid on the border of the editor and from there to the desired area
5. The area in the Editor where the element was placed when the mouse button was released is colored in blue
6. Move the mouse within the positioning aid to the desired area that is displayed in blue
7. Let the mouse button go and the element is placed

If a window is placed on a pre-existing window, both windows are displayed at the same location using tabs.

**Note:** You can read more about positioning in the chapter on the zenon Editor in the User interface/Positioning windows section.

## 9.2 Commands and actions

In the Runtime the following commands and actions are available:

- ▶ Commands effect the recipe process.
- ▶ Actions make it possible to edit recipes.

### 9.2.1 Commands

For a command to be accepted by the phase, the following requirements are necessary:

- ▶ The REE must run.
- ▶ The phase must be active.
- ▶ The phase must be in a state in which the command is allowed.

Via multi-selection the command can be sent to several phases in the same execution cycle.

**Note:** A distinction is made between recipe commands and phase commands:

- ▶ Recipe commands affect the execution of all phases active in a recipe, as well as the status of the recipe itself.
- ▶ Phase commands are only applied to the selected phase functions (multiple selection is possible).

Command	Description
Start recipe	Starts the recipe process.
Recipe pausing	Pauses the recipe process.
Recipe resuming	Resumes a paused recipe.
Hold recipe	Holds the recipe process.
Restart recipe	Restarts all active elements in the held recipe.
Stop recipe	Stops the recipe.
Recipe aborting	Aborts the recipe.
Phase pausing	Pauses the phase.
Phase resuming	Resumes the paused phase.
Phase holding	Holds phase.
Restart phase	Restarts the held phase.
Switch to automatic mode	Switches the REE to automatic mode.
Switch to semi-automatic mode	Switches the REE to semi-automatic mode.
Switch to manual mode	Switches the REE to manual mode.
Continue recipe only on selected active elements	Continues a recipe at the selected position.
Continue recipe at all execution positions	Continues a recipe on every available position.
Skip active condition	Skips an active condition. Only possible in the manual mode.

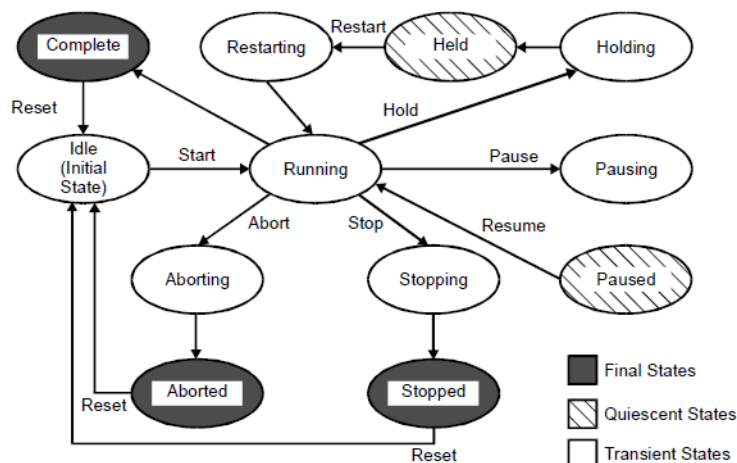
## PERMITTED COMMANDS

Execution conditions apply to recipe commands and phase commands. For example, the control recipe can no longer be stopped if it has the status of `Cancel`.

Command		Start	Stop	Hold	Restart	Abort	Reset	Pause	Resume
Initial State	No Command End State	State Transition Matrix							
Idle		Running							
Running	Complete		Stopping	Holding		Aborting		Pausing	
Complete							Idle		
Pausing	Paused		Stopping	Holding		Aborting			
Paused			Stopping	Holding		Aborting			Running
Holding	Holding		Stopping			Aborting			
Held			Stopping			Aborting			
Restarting	Running		Stopping	Holding		Aborting			
Stopping	Stopped					Aborting			
Stopped						Aborting	Idle		
Aborting	Aborted								
Aborted							Idle		

Note: The `Reset` command is not implemented in zenon Batch Control.

## OVERVIEW OF COMMANDS IN BATCH CONTROL



Note: This overview has been taken from the ANSI/ISA-S88 standard (illustration 18).

### Action on Stop command

After a stop command, the phases, transitions and end simultaneous sequence immediately go to stopped status, even if other elements are still waiting for a condition for stopping. Further subsequent commands such as Cancel are ignored. The Stopped status remains displayed.

## ACTION ON `STOP` COMMAND

After a `stop` command, the **phases**, **transitions** and end **simultaneous sequence** immediately go to `stopped` status, even if other elements are still waiting for a condition for stopping. Further subsequent commands such as `cancel` are ignored. The `stopped` status remains displayed.

### 9.2.2 Commands and actions

In the Runtime the following commands and actions are available:

- ▶ Commands effect the recipe process.
- ▶ Actions make it possible to edit recipes.

## COMMANDS

For a command to be accepted by the phase, the following requirements are necessary:

- ▶ The REE must run.
- ▶ The phase must be active.
- ▶ The phase must be in a state in which the command is allowed.

Via multi-selection the command can be sent to several phases in the same execution cycle.

Command	Description
Start recipe	Starts the recipe process.
Recipe pausing	Pauses the recipe process.
Recipe resuming	Resumes a paused recipe.
Hold recipe	Holds the recipe process.
Restart recipe	Restarts the recipe.
Recipe stopping	Stops the recipe.
Recipe aborting	Aborts the recipe.
Phase pausing	Pauses the phase.
Phase resuming	Resumes the paused phase.
Phase holding	Holds phase.
Restart phase	Restarts the held phase.
Switch to automatic mode	Switches the REE to automatic mode.
Switch to semi-automatic mode	Switches the REE to semi-automatic mode.
Switch to manual mode	Switches the REE to manual mode.
Continue recipe only on selected active elements	Continues a recipe at the selected position.
Continue recipe at all execution positions	Continues a recipe on every available position.
Skip active condition	Skips an active condition. Only possible in the manual mode.

## PERMITTED COMMANDS

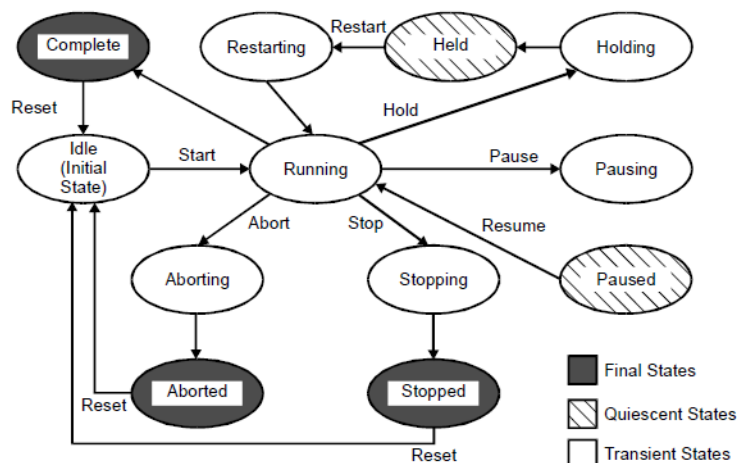
Execution conditions apply to recipe commands and phase commands. For example, the control recipe can no longer be stopped if it has the status of `Cancel`.



Command		Start	Stop	Hold	Restart	Abort	Reset	Pause	Resume
Initial State	No Command End State	State Transition Matrix							
Idle		Running							
Running	Complete		Stopping	Holding		Aborting		Pausing	
Complete							Idle		
Pausing	Paused		Stopping	Holding		Aborting			
Paused			Stopping	Holding		Aborting			Running
Holding	Holding		Stopping			Aborting			
Held			Stopping			Aborting			
Restarting	Running		Stopping	Holding		Aborting			
Stopping	Stopped					Aborting			
Stopped						Aborting	Idle		
Aborting	Aborted								
Aborted							Idle		

Note: The `Reset` command is not implemented in zenon Batch Control.

## OVERVIEW OF COMMANDS IN BATCH CONTROL



Note: This overview has been taken from the ANSI/ISA-S88 standard (illustration 18).

## ACTION ON STOP COMMAND

After a `stop` command, the **phases**, **transitions** and end **simultaneous sequence** immediately go to `stopped` status, even if other elements are still waiting for a condition for stopping. Further subsequent commands such as `cancel` are ignored. The `stopped` status remains displayed.

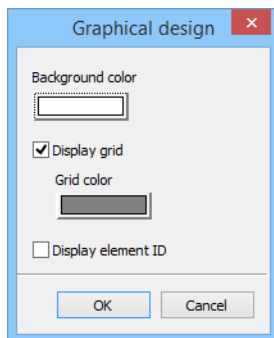
## ACTIONS

Action	Description
Check recipe for errors	Checks the recipe for errors and displays error messages.
Edit element	Opens the corresponding dialog for editing the selected element.
Graphical design	Opens the dialog (on page 131) to configure background colors, grid, and display of element ID.
Duplicate recipe	Copies the selected recipe and adds it as copy to the list.
Create control recipe	Creates a control recipe on the basis of the approved master recipe.
New master recipe	Opens the dialog (on page 140) for creating a new recipe.
Save master recipe	Saves all changes which were done since the last saving.
Delete	Deletes the selected element.
Exchange phase	Opens the dialog (on page 157) for selecting a phase. The present phase is replaced by the newly selected phase.
Edit mode	Switches the mouse cursor from adding an element to edit mode. The switch back to the edit mode can also be achieved by pressing the <code>Esc</code> key.
Insert phase	Occupies the mouse cursor with a phase (on page 155). It can be added to any allowed, free location via click.
Insert transition	Occupies the mouse cursor with a transition (on page 176). It can be added to any allowed, free location via click.
Insert begin simultaneous sequence	Occupies the mouse cursor with a begin parallel branch (on page 179). It can be added to any allowed, free location via click.
Insert end simultaneous sequence	Occupies the mouse cursor with an end parallel branch (on page 179). It can be added to any allowed, free location via click.
Insert Begin branch	Occupies the mouse cursor with a begin branch (on page 176). It can be added to any allowed, free location via click.
Insert End branch	Occupies the mouse cursor with an end branch (on page 176). It can be added to any allowed, free location via click.
Insert unit allocation	Occupies the mouse cursor with a unit allocation (on page 152). It can be added to any allowed, free location via click.
Insert jump target	Occupies the mouse cursor with a jump target (on page

	181). It can be added to any allowed, free location via click.
Switch recipe to test mode	Switches recipe to the test mode (on page 190).
Release recipe	Releases (on page 192) the recipe. With this a control recipe can be created.

## 9.3 Graphical design

Clicking on the symbol for the graphical design in the toolbar opens the dialog for configuring the colors, grid settings and display of the element ID.



Parameters	Description
<b>Background color</b>	Defines the background color of the diagram. Click on the color in order to open the palette for selecting a color.
<b>Display grid</b>	<ul style="list-style-type: none"> <li>▶ Active: Display the grid</li> <li>▶ Inactive: Grid is hidden.</li> </ul> Can only be configured for PFC recipes.
<b>Grid color</b>	Defines the line color of the grid. Click on the color in order to open the palette for selecting a color.
<b>Display element ID</b>	Inactive: No element ID is shown in the recipe. <b>Note:</b> This setting is recommended for normal operation.  Active: The ID of the elements is displayed in the recipe. The exception is lines. The display is in the upper left-hand corner of the element. <b>Note:</b> This setting is recommended for troubleshooting.
<b>OK</b>	Applies all settings and closes the dialog.
<b>Cancel</b>	Discards all changes and closes the dialog.

## 9.4 Engineering rules for recipes

At the engineering the rules defined in standard ANSI/ISA-S88 are generally true.

Important principles:

### GENERAL

- ▶ For all elements all connection points must be connected.  
Exception: Jump targets. Only two of the three input connection points need to be linked there.
- ▶ The **begin element** is always present only once with PFC recipes and marks the beginning of the process. It is not visible with matrix recipes.
- ▶ The **end element** is always present only once with PFC recipes and marks the end of the process. It is not visible with matrix recipes.
- ▶ Phases can be inserted anywhere. You can also place several phases in succession.
- ▶ At least one active phase should be present in the recipe.

### TRANSITIONS

- ▶ Transitions only exist in PFC recipes.
- ▶ Two transitions may not lie one after the other.

### BRANCHES

- ▶ Branches only exist in PFC recipes.
- ▶ The first element after a **Begin branch** must be a transition.
- ▶ The individual branches which start at **Begin branch** must all end in an **End branch** never in an **End parallel branch**. Any element can be placed between begin and end of a branch even parallel branches as long as they are closed before the **End branch** element. An end branch can be replaced with jump targets at any point, including within a parallel branch.
- ▶ It is not necessary to have an **End branch** for each **Begin branch**. You can, for example, have two **Begin branch** elements ending in one **End branch**, or the other way round.
- ▶ It is not necessary to have an **End branch** for a **Begin branch**. It can simply end in a line. If for example you have a **Begin branch** element with two paths and one of the paths ends in a jump target, it does not make sense to have an **End branch**.

### PARALLEL BRANCHES

- ▶ Each parallel branch must contain at least one phase.

- ▶ The first element after a **Begin parallel branch** must not be a transition.
- ▶ The individual branches that start at a **Begin parallel branch** must all end in one **End parallel branch**, but must never end in an **End branch**. You may use any elements between **Begin parallel branch** and **End parallel branch** even branches as long as they are closed before the **End parallel branch**.
- ▶ Not all branches which were started in a **Begin parallel branch** must end in an **End parallel branch**. It is enough when all branches converge over an **End parallel objects**. Equally branches from different **Begin parallel branch** objects may converge in a single **End parallel branch**.
- ▶ parallel branches allows embedding of additional parallel branches.  
In doing so: each embedded parallel branch must recombine with the superordinate parallel branch

## LINES IN THE PFC EDITOR

- ▶ Lines may be used as connections between any objects. It is allowed to add any number of lines after another.
- ▶ Lines must not be used to connect two equal connection points.  
For example: The both inputs of two phases must not be connected directly with a line. In the engineering this connection is allowed. It is however displayed in red (error) and in the validation (on page 194) an error message is displayed.

## JUMP TARGETS

- ▶ Jump targets only exist in PFC recipes.
- ▶ Jump targets correspond to an **end branch**. They are intended to
  - jump between branches,
  - jump out of branches,
  - engineer loops

For this, the following applies: A path which ends in a jump target must have started with a **Begin branch**. Otherwise the end is not reached.
- ▶ Jump targets consist of tree inputs and one output. At least two inputs and the output must always be connected. At this it makes no difference which input connection point is connected.
- ▶ Jump targets can be switched consecutively if at least two input connection points are allocated.
- ▶ Jumps are prohibited:
  - between parallel branches
  - to jump out of a parallel branch
  - to jump in a parallel branch.

## CONFIGURATION OF OBJECTS

### REACTIONS

- ▶ Reactions can appear in each object state.
- ▶ For each reaction type several reactions are possible. They are sorted at the triggering and are executed in accordance to their priority. At this 1 is the highest priority.
- ▶ All variables of all parameters are signed in to the driver for reading. If a value is needed at a reaction but is not yet available or invalid, the alternate value is written. The writing of the value is done without write confirmation.
- ▶ Some of the reactions are triggered only once in the process - e.g. time outs. If the phase is restarted, these reactions are also retriggered if necessary.

## 9.5 Status line

The status line is automatically adapted to each recipe that contains the focus and initialized with its data. The status line the following is displayed:

- ▶ Mode of the recipe
  - Edit mode
  - Manual mode
  - Semi-automatic mode
  - Automatic mode
- ▶ Status of the recipe:
  - Ready for start
  - Error: Currently there is an error in the recipe. The number of errors is also displayed.
  - Historical error: During the execution at least one error occurred. Currently not error exists.
  - No error: Until now the execution runs faultless.
- ▶ Current execution status (on page 231).
- ▶ Mouse cursor position and changes such as moving the connection line
- ▶ Deleting, adding or selecting of elements
- ▶ Approval of a recipe
- ▶ Finished without errors
- ▶ Zoom level of the current view; this can also be controlled here

The global statuses of the main recipe are also displayed when executing operations.

## 9.6 Recipe types and recipe states

Batch Control distinguishes between the two following recipe types:

- ▶ **PFC recipes:** For free, complex processes. These are created with the PFC editor (on page 147).
- ▶ **Matrix recipes:** For simple, sequential, parallel processes. They are created with the matrix editor (on page 183).

The following type of recipes are available:

- ▶ **Master recipes** (on page 139):  
Form the basis for the control recipes. The created in status `Editable`. If they are in status `Released`, they can no longer be modified.
- ▶ **Operations** (on page 194):  
Form a substructure that can be embedded in recipes. This can provide a better overview in complex recipes. Operations are created in a similar manner to matrix recipes or PFC recipes. The operations are created on the basis of templates and as an instance in existing matrix recipes or PFC recipes.
- ▶ **Control recipes** (on page 204):  
Are recipes which can be executed once. Each control recipe is based on a master recipe. Control recipes can be modified concerning the process. The process is defined by the master recipe. Changeable are only command tags for which option **Changeable in control recipe** was activated.

Recipes are created and edited in Runtime. If a screen that contains recipes that have not been saved is closed, the user is asked if the recipes are to be saved. If there is no input from the user within 15 seconds, the unsaved changes are discarded and the screen is closed.

### STATUS FOR MASTER RECIPES

Master recipes can have the following states:

- ▶ **Editable:** In this state everything can be changed. Each new master recipe is created in this status.
- ▶ **Test mode** (on page 190): In this status the recipe behaves similar as a control recipe. As them the process cannot be modified. It can be executed and all commands, actions, modes, etc. are available. You can also change all command tags for which option **Changeable in recipe** was activated.  
Exception: If a phase is active at the moment, changes are not possible. Decisive for this is the status at clicking button OK. If the phase is active, the value changes are not taken over and an error message is displayed.  
In test mode recipes can be executed consecutively several times. This is not the case for control recipes.  
Changes from the Editor are only applied after reloading or after Runtime has been restarted, however only once the recipe has been ended.

- ▶ **Released** (on page 192): In this status no changes to the recipe are possible  
**Note:** Also changes to the phases or to the reactions in the Editor are not transferred to a master recipe. The status at the release are frozen.  
 Exception: Changes to variable and function are not considered by these protection mechanisms. This can lead to a master recipe and all control recipes based on it becoming invalid. If e.g. the data type of a variable is changed from Bool to String, the validation function is no longer run through and the recipe can no longer be used as long as the error is not fixed in the Editor. Likewise the units themselves are not protected as they are used recipe-spanning. If a unit name is changed in the Editor, it immediately takes effect after reloading or a Runtime restart on all phases which are based on it.

To be able to release a master recipe, the recipe must be validated without errors. The validation (on page 194) is done automatically during the release and cannot be avoided. Control recipes can only be created from released recipes.

If a master recipe with status `Release` should be modified, you must create a copy of the master recipe via command `duplicate`. The copy gets status `Editable`.

## STATUS FOR CONTROL RECIPES

Control recipes can have the following states:

- ▶ **Prepared:** A newly created control recipe has this status. In this status, it can be started and command tags, for which option **Changeable in control recipe** was activated, can be changed.
- ▶ **In execution:** The control recipe was started and is processed. It remains in this status until it is `Completed`, `Stopped` or `Canceled`. You can learn the exact state of the recipe from the execution status (on page 231).  
 In this status, command parameters for which the **Changeable in control recipe** option was activated can be changed.  
 Exception: If a phase is active at the moment, changes are not possible. Decisive for this is the status at clicking button OK. If the phase is active, the value changes are not taken over and an error message is displayed.
- ▶ **Finished:** The recipe reached its final state. It can no longer be restarted and it also cannot be changed. Changes to command tags are not possible. Finished control recipes can be duplicated and deleted.

## 9.7 Control strategies

Control strategies can be selected in Runtime when configuring the phase in the PFC recipe (on page 155) or matrix recipe (on page 188). They must have already been configured in the Editor (on page 44).

Control strategies can only be changed:

- ▶ In master recipes
  - In editing mode



- Test mode
- ▶ In operation templates:
  - In editing mode

The control strategy cannot be changed in operation instances.

## COMMAND PARAMETERS IN THE PHASE

If control strategies are activated for a phase (**Active control strategies** property in the Editor), all command parameters are initially removed when this phase is inserted into a recipe.

If the control strategy is changed in the dialog to edit a phase, then:

- ▶ All parameters currently in this phase are removed
- ▶ The linked parameters in the newly-selected control strategy are inserted

These parameters are only initialized with the values that the parameter linkings currently have. The existing Runtime configuration of the parameters is lost in the process.

If a control strategy is selected for a phase, then the parameters to write to **control strategy number** are inserted in addition to the linked parameters. This parameter gets the parameter number as a numerical value. It is not a recipe parameter. If the same parameter is already linked to the control strategy, the configuration of the linking is ignored!

## SYNCHRONIZATION OF CHANGES TO CONTROL STRATEGIES AND CONTROL STRATEGY ACTIVATION




During synchronization (on page 212), changes made in the Editor for control strategies are carried over for phases. For details, see the **synchronization** (on page 212) chapter.

## DISPLAY OF THE CONTROL STRATEGY

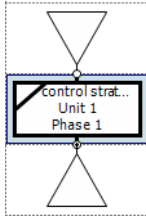
Selected control strategies are displayed:

- ▶ In the tooltip of the phase (no display if no control strategy has been selected)
- ▶ In the phase in the recipe.

Matrix:

	Unit 1 Phase 1		
1 Step 1	control strategy 1		
			

PFC:



- ▶ In the unit information for the active phases as an appendix
- ▶ In the report (on page 250) for a phase

## CONTROL STRATEGIES AND VARIABLES

If a phase is selected in the recipe, the variables for the control strategy are filled.

Parameters	Data type	Description
<b>Control strategy name</b>	STRING	<p>Displays the <b>Name</b> of the control strategy that is linked to the selected phase.</p> <p>Is empty if:</p> <ul style="list-style-type: none"> <li>▶ No phase is selected</li> <li>▶ The phase does not use any control strategies</li> <li>▶ There is no control strategy currently linked to the phase</li> <li>▶ With multiple selection</li> </ul>
<b>Description of control strategy</b>	STRING	<p>Displays the <b>Description</b> of the control strategy that is linked to the selected phase.</p> <p>Is empty if:</p> <ul style="list-style-type: none"> <li>▶ No phase is selected</li> <li>▶ The phase does not use any control strategies</li> <li>▶ There is no control strategy currently linked to the phase</li> <li>▶ With multiple selection</li> </ul>
<b>control strategy number</b>	LINT	<p>Displays the <b>control strategy number</b> of the control strategy that is linked to the selected phase.</p> <p>Is -1 if:</p> <ul style="list-style-type: none"> <li>▶ No phase is selected</li> <li>▶ The phase does not use any control strategies</li> <li>▶ There is no control strategy currently linked to the phase</li> <li>▶ With multiple selection</li> </ul>

## 9.8 Master recipes

Master recipe are the basis of control recipes. The recipe process is defined and tested with the help of master recipes. After a master recipe is released, its content and structure can no longer be changed.

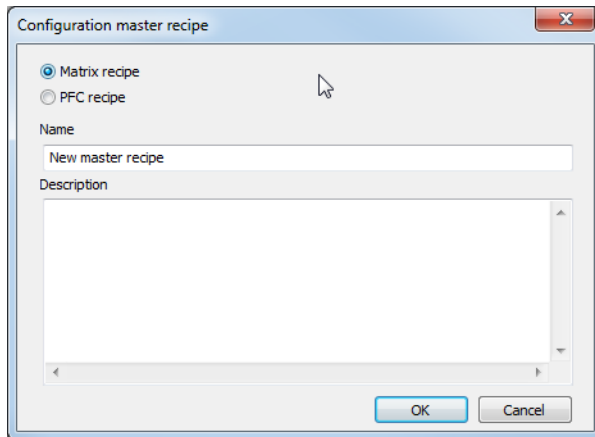
For the creation of master recipes two editors are available:

- ▶ Matrix editor
- ▶ PFC editor

Depending on the license, either just the matrix editor or both editors are available.

### 9.8.1 Create master recipe

A click on the **Create master recipe** button opens the **Master recipe configuration** dialog.



Parameters	Description
<b>Matrix recipe</b> (on page 182)	<p>Activate this radio button if you want to create a matrix recipe (on page 182).</p> <p>Note: Only possible if the corresponding license is available.</p>
<b>PFC recipe</b> (on page 146)	<p>Activate this radio button if you want to create a PFC recipe (on page 146).</p> <p>Note: Only possible if the corresponding license is available.</p>
<b>Name</b>	<p>Unique name for the recipe. The name must not contain a dot (.), a question mark (?), a @ or an asterisk (*).</p> <p>Maximum length: 256 characters.</p> <p>Note: When you copy a recipe the existing name is complemented with the prefix "<b>Copy of</b>". If the maximum length is exceeded by this, the name is shortened to the allowed length starting from the last character.</p> <p>The uniqueness is checked in the entire network. Therefore it can happen that you cannot take over the name as another user on another computer in the zenon network already has used the same name and you do not see the recipe in the list of the master recipes yet.</p> <p>The recipe names can be changed later but only as long as the recipe is in status <code>Editable</code>.</p>
<b>Description</b>	<p>Optional description for the recipe that is to be created.</p> <p>You can change the description later, but only as long as the recipe is in <code>editable</code> status. To change the description, select the <b>Rename master recipe</b> symbol.</p>
<b>OK</b>	<p>Applies all settings and created a new recipe.</p>

<b>Cancel</b>	Closes the dialog without creating a recipe.
---------------	--

## 9.8.2 Toolbar and context menu for master recipe list view

### TOOLBAR

Symbol	Description
<b>New master recipe</b>	Opens the dialog for creating a new master recipe.
<b>Open master recipe in Editor</b>	Opens the selected recipe in the recipe editor.
<b>Create new version</b>	Creates a new version (on page 192) of the selected master recipe. This must be approved or marked as obsolete.
<b>Rename master recipe</b>	Opens dialog to rename the selected recipe.
<b>Duplicate master recipe</b>	Creates a copy of the selected recipe and opens the dialog to rename the duplicate.
<b>Delete master recipe</b>	Deletes selected recipes.
<b>Export selected XML</b>	Exports the selected master recipe as an XML file.
<b>Import XML</b>	Imports the selected XML file as a master recipe(s).
<b>Switch master recipe to test mode</b>	Switches selected recipe to test mode after requesting confirmation.
<b>Switch master recipe to edit mode</b>	Switches selected recipe to edit mode after requesting confirmation.
<b>Release master recipe</b>	Approves selected recipe after requesting confirmation.

<b>New control recipe</b>	Opens the dialog for creating a new control recipe.
<b>Include running control recipes in the display</b>	Includes running control recipes in the display.
<b>Include prepared control recipes in the display</b>	Includes prepared control recipes in the display.
<b>Include finished control recipes in the display</b>	Includes finished control recipes in the display.
<b>Include outdated control recipes in the display</b>	Includes outdated control recipes in the display.
<b>Display list of attendant control recipes in control recipe list</b>	Shows all control recipes that belong to approved control recipes.
<b>Column selection</b>	Opens the dialog for selecting the columns which should be displayed.
<b>Column Format</b>	Opens the dialog for configuring the column formats.

## CONTEXT MENU

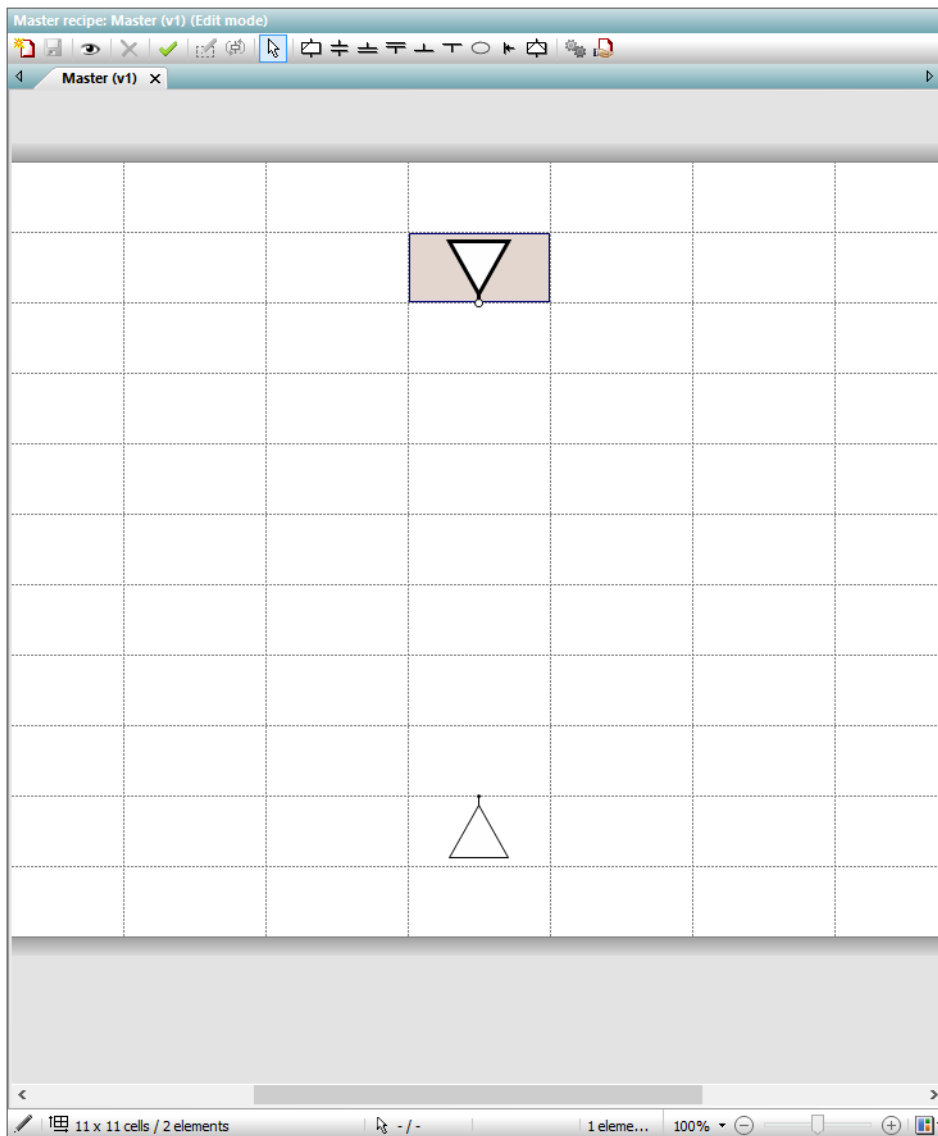
Command	Description
<b>New master recipe</b>	Opens the dialog for creating a new master recipe.
<b>Open in Recipe Editor</b>	Opens the selected recipe in the recipe editor.
<b>Rename...</b>	Opens dialog to rename the selected recipe.
<b>Duplicate...</b>	Creates a copy of the selected recipe and opens the dialog to rename the duplicate.
<b>Delete</b>	Deletes selected recipes.
<b>Export selected XML...</b>	Exports the selected master recipe as an XML file.
<b>Import XML...</b>	Imports the selected XML file as a master recipe(s).
<b>Switch to edit mode</b>	Switches selected recipe to edit mode after requesting confirmation.



<b>Switch to test mode</b>	Switches selected recipe to test mode after requesting confirmation.
<b>Release</b>	Approves selected recipe after requesting confirmation.
<b>Highlight as outdated</b>	Marks the selected recipe as outdated.
<b>New control recipe</b>	Opens the dialog for creating a new control recipe.
<b>Display associated control recipes in list</b>	Shows all control recipes that belong to selected approved control recipes.

### 9.8.3 PFC recipe

If you selected PFC recipe in the **master recipe configuration** dialog and exited the dialog with OK, the newly-created recipe opens on a new tab in the PFC editor (on page 147).

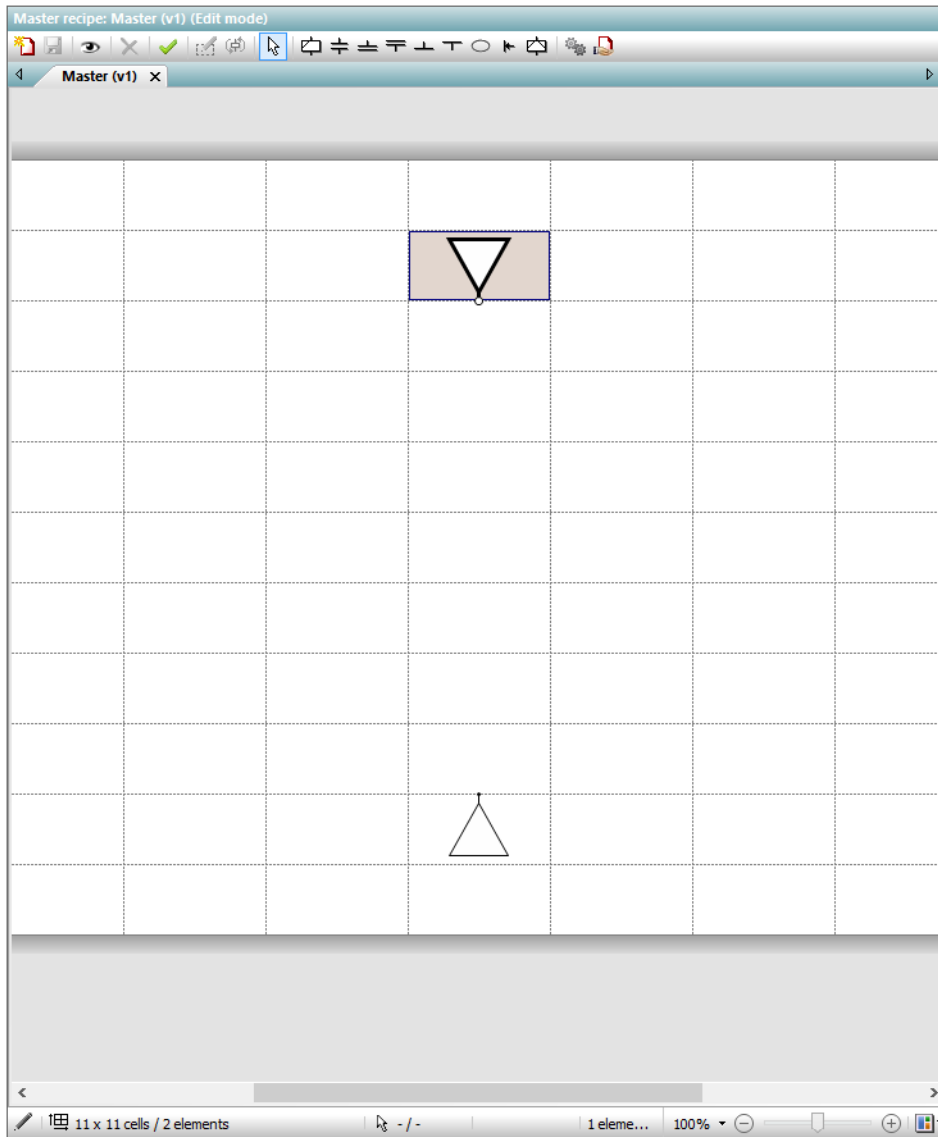


As each recipe needs a begin and an end element, these two elements already exist in the recipe and cannot be deleted from it.

Add the desired phases, transitions, branches, parallel branches and unit allocations to your recipe. Fields which are unsuitable for adding an element turn red when you move the element above it.

## PFC editor

In the PFC editor you can create your recipes graphically.



## TECHNICAL DETAILS

- ▶ Sheet size:
  - Default: 11 x 11 cells
  - Minimum: 5 x 5 cells
  - Maximum: 500 x 1000 cells
- ▶ Cell size

- Default: 155 x 111 pixels
- ▶ Outside edge: 100 pixel
- ▶ Grid: is displayed by default; can also be hidden
- ▶ Scroll bar: Is displayed if the document is larger than the frame.
- ▶ Scrolling with a mouse wheel: up and down or, if you press and hold `Shift`, left and right.
- ▶ Zooming: `Ctrl + mouse wheel`
- ▶ Selecting elements: `left mouse click`
- ▶ Multiple selection: `Ctrl + mouse click`
- ▶ Move symbol: Click element and move it over the diagram while holding the left mouse button pressed. Content can be dropped to cells with green background. If a cell turns red when you move over it, you cannot drop the content.

## Toolbar and context menu PFC recipe

### TOOLBAR PFC EDITOR: EDIT MODE



Parameter	Description
<b>New master recipe...</b>	Opens the dialog for creating a new master recipe.
<b>Save master recipe</b>	Saves the master recipe which is open for editing.
<b>Graphical design</b>	Opens the dialog (on page 131) to configure background colors, grid, and display of element ID.
<b>Delete</b>	Deletes the selected elements from the diagram. Multiple selection via <code>Ctrl+mouse click</code> .
<b>Check recipe for errors</b>	Checks recipe for errors and displays found errors in an information window. For several errors the first errors are displayed.  The error message contains the error number, the ID of the element, its location and a message in plain text.
<b>Edit element</b>	Opens the corresponding dialog for editing the selected element.
<b>Replace phase/operation</b>	Opens dialog to select a phase or an operation and replaces the existing phase with the newly-selected one or the existing operation with a newly-selected one. Several phases or operations can be selected and replaced together.  Shortcut: <code>Shift+double click</code>
<b>Edit mode</b>	Toggles between insert mode and edit mode.
<b>Insert phase</b>	Adds a phase.
<b>Insert transition</b>	Adds a transition.
<b>Insert begin simultaneous sequence</b>	Adds a begin simultaneous sequence.
<b>Insert end simultaneous sequence</b>	Adds an end simultaneous sequence.
<b>Insert Begin branch</b>	Adds a begin branch.
<b>Insert End branch</b>	Adds an end branch.
<b>Insert unit allocation</b>	Adds a unit allocation.
<b>Insert jump target</b>	Adds a jump target.

<b>Insert operation</b>	Adds an operation (on page 194).
<b>Switch recipe to test mode</b>	Switches recipe to the test mode. For this the recipe must be without errors.
<b>Release recipe</b>	Releases the recipe. With this a control recipe can be created.

**Note:** Insertions remain active until you change to the edit mode using the **Edit mode** symbol, the `Esc` key or you change to another insert option via a symbol for adding a new element.

#### TOOLBAR PFC RECIPE TEST MODE



Parameter	Description
<b>Start recipe</b>	Starts the recipe process.
<b>Recipe pausing</b>	Pauses the recipe.
<b>Recipe resuming</b>	Resumes paused recipe.
<b>Recipe holding</b>	Holds recipe.
<b>Restart recipe</b>	Restarts held recipe.
<b>Recipe stopping</b>	Stops the recipe.
<b>Recipe aborting</b>	Aborts the recipe process.
<b>Phase pausing</b>	Pauses the phase.
<b>Phase resuming</b>	Resumes the process of a paused phase.
<b>Phase holding</b>	Holds phase.
<b>Restart phase</b>	Restarts held phase.
<b>Escape phase</b>	Starts process to exit from the phase.
<b>Check recipe for errors</b>	Starts recipe validation (on page 194).
<b>Edit element</b>	Opens the corresponding dialog for editing the selected element.
<b>Graphical design</b>	Opens the dialog (on page 131) to configure background colors, grid, and display of element ID.
<b>Switch to automatic mode</b>	Switches process to <code>automatic</code> mode.
<b>Switch to semi-automatic mode</b>	Switches process to <code>semi-automatic</code> mode.
<b>Switch to manual mode</b>	Switches process to <code>manual</code> mode.
<b>Continue recipe only on selected active elements</b>	Continues a recipe at the selected position.
<b>Continue recipe at all execution positions</b>	Continues a recipe on every available position.
<b>Skip active condition</b>	Skips an active condition. Only possible in the <code>manual</code> mode.
<b>Edit mode</b>	Switches from test mode to edit mode.
<b>Release recipe</b>	Releases the recipe. With this a control recipe can be created.

## PFC RECIPE TOOLBAR: APPROVED



Parameter	Description
<b>Check recipe for errors</b>	Checks recipe for errors and displays found errors in an information window. For several errors the first errors are displayed.  The error message contains the error number, the ID of the element, its location and a message in plain text.
<b>Edit element</b>	Opens the corresponding dialog for editing the selected element.  The dialog is opened in write-protected mode, because it is no longer possible to edit approved recipes.
<b>Graphical design</b>	Opens the dialog (on page 131) to configure background colors, grid, and display of element ID.
<b>Duplicate recipe</b>	Only active if precisely one recipe was selected. Creates a copy of the selected recipe. At the creation of the copy, the version of the recipe saved on the hard disk is used. If the recipe is just edited in another computer and the changes have not yet been saved, the changes are not applied. The dialog for the input of a unique name and the description is opened.
<b>Create control recipe</b>	Creates a control recipe on the basis of the approved master recipe.



### Information

*The functions of the individual symbols can also be configured using buttons and thus be made touch-operable. Tool bars can therefore also be hidden (on page 117).*

## Begin element

First element in the process. It is automatically created in the editor when a recipe is created and it cannot be deleted.

## Allocate and configure unit

To allocate a unit:

1. in the toolbar select the symbol for **Insert unit allocation**



2. move the mouse to the desired position
3. place the unit
4. the unit is added

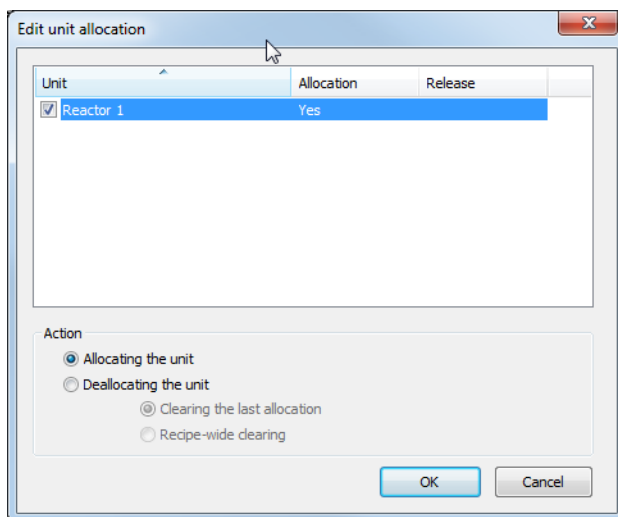
To configure the unit allocation:

1. double click the unit
2. The dialog for configuration is opened

## CONFIGURATION

In a configuration dialog you can select the units which exist in the batch recipe for the Runtime. It is distinguished between allocation and release. Releases can be set selectively and globally. Units which were selected before and no longer exist in the recipe, are displayed with an appropriate note.

If the dialog is opened in an approved recipe, then it is displayed as "read only". All entries can be viewed but not changed.



Parameters	Description
<b>List units</b>	Displays existing units and their assigned actions.
<b>Action</b>	Assigns an action to the unit selected in the list.
<b>Allocation of the unit.</b>	Allocates the unit.
<b>Release of the unit.</b>	Releases the unit: <ul style="list-style-type: none"> <li>▶ Release of the last allocation: The last allocation is released.</li> <li>▶ Recipe-spanning release: All allocations in the recipe are released.</li> </ul>
<b>OK</b>	Applies settings and closes the dialog.
<b>Cancel</b>	Discards changes and closes dialog.

## ALLOCATE AND RELEASE UNITS

The allocation of a unit by element unit allocation or by a phase is only possible if the unit is not yet allocated or only in the same recipe. Allocation and release of units is always done in a cycle. It is always waited until all units which should be allocated are released. Then all allocations and releases are down at the same time. Phases which are located in front of an element unit allocation remain active until the allocation was successful.

Unit allocations remain in place as long as the phase is active. When the recipe is finished and there are still allocations of elements **Unit allocation** active, they are released implicitly.

**Note:** A phase with the `paused` or `held` status does not attempt to allocate the unit. This also applies if the phase is switched to `paused` or `held` whilst waiting. An attempt to allocate the unit is only made after a restart. But: In manual mode you can force the allocation of a unit by another recipe. The recipe with the first allocation keeps the control and takes priority at the execution. If this recipe withdraws its main allocation, the recipe with the longest active forced allocation takes over the main allocation.

## ACTIONS

For each unit which is used in the recipe, you can define an action:

- ▶ no action
- ▶ allocate
- ▶ release

## TOOLTIP

In the tool tip of element **Unit allocation** all units are displayed which are marked for allocation or release. During the execution the Execution duration (on page 43) is stated and all units for which you must wait are color-coded. You must wait for units if they are allocated in another recipe.

## Add and configure phase

To add a phase:

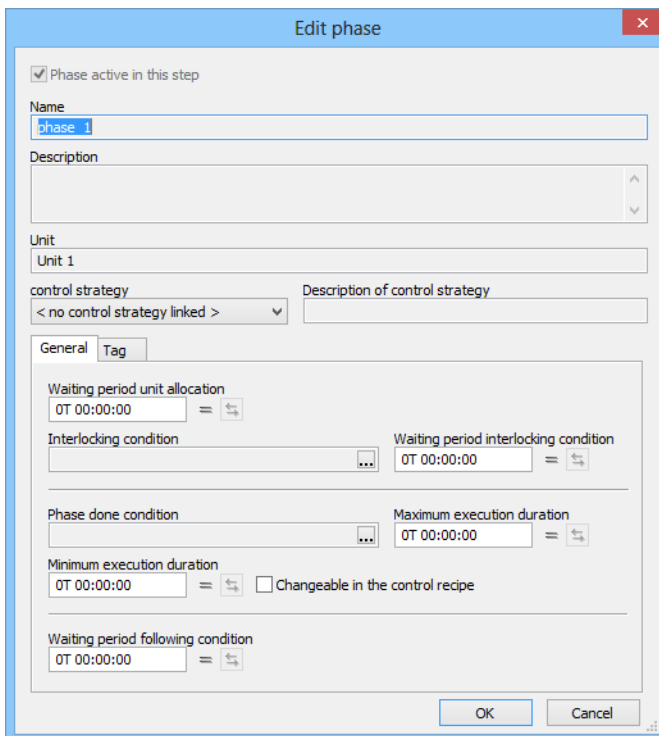
1. in the toolbar select the symbol for **Insert phase**
2. move the mouse to the desired position
3. locate the phase
4. the dialog for selecting a phase (on page 157) is opened
5. select the desired phase

To configure a phase:

1. double click the phase
2. The dialog for configuration is opened  
(if an element has not yet been assigned a phase, double-clicking opens the dialog to select a phase)

**Note:** The `shift` key plus a `double click` always opens the dialog to select a phase.

## CONFIGURATION



Parameters	Description
<b>Phase active in this step</b>	Active: Phase is active in this step
<b>Name</b>	Name of the phase. Only display.
<b>Description</b>	Comment about the phase. Only display.
<b>Unit</b>	Unit on which the phase is carried out. Only display.
<b>Control strategy</b>	Selection of a control strategy (on page 136) from a drop-down list. Only available if control strategies have been configured (on page 44) for this phase.  Default: no Control strategy linked
<b>Description of control strategies</b>	Description of the selected control strategy.  Display of the description entered in the Editor only.
<b>General</b>	Tab for configuration of general properties.
<b>Parameters</b>	Tab for configuration of parameters.
<b>OK</b>	Applies all changes on all tabs and closes the dialog.
<b>Cancel</b>	Discards all changes on all tabs and closes the dialog.

Configuration of the tabs see chapter:

- ▶ General: (on page 157) Display and configuration of the settings for the phase
- ▶ Parameter (on page 162): Configuration of the tags

The entry of a reason can be requested in order to make changes. To do this, either the **Reason for value change necessary** property must be activated in the **Edit tag** node for the module in general, or the **Reason for value change necessary** property in the **General** group for individual functions.

## RULES FOR EDITING A PHASE

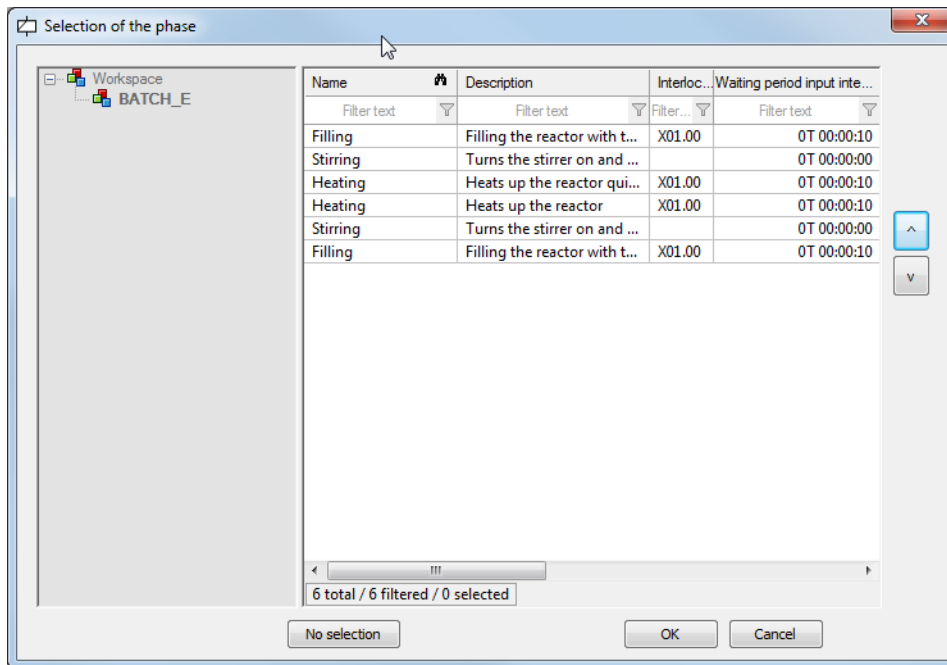
A phase can be edited:

- ▶ In a master recipe in edit mode: If the user has sufficient rights.
- ▶ In a master recipe in test mode: If the user has sufficient rights and the phase is not active.
- ▶ In a control recipe with `prepared` status: If the user has sufficient rights and the **Changeable in the control recipe** option has been activated.
- ▶ In a control recipe with `running` status: If the user has sufficient rights, the **Changeable in the control recipe** option has been activated and the phase is not active.

The phase can no longer be edited in pre-configured control recipes and in approved master recipes.

## Selection phase

If a phase is added, the dialog for selecting a phase is opened.




Parameter	Description
<b>Project tree</b>	Displays the current project from which the phases can be selected.
<b>List field phases</b>	<p>In the list all phases engineered in the Editor are displayed.</p> <p>The list can be filtered. The filtering is case-sensitive. Placeholders * and ? can be used.</p>
<b>Cursor keys</b>	Move selected phase up or down.
<b>None</b>	Deletes already selected phases from the element.
<b>OK</b>	Applies settings and closes the dialog.
<b>Cancel</b>	Discards all changes and closes the dialog.

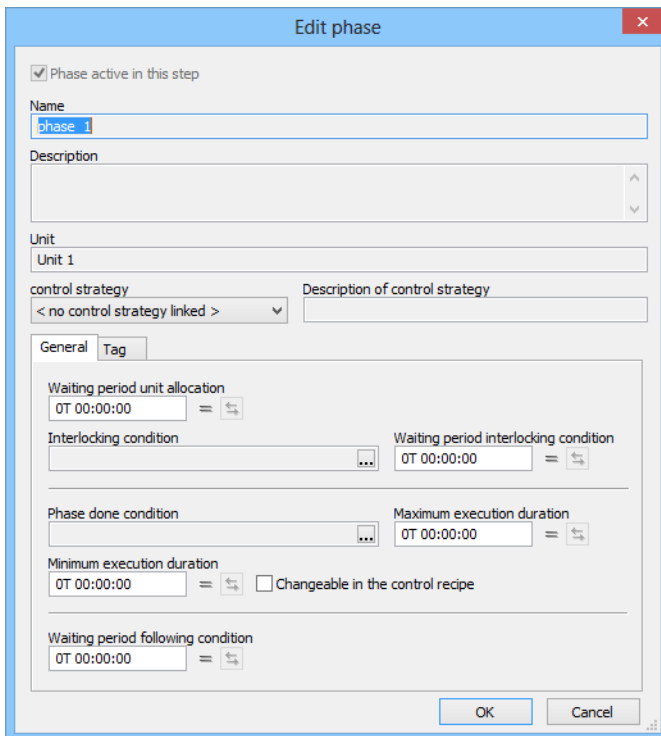
All settings of the dialog are saved user-specifically when the dialog is closed.

## General

The properties of tab **General** are set in the Editor with the exception of **Changeable in the control recipe**. The values can be changed in the master recipe. A symbol indicates whether the value in the dialog matches the value in the Editor. For different values you can again apply the value defined in the Editor.

Meaning of the symbols next to the values:

Parameter	Description
=	Value in the dialog and the value in the Editor match.
<>	Value in the dialog and the value in the Editor do not match.
	Only active if the values in the recipe and Editor do not match. Click on button to apply the value from the Editor. It overrides the value in the master recipe.



For information on configuration of the basic data on these tabs, see the Add and configure phases (on page 155) chapter.

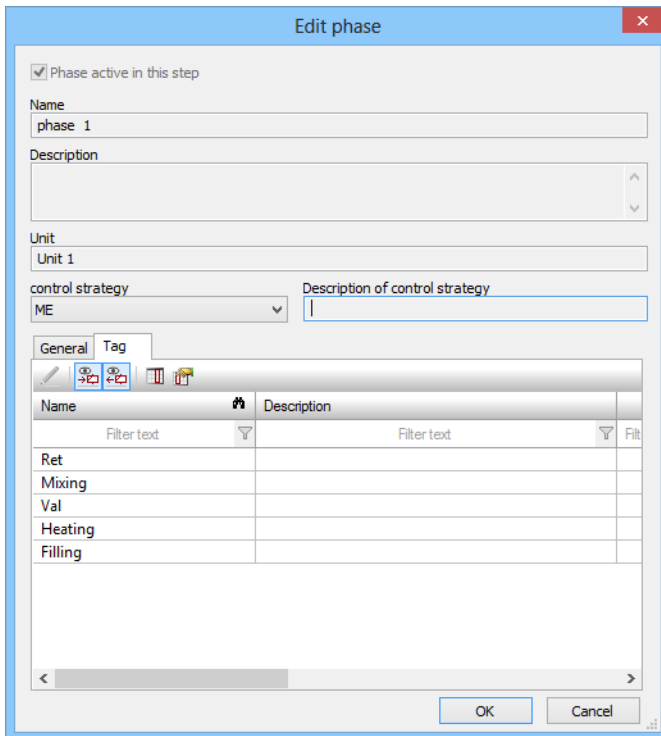
Parameter	Description
<b>Waiting period unit allocation</b>	<p>Time in days, hours, minutes and seconds which is waited for the allocation of the unit. The waiting period can be changed in the Runtime as long as the recipe has status <code>Editable</code>.</p> <p>After the defined period has exceeded, the event <code>Waiting period unit allocation exceeded</code> is triggered and the element is highlighted. Additional actions must be defined by the engineer. If no further actions take place, the waiting is continued.</p> <p>Maximum: 9999T 23:59:59 Default: 0d 00:00:00</p> <p>Note: A unit can only be allocated by a single recipe with status <code>In execution</code> at a time.</p> <p>Value is predefined in the Editor and can be changed here as long as the recipe is not released.</p>
<b>Input lock</b>	<p>Defines conditions for input interlocking. Click on button <code>...</code> or an entry to open the formula editor (on page 255) for defining the condition.</p> <p>If an input interlocking is configured, the phase is only executed in the Runtime when the condition for the input interlocking is fulfilled. The formula can consist of one or more command tags and return tags of the phase. Value and status of the variables can be used. The formula returns <code>TRUE</code> or <code>FALSE</code> as result. The condition can be displayed in the Runtime but cannot be changed there.</p> <p>The waiting period for the input interlocking is configured with the help of property <b>Waiting period input interlocking</b>.</p>
<b>Waiting period input interlocking</b>	<p>Time period in days, hours, minutes and seconds in which the condition defined in property <b>Interlocking condition</b> must return value <code>TRUE</code>. The waiting period begins with the beginning of the check of the input interlocking. If the condition is not fulfilled within the waiting period, the event <code>Waiting period input interlocking exceeded</code> is triggered and the waiting is continued. If no reaction was defined for the event which forces another behavior, it is waited until the condition is fulfilled.</p> <p>If 0d 00:00:00 is defined as waiting period, the event is not triggered.</p> <ul style="list-style-type: none"> <li>▶ Minimum value: 0d 00:00:00</li> <li>▶ Maximum value: 9999d 23:59:59</li> <li>▶ Default: 0d 00:00:00</li> </ul> <p>Value is predefined in the Editor and can be changed here as long as the recipe is not released.</p>
<b>Phase done condition</b>	<p>Defines the condition for phase done. Click on button <code>...</code> or an entry</p>

	to open the formula editor (on page 255) for defining the condition. The condition can be displayed in the Runtime but cannot be changed there. The period which is waited for the fulfillment of the condition is defined via property <b>Maximum execution duration</b> .
<b>Maximum execution duration</b>	<p>Period in days, hours, minutes and seconds in which the condition defined in property <b>Phase done condition</b> must return value TRUE. The waiting period begins when writing of the command parameter. If the condition is not fulfilled within the waiting period, the event <code>Maximum execution period exceeded</code> is triggered and the waiting is continued. If no reaction was defined for the event which forces another behavior, it is waited until the condition is fulfilled.</p> <p><b>Note:</b> Time keeps running when the recipe/the phase is paused.</p> <p>If <code>0d 00:00:00</code> is defined as waiting period, the event is not triggered.</p> <ul style="list-style-type: none"> <li>▶ Minimum value: <code>0d 00:00:00</code></li> <li>▶ Maximum value: <code>9999d 23:59:59</code></li> <li>▶ Default: <code>0d 00:00:00</code></li> </ul> <p>Value is predefined in the Editor and can be changed here as long as the recipe is not released.</p> <p>Makes only sense if property <b>Phase done condition</b> was configured.</p> <p>Read more about execution duration in chapter Execution duration (on page 43).</p>
<b>Minimum execution duration</b>	<p>Minimum execution duration of the phase.</p> <ul style="list-style-type: none"> <li>▶ <code>&gt;0</code>: Period which is at least waited after the writing of the command tag regardless of whether the phase done condition is fulfilled.</li> <li>▶ <code>0</code>: Execution duration is not checked</li> </ul> <p>Minimum execution duration can exceed <b>Maximum execution duration</b>. Value is predefined in the Editor and can be changed here as long as the recipe is not released.</p> <p>Changeable in the control recipe if option <b>Changeable in the control recipe</b> was activated in the master recipe.</p> <p>Read more about execution duration in chapter Execution duration (on page 43).</p>
<b>Changeable in the control recipe</b>	<b>Active:</b> Property can be changed in the control recipe.
<b>Waiting period following condition</b>	Period in days, hours, minutes and seconds in which the phase must be deactivated. The waiting period begins when the



	<p>phase done condition is reached. If the condition is not fulfilled within the waiting period, the event <code>Waiting period following condition exceeded</code> is triggered and the waiting is continued. If no reaction was defined for the event which forces another behavior, it is waited until the condition is fulfilled.</p> <p>If <code>0d 00:00:00</code> is defined as waiting period, the event is not triggered.</p> <ul style="list-style-type: none"> <li>▶ Minimum value: <code>0d 00:00:00</code></li> <li>▶ Maximum value: <code>9999d 23:59:59</code></li> <li>▶ Default: <code>0d 00:00:00</code></li> </ul> <p><b>Note:</b> If the following condition exists depends on the recipe structure. Therefore the configuration of a waiting period should not be done in the Editor but in the Runtime. Value is predefined in the Editor and can be changed here as long as the recipe is not released.</p> <p>For more information see chapters <b>Times</b> (on page 29) and <b>Following conditions</b> (on page 242).</p>
<b>OK</b>	Applies all changes on all tabs and closes the dialog.
<b>Cancel</b>	Discards all changes on all tabs and closes the dialog.

## Parameters



For information on configuration of the basic data on these tabs, see the Add and configure phases (on page 155) chapter.

Parameters	Description
List tag	Displays the tag configured in the Editor. Tags can be filtered and sorted according to columns.  Click on symbol <b>Edit tag</b> , double click the tag, menu item in the context menu or press <b>Return</b> to open the dialog (on page 163) for editing a tag.
<b>OK</b>	Applies all changes on all tabs and closes the dialog.
<b>Cancel</b>	Discards all changes on all tabs and closes the dialog.

## TOOLBAR



Symbol	Meaning
<b>Edit tag</b>	Opens the dialog (on page 163) for editing the tag.
<b>Display all command tags</b>	Toggles between the display of the changeable tags and all tags.
<b>Display return tags</b>	In addition to the command tags also displays the return tags or hides them.
<b>Column selection</b>	Opens the dialog (on page 172) for selecting the columns which should be displayed.
<b>Column Format</b>	Opens dialog (on page 174) to format the columns.

## Edit tag

To edit a tag in the Runtime:

1. in dialog Edit phase select tab **Tag**
2. highlight the desired tag
3. open the dialog for editing the tags via a click on symbol **Edit tag**, the context menu, a double click on the tag or press `Return`
4. the dialog for editing is opened

For each data type an own dialog is opened:

- ▶ Numerical
- ▶ Binary
- ▶ String
- ▶ Time period

The properties are normally configured in the Editor and only displayed in the Runtime. Exceptions are values of the data type. They can be adapted if property **Changeable in master recipe** was activated in the Editor.

## NUMERIC TAGS

Change tag properties

Parameter  
Fill in water

Description

Type  
Command

Data type  
Numeric

☐ Changeable in the control recipe

Minimum 30 = Minimum variable 0

Value 50 = Unit liter

Maximum 80 = Maximum variable 65535

Variable  
Menge Wasser

Data type variable  
UINT

Driver  
Intern

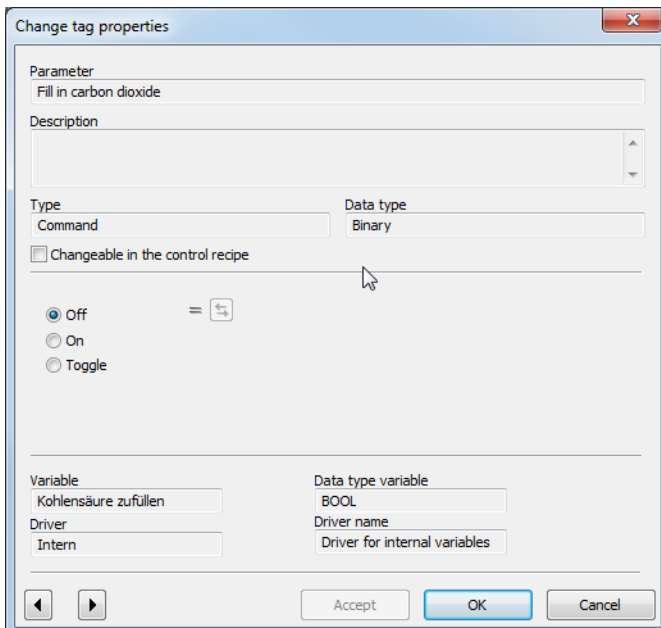
Driver name  
Driver for internal variables

Accept OK Cancel

Parameters	Meaning
Parameters	Name of the TAG. Display only.
Description	Free description of the tag. Display only.
Type	Type of the tag: <code>Command</code> or <code>Return</code> . Display only.
Data Type	Data type of the tag. Display only.
Changeable in the control recipe	<p><b>Active</b> : Value of the property can be changed in the control recipe.</p> <p>Only available if:</p> <ul style="list-style-type: none"> <li>▶ the tag is of type <code>command</code></li> <li>▶ it was configured in the Editor that the tag can be changed in the Batch recipe (property: ). <b>Changeable in master recipe</b>.</li> </ul>
Minimum	<p>Minimum value for the tag.</p> <p>Default from the Editor can be changed when property <b>Changeable in master recipe</b> was activated in the Editor. If the value is changed, it is marked with the symbol (on page 157) on the right side of the value. Changed values can be overwritten with the default from the Editor with the help of the button right next to it. For control strategies, synchronization takes place with the values that were set in the control strategy when linking parameters.</p>
Minimum variable	Allowed minimum value of the variable.
Value	<p>Value of the tag.</p> <p>Default from the Editor can be changed when property <b>Changeable in master recipe</b> was activated in the Editor. If the value is changed, it is marked with the symbol (on page 157) on the right side of the value. Changed values can be overwritten with the default from the Editor with the help of the button right next to it. For control strategies, synchronization takes place with the values that were set in the control strategy when linking parameters.</p>
Unit	Unit of the value.
Maximum	<p>Maximum value for the tag. Display only.</p> <p>Default from the Editor can be changed when property <b>Changeable in master recipe</b> was activated in the Editor. If the value is changed, it is marked with the symbol (on page 157) on the right side of the value. Changed values can be overwritten with the default from the Editor with the help of the button right next to it. For control strategies, synchronization takes place with the values that were set in the control strategy when linking parameters.</p>

<b>Maximum variable</b>	Allowed minimum value of the variable. Display only.
<b>Variable</b>	Variable which is linked to the tag. Display only.
<b>Data type variable</b>	Data type of the variable. Display only.
<b>Drivers</b>	Driver of the variable. Display only.
<b>Driver name</b>	Description of the driver of the variable. Display only.
Cursor keys	<p>Navigating through the tags.</p> <p>They are displayed in the order of the list. At this only tags are displayed which are visible with the current filter and grouping.</p> <p>If changes were done, there is a prompt before you can change to another tag whether the changes should be applied or discarded. If changes should be applied, the input is checked before advancing.</p>
<b>Apply</b>	Applies all changes if the check of the changes was successful. The dialog remains open for further editing.
<b>OK</b>	Applies all changes and closes the dialog if the check of the changes was successful.
<b>Cancel</b>	Discards all changes which have not been taken over yet and closes the dialog.

## BINARY TAG

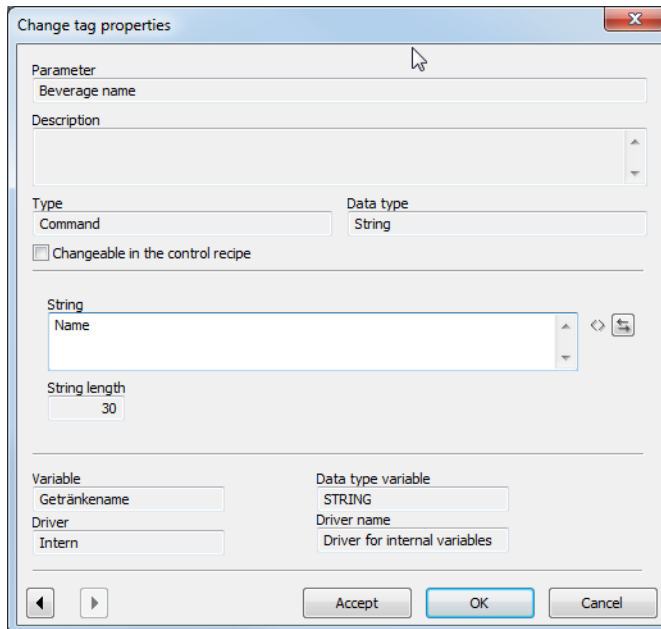


The screenshot shows the 'Change tag properties' dialog box. The 'Parameter' field contains 'Fill in carbon dioxide'. The 'Description' field is empty. The 'Type' is set to 'Command' and the 'Data type' is 'Binary'. There is a checkbox for 'Changeable in the control recipe' which is currently unchecked. Below this, there are three radio buttons: 'Off' (selected), 'On', and 'Toggle'. To the right of these is a small icon with a double-headed arrow. At the bottom, there are four fields: 'Variable' (Kohlensäure zufüllen), 'Data type variable' (BOOL), 'Driver' (Intern), and 'Driver name' (Driver for internal variables). The 'Accept', 'OK', and 'Cancel' buttons are at the bottom right.

Parameters	Meaning
<b>Parameters</b>	Name of the TAG. Display only.
<b>Description</b>	Free description of the tag. Display only.
<b>Type</b>	Type of the tag: <code>Command</code> or <code>Return</code> . Display only.
<b>Data Type</b>	Data type of the tag. Display only.
<b>Changeable in the control recipe</b>	<p><b>Active</b> : Value of the property can be changed in the control recipe.</p> <p>Only available if:</p> <ul style="list-style-type: none"> <li>▶ the tag is of type <code>command</code></li> <li>▶ it was configured in the Editor that the tag can be changed in the Batch recipe (property: <code>).</code> <b>Changeable in master recipe</b>).</li> </ul>
<b>Off</b>	<p>Status: Off.</p> <p>Default from the Editor can be changed when property <b>Changeable in master recipe</b> was activated in the Editor. If the value is changed, it is marked with the symbol (on page 157) on the right side of the value. Changed values can be overwritten with the default from the Editor with the help of the button right next to it. For control strategies, synchronization takes place with the values that were set in the control strategy when linking parameters.</p>
<b>On</b>	Status: On.
<b>Toggle</b>	Toggles between the states.
<b>Variable</b>	Variable which is linked to the tag. Display only.
<b>Data type variable</b>	Data type of the variable. Display only.
<b>Driver</b>	Driver of the variable. Display only.
<b>Driver name</b>	Description of the driver of the variable. Display only.
<b>Cursor keys</b>	<p>Navigating through the tags.</p> <p>They are displayed in the order of the list. At this only tags are displayed which are visible with the current filter and grouping.</p> <p>If changes were done, there is a prompt before you can change to another tag whether the changes should be applied or discarded. If changes should be applied, the input is checked before advancing.</p>
<b>Apply</b>	Applies all changes if the check of the changes was successful. The dialog remains open for further editing.
<b>OK</b>	Applies all changes and closes the dialog if the check of the changes was successful.
<b>Cancel</b>	Discards all changes which have not been taken over yet and

closes the dialog.

## STRING TAG



The screenshot shows the 'Change tag properties' dialog box for a String tag. The dialog has a title bar with a close button. The main content area is divided into several sections:

- Parameter:** A text field containing 'Beverage name'.
- Description:** A large text area with a vertical scrollbar.
- Type:** A dropdown menu set to 'Command'.
- Data type:** A dropdown menu set to 'String'.
- Changeable in the control recipe:** A checkbox that is currently unchecked.
- String:** A section containing a text field labeled 'Name' with the value 'Name', a vertical scrollbar, and a small icon to the right.
- String length:** A text field containing the value '30'.
- Variable:** A section containing two text fields: 'Getränkename' and 'Intern'.
- Data type variable:** A section containing two text fields: 'STRING' and 'Driver for internal variables'.

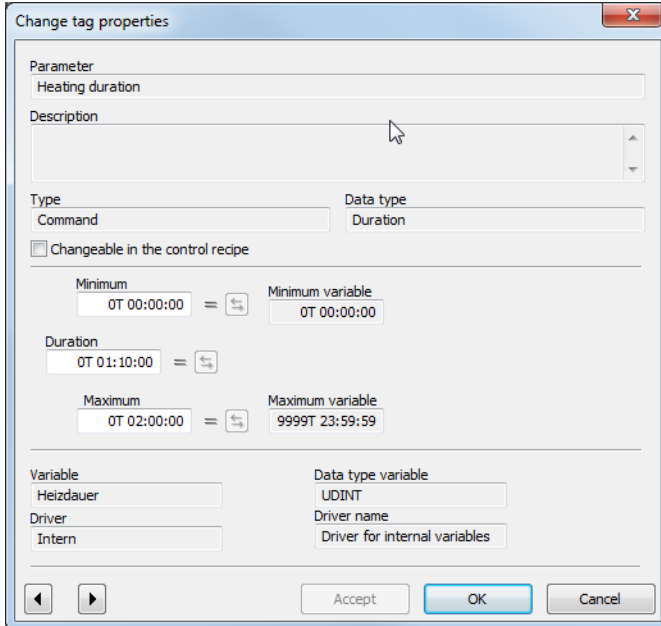
At the bottom of the dialog, there are three buttons: 'Accept', 'OK' (highlighted with a blue border), and 'Cancel'. There are also navigation arrows on the left side of the bottom section.



Parameters	Meaning
<b>Parameters</b>	Name of the TAG. Display only.
<b>Description</b>	Free description of the tag. Display only.
<b>Type</b>	Type of the tag: <code>Command</code> or <code>Return</code> . Display only.
<b>Data Type</b>	Data type of the tag. Display only.
<b>Changeable in the control recipe</b>	<p><b>Active</b> : Value of the property can be changed in the control recipe.</p> <p>Only available if:</p> <ul style="list-style-type: none"> <li>▶ the tag is of type <code>command</code></li> <li>▶ it was configured in the Editor that the tag can be changed in the Batch recipe (property: <code>).</code> <b>Changeable in master recipe</b>).</li> </ul>
<b>String</b>	<p>Alphanumeric character string.</p> <p>Default from the Editor can be changed when property <b>Changeable in master recipe</b> was activated in the Editor. If the value is changed, it is marked with the symbol (on page 157) on the right side of the value. Changed values can be overwritten with the default from the Editor with the help of the button right next to it. For control strategies, synchronization takes place with the values that were set in the control strategy when linking parameters.</p> <p>Possible length is limited by the <b>String length</b> engineered in the variable.</p>
<b>String length</b>	Defines possible length of the string. Display only.
<b>Variable</b>	Variable which is linked to the tag. Display only.
<b>Data type variable</b>	Data type of the variable. Display only.
<b>Driver</b>	Driver of the variable. Display only.
<b>Driver name</b>	Description of the driver of the variable. Display only.
<b>Cursor keys</b>	<p>Navigating through the tags.</p> <p>They are displayed in the order of the list. At this only tags are displayed which are visible with the current filter and grouping.</p> <p>If changes were done, there is a prompt before you can change to another tag whether the changes should be applied or discarded. If changes should be applied, the input is checked before advancing.</p>
<b>Apply</b>	Applies all changes if the check of the changes was successful. The dialog remains open for further editing.
<b>OK</b>	Applies all changes and closes the dialog if the check of the changes was successful.

**Cancel**

Discards all changes which have not been taken over yet and closes the dialog.

**TIME PERIOD**

The dialog box "Change tag properties" is shown. It contains the following fields and options:

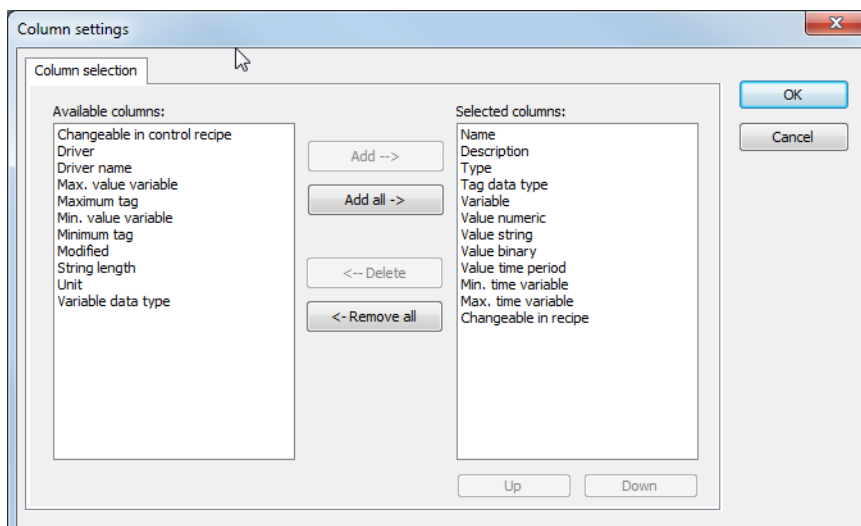
- Parameter:** Heating duration
- Description:** (empty text area)
- Type:** Command
- Data type:** Duration
- ☐ Changeable in the control recipe
- Minimum:** OT 00:00:00
- Minimum variable:** OT 00:00:00
- Duration:** OT 01:10:00
- Maximum:** OT 02:00:00
- Maximum variable:** 9999T 23:59:59
- Variable:** Heizdauer
- Data type variable:** UDINT
- Driver:** Intern
- Driver name:** Driver for internal variables

At the bottom, there are navigation arrows (left and right) and three buttons: Accept, OK, and Cancel.

Parameters	Meaning
Parameters	Name of the TAG. Display only.
Description	Free description of the tag. Display only.
Type	Type of the tag: <code>Command</code> or <code>Return</code> . Display only.
Data Type	Data type of the tag. Display only.
Changeable in the control recipe	<p><b>Active</b> : Value of the property can be changed in the control recipe.</p> <p>Only available if:</p> <ul style="list-style-type: none"> <li>▶ the tag is of type <code>command</code></li> <li>▶ it was configured in the Editor that the tag can be changed in the Batch recipe (property: ). <b>Changeable in master recipe</b>.</li> </ul>
Minimum	<p>Minimum value for the tag in format: <b>d hh:mm:ss</b></p> <p>Default from the Editor can be changed when property <b>Changeable in master recipe</b> was activated in the Editor. If the value is changed, it is marked with the symbol (on page 157) on the right side of the value. Changed values can be overwritten with the default from the Editor with the help of the button right next to it. For control strategies, synchronization takes place with the values that were set in the control strategy when linking parameters.</p>
Minimum variable	Allowed minimum value of the variable.
Duration	<p>Value of the parameter in the format: <b>T hh:mm:ss.</b></p> <p>Default from the Editor can be changed when property <b>Changeable in master recipe</b> was activated in the Editor. If the value is changed, it is marked with the symbol (on page 157) on the right side of the value. Changed values can be overwritten with the default from the Editor with the help of the button right next to it. For control strategies, synchronization takes place with the values that were set in the control strategy when linking parameters.</p>
Unit	Unit of the value.
Maximum	<p>Maximum value for the tag in format: <b>d hh:mm:ss.</b></p> <p>Default from the Editor can be changed when property <b>Changeable in master recipe</b> was activated in the Editor. If the value is changed, it is marked with the symbol (on page 157) on the right side of the value. Changed values can be overwritten with the default from the Editor with the help of the button right next to it. For control strategies,</p>

	synchronization takes place with the values that were set in the control strategy when linking parameters.
<b>Maximum variable</b>	Allowed minimum value of the variable. Display only.
<b>Variable</b>	Variable which is linked to the tag. Display only.
<b>Data type variable</b>	Data type of the variable. Display only.
<b>Driver</b>	Driver of the variable. Display only.
<b>Driver name</b>	Description of the driver of the variable. Display only.
Cursor keys	<p>Navigating through the tags.</p> <p>They are displayed in the order of the list. At this only tags are displayed which are visible with the current filter and grouping.</p> <p>If changes were done, there is a prompt before you can change to another tag whether the changes should be applied or discarded. If changes should be applied, the input is checked before advancing.</p>
<b>Apply</b>	Applies all changes if the check of the changes was successful. The dialog remains open for further editing.
<b>OK</b>	Applies all changes and closes the dialog if the check of the changes was successful.
<b>Cancel</b>	Discards all changes which have not been taken over yet and closes the dialog.

## Column selection

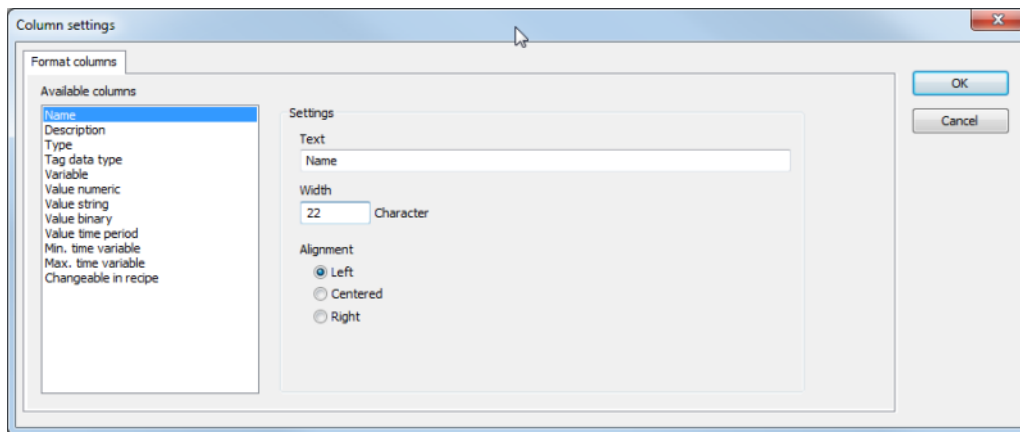


Button	Function
<b>Available columns</b>	List of columns that can be displayed in the table.
<b>Selected columns</b>	Columns that are displayed in the table.
<b>Add -&gt;</b>	Moves the selected column from the available ones to the selected items. After you confirm the dialog with OK, they are shown in the detail view.
<b>Add all -&gt;</b>	Moves all available columns to the selected columns.
<b>&lt;- Remove</b>	Removes the marked columns from the selected items and shows them in the list of available columns. After you confirm the dialog with OK, they are removed from the detail view.
<b>&lt;- Remove all</b>	All columns are removed from the list of the selected columns.
<b>Up</b>	Moves the selected entry upward. This function is only available for unique entries, multiple selection is not possible.
<b>Down</b>	Moves the selected entry downward. This function is only available for unique entries, multiple selection is not possible.

#### CLOSE DIALOG

Parameters	Description
<b>OK</b>	Applies settings and closes the dialog.
<b>Cancel</b>	Discards all changes and closes the dialog.
<b>Help</b>	Opens online help.

## Column Format



Parameter	Description
<b>Available columns</b>	List of the available columns via <b>Column selection</b> . The column selected here is configured using the settings in the <b>Parameters</b> section.
<b>Parameter</b>	Settings for selected column.
<b>Labeling</b>	Name for column title. The column title is online language switchable. To do this, you must enter the @ character in front of the name.
<b>Width</b>	Width of the column in characters.
<b>Alignment</b>	Alignment. Possible settings: <ul style="list-style-type: none"> <li>▶ Left-justified: Text is justified on the left edge of the column.</li> <li>▶ Centered: Text is displayed centered in the column.</li> <li>▶ Right-justified: Text is justified on the right edge of the column.</li> </ul>
<b>OK</b>	Applies settings and closes the dialog.
<b>Cancel</b>	Discards settings and closes the dialog.

## Insert operation

To insert an operation:

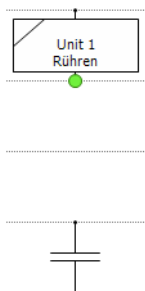
1. in the toolbar select the symbol for **Insert operation**
2. move the mouse to the desired position
3. place the operation
4. the dialog for selecting a template (on page 201) for the operation is opened
5. select the desired operation

6. The operation is inserted

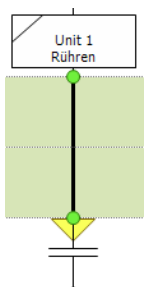
## Lines

Lines connect elements via free connection points. To connect connection points with each other:

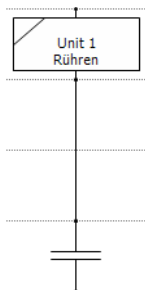
1. Activate a point with the help of the mouse:  
The connection point turns green. Red means that the connection point is already taken.



2. Drag a line to another connection point:  
A yellow arrow shows the direction of the line.. Green fields can be crossed. Red fields may not be crossed by the line.



3. As soon as the yellow tip of the line touches the next connection point, the line is created.



## USING TREND CURVES

Lines:

- ▶ are dragged with the mouse

- ▶ can be moved (press and hold key `Ctrl`)  
At this all existing connections are separated and it is tried to reconnect the line if there are objects with connections points in the right direction at the target.  
If several lines are highlighted, the line, in whose cell the mouse cursor is, is moved.
- ▶ can be deleted by highlighting them and pressing `Del`
- ▶ are deleted when re-dragging them from beginning to end
- ▶ have a tool tip displaying its ID.

If a line reaches a connection point of an object, the connection point becomes active. If a connection is possible, it turns green otherwise red. Connections connecting two connections points of the same type - two inputs, two outputs, etc. - are not allowed. The line can be added in any case. Not allowed connections are displayed in red and trigger a corresponding error message at testing.

The connection points of the elements are always displayed in the edit mode even if the connection point in question is connected. In status "Release" no connection points are displayed.

Properties connection point:

- ▶ connected: highlighted red; connection is separated when the line is dragged and a new connection point can be chosen
- ▶ open: highlighted green; at dragging a new line is created

## Transition

Transitions are used after phases in order to ensure a defined transition from one phase to another. Transitions display their internal status during the process and inform via a tool tip about status and process duration.

For details about transitions see section Engineering in the Editor (on page 15) in chapter Transitions (on page 27).

## Branches

Recipes can branch (on page 178) and run in simultaneous sequences (on page 179).

Branches and simultaneous sequences consist of:

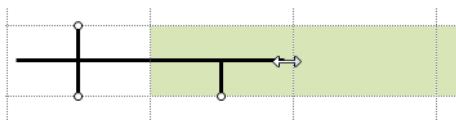
- ▶ Single or double horizontal lines
- ▶ Connection pieces (consisting of connection line and connection point)



## CREATE A BRANCH

To create a branch:

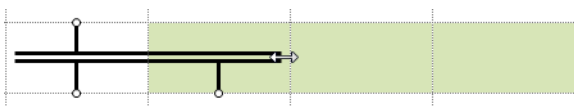
1. select the symbol **Insert begin branch**
2. put the branch on the desired location
3. connect the input connection point with a output connection point of the preceding object
4. connect both output connection points with the desired following objects
5. close a branch with object **Insert end branch**



## INSERT PARALLEL BRANCHES

To create a parallel branch:

1. select the symbol **Insert begin parallel branch**
2. put the branch on the desired location
3. connect the input connection point with a output connection point of the preceding object
4. connect both output connection points with the desired following objects
5. close a parallel branch with object **Insert end parallel branch**



## MODIFY AND MOVE

Branches and parallel branches can be moved and changed in size.

### MOVE

To move an object:

1. click on the object
2. keep the mouse button pressed
3. Move the object to the desired position

## CHANGE SIZE

In this way object **Begin/End branch/parallel branch** can be extended and shortened. To change their size:

1. move the mouse cursor over the object until it turns into a double arrow
2. press and hold the left mouse button and move the mouse in the desired direction:
  - away from the object to extend it
  - into the object to shorten it
3. at extending a new connection piece is added;
  - all fields which are concerned by the extension are marked green
  - to add several need connection pieces the process must be repeated
4. at shortening all corresponding connection pieces are deleted

## Branches

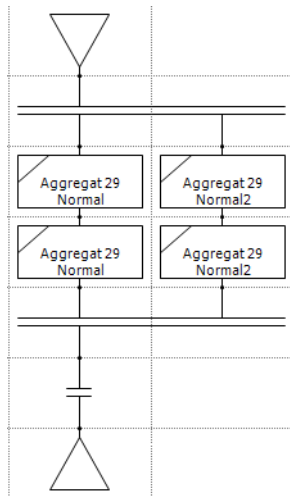
A branch offers the possibility to execute one of several possible ways. For this it is necessary that the first element at the beginning of a branch is a transition. This means that **Begin branch** can only be followed by a transitions (on page 27).

Procedure:

- ▶ The path is chosen for which the transition is **TRUE** first.
- ▶ Then it is waited until all transitions have a value.
- ▶ If several transitions are **TRUE** at the same time, always the leftmost path for which the transition is **TRUE** is selected.

For begin and end the following is true: If there is a phase in front of the element and a transition behind, the phase remains active until the transition was completed.

In a branch the objects are processed sequentially. Each branch processes its objects independent of other branches.



## Parallel branches

At the parallel branch an execution path parts into several execution paths which are executed in parallel during the process. For the activation of the different elements within a parallel branch you cannot define a certain order.

In the process the respective intermediate area of the **end parallel branch** is also colored. The color matches the coloring (on page 224) of the phase.

**Phase completed** is displayed as active as soon as the first previous element has been completed. This means that a phase is *Finished* or a transition is passed. Transitions are marked as completed as soon as they are passed. Phases wait at **Phase completed** until the **end parallel branch** is completed. Completed means that either the following phase is active or the following transition is inactive.

During the execution the status is color-coded.

## Split up and combine branches

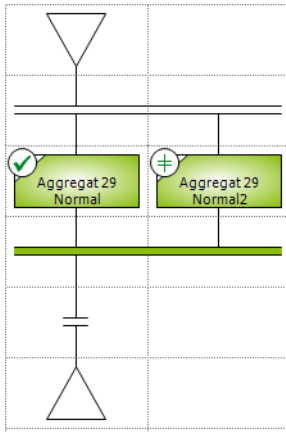
For parallel branches the branch splits up a **Begin parallel branch** and combines the single branches at **End parallel branch**. The paths of the parallel objects are independent branches. Only at **End simultaneous sequence** all branches are synchronized.

The possible branches are defined by the engineering. It is evident in Runtime if the separate branches are allocated or unlocked. A branch is active as long as an object on it is active.

The object types Begin branch, End branch and Jump target do not allocate and release branches as these objects are processed in the same branch. Combining branches is not allowed.

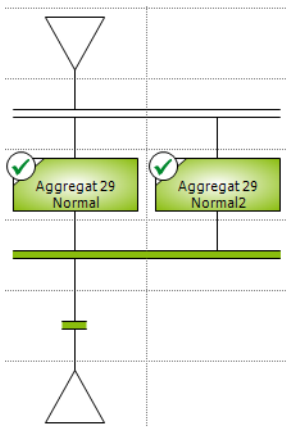
## END PARALLEL BRANCH

An **End parallel branch** combines the branches of the connected paths. The object after **End simultaneous sequence** is activated when all paths reaches **End simultaneous sequence** with their process.



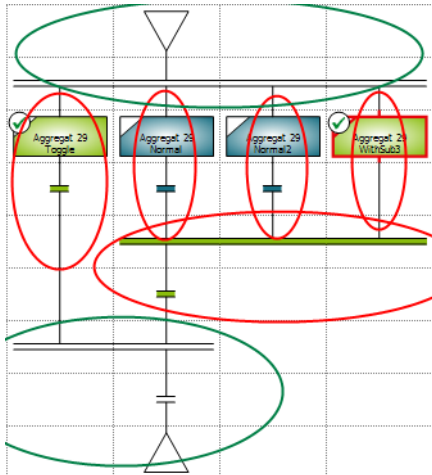
The left path is ready. Therefore **End simultaneous sequence** is already active. The transition after **End simultaneous sequence** is not yet active as the right branch is not yet completed.

When the right branch is also completed:



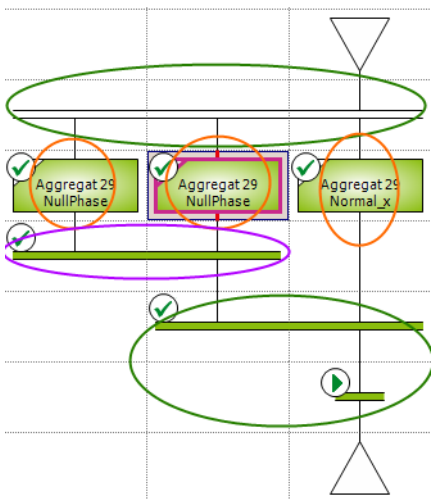
The transition after **End simultaneous sequence** is activated. All objects which were active before are still active. Instead of the transition there could also be another **End simultaneous sequence**.

### A LITTLE MORE COMPLEX ALTERNATIVE:



The areas highlighted in green are a branch.

### CASCADED END SIMULTANEOUS SEQUENCE:



The areas highlighted in green are a branch.

### Jump target

Adds a jump target.

Jump targets make it possible to

- jump between branches

- To jump out of branches
- engineer loops

Jump targets consist of tree inputs and one output. At this the output is always at the bottom and the inputs are located at the top and the sides. You can connect any input connection points. A path which ends in a jump target must have started with a **Begin branch**. Otherwise the end is not reached. Jump targets are not allowed for parallel branches.

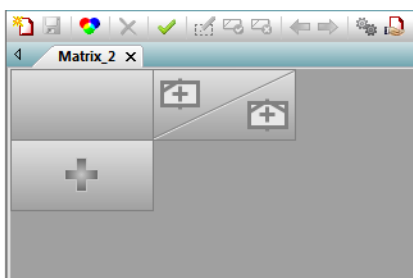
During the editing all connection points are visible. In the checking mode only the connection points which are connected are displayed.

### End element

Last element in the process. It is automatically created in the editor when a recipe is created and it cannot be deleted.

## 9.8.4 Matrix recipe

If you selected matrix recipe in dialog **Configuration master recipe** and exited the dialog with OK, the newly created recipe opens on a new tab in the matrix editor (on page 183).



## Matrix editor

In the matrix editor you can create your recipes and operations in a matrix.

## Batch Control

JobID 

### Master recipes

Master recipe name	Master recipe...	Master recipe s...	Version	Initial...	New...
Filter text	Filter text	Filter text	Filter...	Filter...	Open
Master2		Released	1		Edit
Master1		Released	1		Duplicate
Master2		Edit mode	2		Rename
M1		Edit mode	1		Test
					Release
					Delete
					new version
					Export...
					ol. selection...
					Format cols...

### Display control recipes

☒ prepared
 ☒ completed
 ☐ currently executed
 ☒ outdated
 ☐ Dynamically update
 [show control recipes](#)

### Control recipes

Master recipe name	Master recipe descr...	Master recipe s...	Version	New...
Filter text	Filter text	Filter text	Filter...	Open
Master1		Released	1	Duplicate
				Rename
				Delete
				Start
				Export...
				ol. selection...
				Format cols...

	Storage Tank 2 fill	Mixing Tank 2 add water	Mixing Tank 2 add sugar	Storage Tank 1 flush
1 Step	Active	Active	Active	Active
2 Step	Inactive	Inactive	Inactive	Active
3 Step	Inactive	Inactive	Inactive	Inactive

## TECHNICAL DETAILS

- ▶ Matrix:
  - Columns contain phases (on page 188) and operations (on page 189):  
Phases are inserted by clicking on the symbol above the diagonal  
Phases are inserted by clicking on the symbol below the diagonal
  - Lines contain steps with active/inactive phases.
- ▶ Insert column/row  
Clicking on the plus sign inserts a new line or column with a phase or operation into the matrix.
- ▶ Delete column/row Press the **Del** key to delete the highlighted row or column.
- ▶ Move lines/columns: Lines and columns can be moved via drag & drop with the help of the mouse. Individual labels of steps remain; the step number is adapted automatically.
- ▶ Border cells:
  - Double click on phase: opens the dialog (on page 188) for configuring the phase.

- Double click on step: opens the dialog for labeling (on page 187) the step.
- ▶ Selection of lines/columns:
  - left mouse click in border cell: selects a line/column.
  - Ctrl+mouse click in border cell: selects additional lines/columns.
- ▶ Cells:
  - left mouse click: selects empty cell.
  - Shift + click: activates/deactivates phase.
  - Ctrl+click: selects several cells.
  - Double click on cell: opens the dialog for configuring (on page 188) the phase.
- ▶ Scroll bar: Is displayed if the document is larger than the frame.
- ▶ Scrolling with a mouse wheel: up and down or, if you press and hold Shift, left and right.

## Toolbars matrix recipe

### TOOLBARS MATRIX RECIPE: EDIT MODE





<b>New master recipe...</b>	Opens the dialog for creating a new master recipe.
<b>Save master recipe</b>	Saves the master recipe which is open for editing. If another recipe is opened, the current recipe is saved automatically.
<b>Graphical design</b>	Opens the dialog (on page 131) to configure background colors, grid, and display of element ID.
<b>Delete</b>	Deletes the selected elements from the diagram. Multiple selection via <code>Ctrl+mouse click</code> .
<b>Check recipe for errors</b>	Checks recipe for errors and displays found errors in an information window. For several errors the first errors are displayed.  The error message contains the error number, the ID of the element, its location and a message in plain text.
<b>Edit element</b>	Opens the corresponding dialog for editing the selected element.
<b>Activate selected elements</b>	Activates selected elements.
<b>Deactivate selected elements</b>	Deactivates selected elements.
<b>Move selected columns to the left or move selected steps up</b>	Moves the selected column to the left by one position or moves selected step up by one position.
<b>Move selected columns to the right or move selected steps down</b>	Moves the selected column to the right by one position or moves selected step down by one position.
<b>Switch recipe to test mode</b>	Switches recipe to the test mode. For this the recipe must be without errors.
<b>Release recipe</b>	Releases the recipe. With this a control recipe can be created.

## TOOLBARS MATRIX RECIPE: TEST MODE



Parameter	Description
<b>Start recipe</b>	Starts the recipe process.
<b>Recipe pausing</b>	Pauses the recipe.
<b>Recipe resuming</b>	Resumes paused recipe.
<b>Recipe holding</b>	Holds recipe.
<b>Restart recipe</b>	Restarts held recipe.
<b>Recipe stopping</b>	Stops the recipe.
<b>Recipe aborting</b>	Aborts the recipe process.
<b>Phase pausing</b>	Pauses the phase.
<b>Phase resuming</b>	Resumes the process of a paused phase.
<b>Phase holding</b>	Holds phase.
<b>Restart phase</b>	Restarts held phase.
<b>Escape phase</b>	Starts process to exit from the phase.
<b>Check recipe for errors</b>	Starts recipe validation (on page 194).
<b>Edit element</b>	Opens the corresponding dialog for editing the selected element.
<b>Graphical design</b>	Opens the dialog (on page 131) to configure background colors, grid, and display of element ID.
<b>Switch to automatic mode</b>	Switches process to <code>automatic</code> mode.
<b>Switch to semi-automatic mode</b>	Switches process to <code>semi-automatic</code> mode.
<b>Switch to manual mode</b>	Switches process to <code>manual</code> mode.
<b>Continue recipe only on selected active elements</b>	Continues a recipe at the selected position.
<b>Continue recipe at all execution positions</b>	Continues a recipe on every available position.
<b>Skip active condition</b>	Skips an active condition. Only possible in the <code>manual</code> mode.
<b>Edit mode</b>	Switches from test mode to edit mode.
<b>Release recipe</b>	Releases the recipe. With this a control recipe can be created.

## MATRIX RECIPE TOOLBAR: APPROVED



Parameter	Description
<b>Check recipe for errors</b>	Checks recipe for errors and displays found errors in an information window. For several errors the first errors are displayed.  The error message contains the error number, the ID of the element, its location and a message in plain text.
<b>Edit element</b>	Opens the corresponding dialog for editing the selected element.
<b>Graphical design</b>	Opens the dialog (on page 131) to configure background colors, grid, and display of element ID.
<b>Duplicate recipe</b>	Only active if precisely one recipe was selected. Creates a copy of the selected recipe. At the creation of the copy, the version of the recipe saved on the hard disk is used. If the recipe is just edited in another computer and the changes have not yet been saved, the changes are not applied. The dialog for the input of a unique name and the description is opened.
<b>Create control recipe</b>	Creates a control recipe on the basis of the approved master recipe.

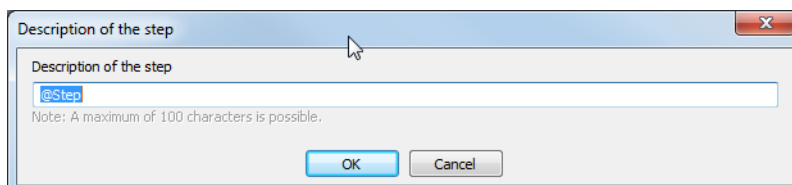


### Information

*The functions of the individual symbols can also be configured using buttons and thus be made touch-operable. Tool bars can therefore also be hidden (on page 117).*

## Name steps

Steps in the matrix editor can be named individually. Double click on the cell to open the dialog for entering an individual name.



The name can have up to 100 characters and is language switchable is preceded by a @.

## Add and configure phase

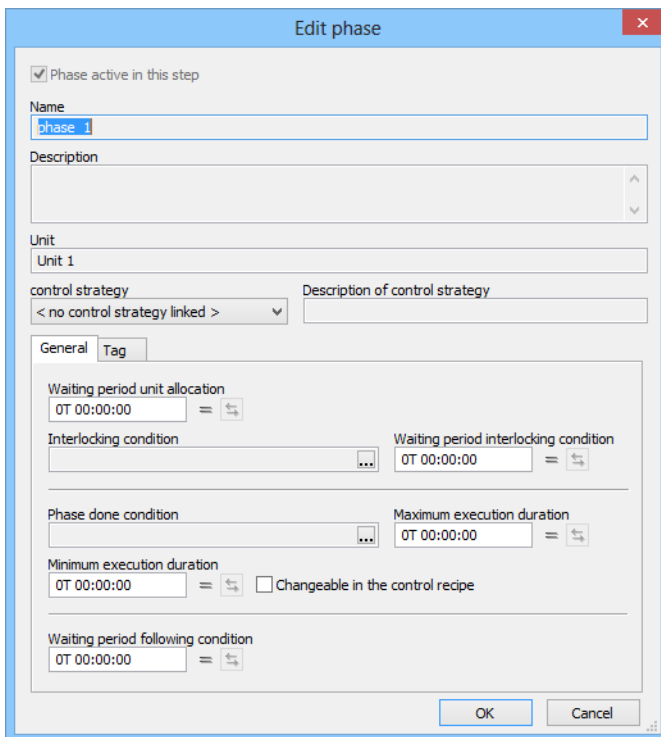
To add a phase:

1. Click on the plus sign above the diagonal in the last column of the matrix
2. the dialog for selecting a phase is opened
3. select the desired phase

To configure a phase:

1. double click the matrix field with the desired phase
2. The dialog for configuration is opened

## CONFIGURATION



The 'Edit phase' dialog box is shown with the following fields and options:

- ☒ Phase active in this step
- Name:
- Description:
- Unit:
- control strategy:
- Description of control strategy:
- General tab selected, Tag tab is also visible.
- Waiting period unit allocation:  =
- Interlocking condition:
- Waiting period interlocking condition:  =
- Phase done condition:
- Maximum execution duration:  =
- Minimum execution duration:  =  ☐ Changeable in the control recipe
- Waiting period following condition:  =
- OK and Cancel buttons at the bottom.

Parameters	Description
<b>Phase active in this step</b>	Active: Phase is active in this step
<b>Name</b>	Name of the phase. Only display.
<b>Description</b>	Comment about the phase. Only display.
<b>Unit</b>	Unit on which the phase is carried out. Only display.
<b>Control strategy</b>	Selection of a control strategy (on page 136) from a drop-down list. Only available if control strategies have been configured (on page 44) for this phase.  Default: no Control strategy linked
<b>Description of control strategies</b>	Description of the selected control strategy.  Display of the description entered in the Editor only.
<b>General</b>	Tab for configuration of general properties.
<b>Parameters</b>	Tab for configuration of parameters.
<b>OK</b>	Applies all changes on all tabs and closes the dialog.
<b>Cancel</b>	Discards all changes on all tabs and closes the dialog.

Configuration of the tabs see section PFC editor (on page 147)/chapter:

- ▶ General: (on page 157) Display and configuration of the settings for the phase
- ▶ Parameter (on page 162): Configuration of the tags

## RULES FOR EDITING A PHASE

A phase can be edited:

- ▶ In a master recipe in edit mode: If the user has sufficient rights.
- ▶ In a master recipe in test mode: If the user has sufficient rights and the phase is not active.
- ▶ In a control recipe with `prepared` status: If the user has sufficient rights and the **Changeable in the control recipe** option has been activated.
- ▶ In a control recipe with `running` status: If the user has sufficient rights, the **Changeable in the control recipe** option has been activated and the phase is not active.

The phase can no longer be edited in pre-configured control recipes and in approved master recipes.

## Insert operation

To insert an operation (on page 194):

1. Click on the plus sign below the diagonal in the last column of the matrix
2. the dialog for selecting a template (on page 201) for operation is opened
3. Select the desired template
4. A new operation is inserted

### 9.8.5 Master recipe - test mode

The test mode is used to test master recipes without releasing it and creating control recipes. In addition in the test mode changes in the Editor can be applied directly via reloading the Runtime.

**Exception:** During the execution of a recipe, the reloading of a recipe is delayed. Not until the recipe is finished, stopped or aborted, the reloading process is executed.

In test mode you cannot can the principle recipe process. You can only change values of the command tags. Changes are directly saved in the master recipe. It is not necessary to save explicitly.



Parameters	Description
<b>Start recipe</b>	Starts the recipe process.
<b>Recipe pausing</b>	Pauses the recipe.
<b>Recipe resuming</b>	Resumes paused recipe.
<b>Recipe holding</b>	Holds recipe.
<b>Restart recipe</b>	Restarts held recipe.
<b>Recipe stopping</b>	Stops the recipe.
<b>Recipe aborting</b>	Aborts the recipe process.
<b>Phase pausing</b>	Pauses the phase.
<b>Phase resuming</b>	Resumes the process of a paused phase.
<b>Phase holding</b>	Holds phase.
<b>Restart phase</b>	Restarts held phase.
<b>Check recipe for errors</b>	<p>Checks recipe for errors and displays found errors in an information window. For several errors the first errors are displayed.</p> <p>The error message contains the error number, the ID of the element, its location and a message in plain text.</p>
<b>Edit element</b>	Opens the corresponding dialog for editing the selected element.
<b>Graphical design</b>	Opens the dialog (on page 131) to configure background colors, grid, and display of element ID.
<b>Switch to automatic mode</b>	Switches process to <code>automatic</code> mode.
<b>Switch to semi-automatic mode</b>	Switches process to <code>semi-automatic</code> mode.
<b>Switch to manual mode</b>	Switches process to <code>manual</code> mode.
<b>Continue recipe only on selected active elements</b>	Continues a recipe at the selected position.
<b>Continue recipe at all execution positions</b>	Continues a recipe on every available position.
<b>Skip active condition</b>	<p>Skips an active condition.</p> <p>Only possible in the <code>manual</code> mode.</p>
<b>Switch recipe to edit mode</b>	Switches from test mode to edit mode.
<b>Release recipe</b>	Releases the recipe. With this a control recipe can be created.

### 9.8.6 Release master recipe

You can release a master recipe by selecting it and clicking button **Release master recipe**. Several recipes can also be selected and approved together. Approval must be confirmed by means of a dialog.

As soon as a recipe has been approved, the dialog to allocate the unit can no longer be opened. All information about the unit is displayed in the tooltip. Dialogs for transitions and phases can continue to be opened as write-protected.



#### Information

Only master recipes without errors can be released. A released master recipe can no longer be edited.

For each recipe you can create a copy of the released master recipe by clicking on button **Duplicate recipe**. This copy can then be edited.

Recipes can only be approved if all operations contained therein have also been approved.

### 9.8.7 Highlight recipe as outdated

Recipes that are no longer valid but have not been deleted should be marked as outdated. If a recipe is set to this status, it can no longer be edited or approved. No control recipe can be created on the basis of this recipe either. The recipe can however be duplicated and thus be used as the basis for new master recipes.

Only recipes that have the status `approved` can be marked as outdated. The following applies for attendant control recipes:

- ▶ Control recipes that are currently being executed continue to be executed
- ▶ Control recipes with the `prepared` status can no longer be executed

### 9.8.8 Versioning for master recipes

Master recipes can also be versioned. In doing so, a copy of an approved or obsolete report is created. This copy is in edit mode and contains a unique version number. The new recipe can be edited, but not renamed. Individual versions, including the source recipe itself, can be deleted.



## CREATING A VERSION

To use versioning in Runtime:

1. In the Editor, navigate to the **General/Versioning** properties group in the Batch Control node.
2. Activate the **Versioning active** property.
3. Versioning is switched on and used in Runtime.

To create a new version of a recipe in Runtime:

1. Select the desired master recipe.  
Note: The recipe must be approved or obsolete.
2. Select, in the context menu or on the toolbar, **Create new version** or click on the corresponding button in the screen.
3. A new recipe is created.

## RULES FOR VERSIONING

The following applies to versioning:

- ▶ A new version of a recipe contains the same name as the source recipe.
- ▶ New versions of a recipe cannot be renamed. Not even if the version is in editing mode and it is the only remaining version.
- ▶ The description can be changed for each version.
- ▶ Each version contains a unique version number that is issued on a serial basis. Version numbers of deleted recipes remain blocked and are not reissued.
- ▶ The version number of the new recipe and the version number of the source recipe are displayed in their own columns in the list of master recipes.
- ▶ Version numbers are also displayed in the title bar of the recipe editor and in the tab of the recipe as well as in the list of the control recipes and in tooltips.

## ONLY APPROVE ONE VERSION

It is possible to only allow one version of each master recipe to be approved. To do this, activate the **Only approve one version** property in the Editor. Only one version of each master recipe can be approved in Runtime. If a different version is approved, the one that was approved before must first be deleted or marked as obsolete.

## 9.9 Validate recipe

Recipes can be checked for error during the engineering. To validate a recipe, click on the corresponding symbol in the toolbar of the recipe editor in Runtime (green tick - **Check recipe for errors**). With this the recipe is checked for functionality according to internal rules. The following is especially checked:

- ▶ Syntax (all lines connected, processable from begin to end, etc.)
- ▶ Variables
- ▶ Datatypes
- ▶ Control Strategy: Linking of control strategies and value of the linked control strategy parameter to limits of the variable

The result of the check is displayed as pop-up in plain text. Found errors are also saved in the log file which can be analyzed with the Diagnosis Viewer.

Rules which must be adhered to during the engineering can be found in chapter Engineering rules for recipes (on page 132).



### Attention

*Operations that are not connected at the time of validation are ignored during validation. Their content and processes are not checked.*

## 9.10 Operations

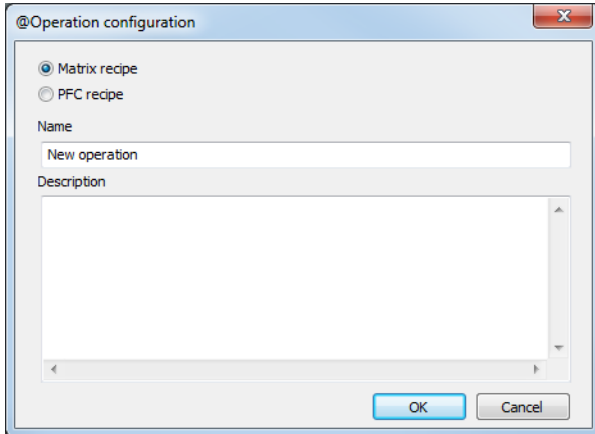
Operations form a substructure that can be embedded in recipes. This can provide a better overview in complex recipes. Operations are created in a similar manner to matrix recipes or PFC recipes. The operations are created on the basis of templates (on page 201) and as an instance in existing matrix recipes or PFC recipes.

### CREATING AN OPERATION

To create a new operation:

1. In the Recipe Editor (on page 116) switch the list of operations to visible (activate checkbox)
2. Select, in the toolbar or context menu (on page 197) of the list, the **New operation** command

### 3. The dialog for configuring an operation is opened



Parameter	Description
<b>Matrix recipe</b>	<p>Activate this radio button if you would like to create an operation on the basis of a matrix recipe (on page 182).</p> <p>Note: Only possible if the corresponding license is available.</p>
<b>PFC recipe</b>	<p>Activate this radio button if you would like to create an operation on the basis of a PFC recipe (on page 146).</p> <p>Note: Only possible if the corresponding license is available.</p>
<b>Name</b>	<p>Unique name for the operation. The name must not contain a dot (.), a question mark (?), a @ or an asterisk (*).</p> <p>Maximum length: 256 characters.</p> <p>Note: When you copy an operation the existing name is complemented with the prefix "<b>Copy of</b>". If the maximum length is exceeded by this, the name is shortened to the allowed length starting from the last character.</p> <p>The uniqueness is checked in the entire network. Therefore it can happen that you cannot take over the name as another user on another computer in the zenon network already has used the same name and you do not see the recipe in the list of the operations yet.</p> <p>The recipe names can be changed later but only as long as the recipe has the status <code>editable</code>.</p>
<b>Description</b>	<p>Optional description for the operation which should be created.</p> <p>The description can be changed later but only as long as the operation has the status <code>editable</code>. To change the description select the symbol <b>Rename operation</b>.</p>
<b>OK</b>	Applies all settings and created a new operation.
<b>Cancel</b>	Closes the dialog without creating an operation.

Configuration in the Recipe Editor is similar to the creation and configuration of PFC recipes (on page 146) and Matrix recipes (on page 182).

Changes to operations are only visible for the user in the operation instance if the operation instance has been saved.

## OPENING OPERATIONS

Existing operations can be opened by:

- ▶ Double-clicking on an operation in the list of the operations
- ▶ The **Open in Recipe Editor** command in the context menu of an operation
- ▶ Clicking on the **Open** symbol in the toolbar

## USING OPERATIONS

Operations can be inserted in recipes in the matrix editor or in the PFC editor and used as part of the recipe there.

You insert operations:

- ▶ In the PFC editor using the **Insert operation** (on page 174) symbol
- ▶ In the matrix recipe using the **Add operation** (on page 189) symbol

Phases and commands can be executed within operations. The position of the object in the recipe is also given in the CEL when a command is executed.

When inserting an operation into a recipe, the currently-saved version is always inserted. If the operation is open for editing, all unsaved changes are thus also not part of the inserted instance.

## SAVING OPERATIONS

Operations are, as instances, always part of the master recipe in which they are integrated. If an operation is saved, the attendant master recipe is also saved automatically.

## TOOLTIP

Operations have a tooltip. The following are displayed as soon as they have been executed once:

- ▶ Execution status:  
Contains current status and original status (status from which the operation comes). The original status is evident from the visual coloring and does not always correspond to the actual last status.
- ▶ Execution counter
- ▶ Error:  
Shows the number of objects that currently have an error status and the number of objects that have had an error status. Each object is always only used for one error.

- ▶ Exit from phase:  
Number of objects that are being exited from or have been exited from
- ▶ Overall duration with time when it was deactivated and deactivated
- ▶ Information about the status of the internal objects: Number of objects with the respective status (except **idle**)

### 9.10.1 Toolbar and context menu operations

#### TOOLBAR LIST

Parameter	Description
<b>New operation</b>	Creates a new operation.
<b>Open operation in Editor</b>	Opens the selected operation in the recipe editor.
<b>Rename the operation</b>	Opens the dialog to name a recipe.
<b>Duplicate operation</b>	Duplicates the selected operation.
<b>Delete operation</b>	Deletes the selected operation.
<b>Export selected XML</b>	Exports all the selected operation to an XML file.
<b>Import XML</b>	Imports the selected XML file as operation(s).
<b>Release operation</b>	Checks the selected operation and approves it if no errors were found.
<b>Column selection</b>	Opens the dialog to select a column (on page 65).
<b>Column Format</b>	Opens the dialog to format a column (on page 67).

## CONTEXT MENU

Parameter	Description
<b>New operation</b>	Creates a new operation.
<b>Rename</b>	Opens the dialog to name a recipe.
<b>Duplicate</b>	Duplicates the selected operation.
<b>Delete</b>	Deletes the selected operation.
<b>Export selected XML...</b>	Exports all the selected operation to an XML file.
<b>Import XML...</b>	Imports the selected XML file as operation(s).
<b>Open in Editor</b>	Opens the selected operation in the recipe editor.
<b>Release</b>	Checks the selected operation and approves it if no errors were found.

## TOOLBAR PFC EDITOR: EDIT MODE



Parameter	Description
<b>New operation...</b>	Opens the dialog for creating a new operation.
<b>Save operation</b>	Saves the master recipe which is open for editing.
<b>Graphical design</b>	Opens the dialog (on page 131) to configure background colors, grid, and display of element ID.
<b>Delete</b>	Deletes the selected elements from the diagram. Multiple selection via <code>Ctrl+mouse click</code> .
<b>Check operation for errors</b>	Checks recipe for errors and displays found errors in an information window. For several errors the first errors are displayed.  The error message contains the error number, the ID of the element, its location and a message in plain text.
<b>Edit element</b>	Opens the corresponding dialog for editing the selected element.
<b>Phase</b>	Opens dialog for selecting a phase and replaces the existing phase by the newly selected phase. Several phases can be selected and replaced together  Shortcut: <code>Shift+double click</code>
<b>Edit mode</b>	Toggles between insert mode and edit mode.
<b>Insert phase</b>	Adds a phase.
<b>Insert transition</b>	Adds a transition.
<b>Insert begin simultaneous sequence</b>	Adds a begin simultaneous sequence.
<b>Insert end simultaneous sequence</b>	Adds an end simultaneous sequence.
<b>Insert Begin branch</b>	Adds a begin branch.
<b>Insert End branch</b>	Adds an end branch.
<b>Insert unit allocation</b>	Adds a unit allocation.
<b>Insert jump target</b>	Adds a jump target.

<b>Release operation</b>	Releases the recipe. With this a control recipe can be created.
--------------------------	---

**Note:** Insertions remain active until you change to the edit mode using the **Edit mode** symbol, the `Esc` key or you change to another insert option via a symbol for adding a new element.

#### PFC RECIPE TOOLBAR: APPROVED

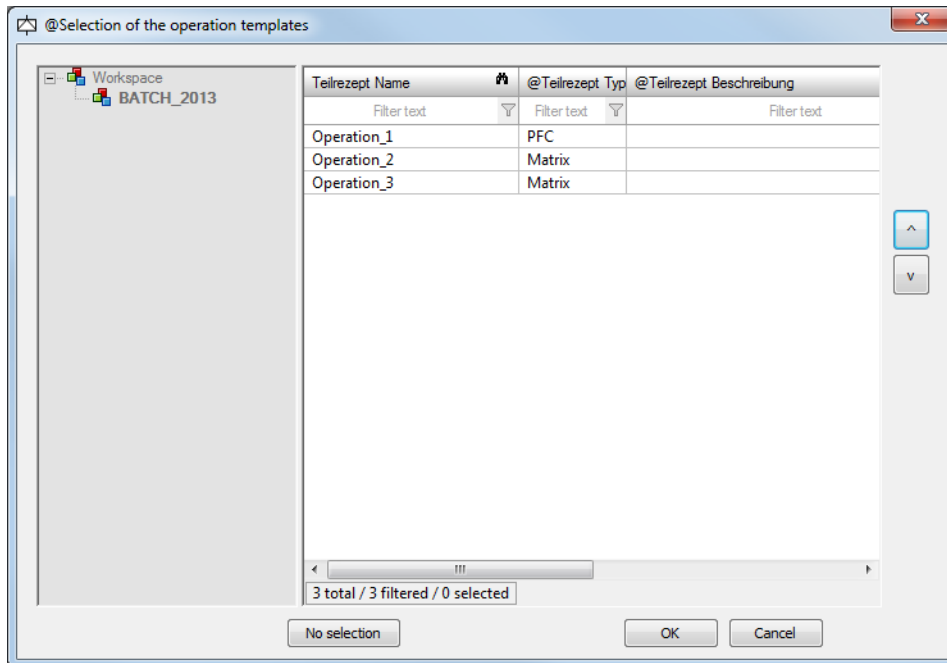


Parameter	Description
<b>Check operation for errors</b>	Checks recipe for errors and displays found errors in an information window. For several errors the first errors are displayed.  The error message contains the error number, the ID of the element, its location and a message in plain text.
<b>Edit element</b>	Opens the corresponding dialog for editing the selected element.  The dialog is opened in write-protected mode, because it is no longer possible to edit approved recipes.
<b>Graphical design</b>	Opens the dialog (on page 131) to configure background colors, grid, and display of element ID.
<b>Duplicate operation</b>	Only active if precisely one recipe was selected. Creates a copy of the selected recipe. At the creation of the copy, the version of the recipe saved on the hard disk is used. If the recipe is just edited in another computer and the changes have not yet been saved, the changes are not applied. The dialog for the input of a unique name and the description is opened.



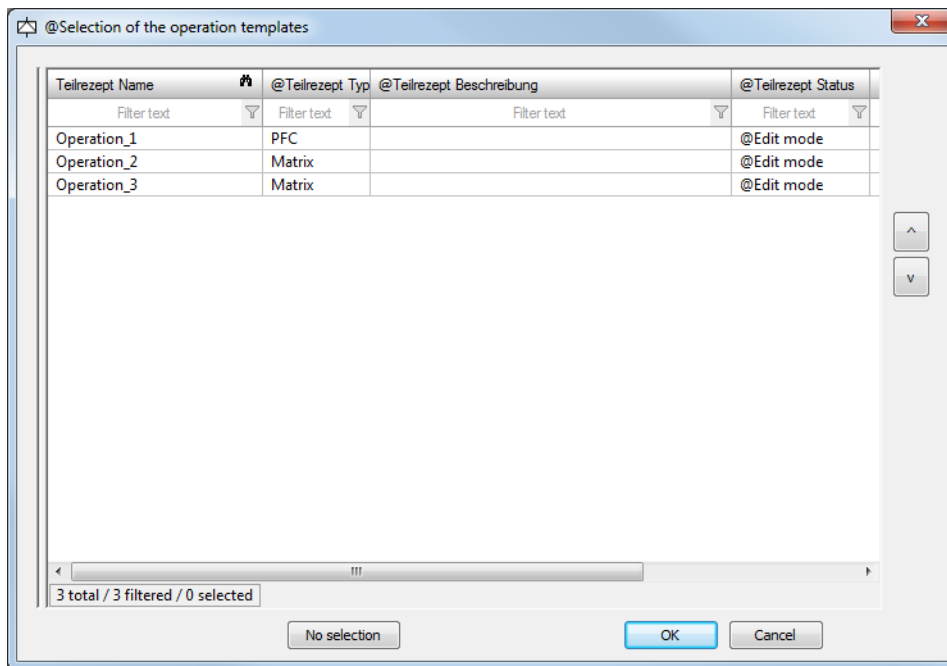
### 9.10.2 Selection of the template for an operation

Operations can be inserted into recipes as an instance. When inserting an operation into a matrix recipe (on page 189) or a PFC recipe (on page 174), the dialog to select a template for an operation is opened. This contains all previously-configured (on page 194) operations:



Parameters	Description
<b>List workspace</b>	<p>In the standard dialog to display the projects present in the workspace. No function in Batch Control, because operations can always be selected from the respective active project.</p> <p><b>Hint:</b> Drag this area down to the minimum size. It is then only displayed as minimized* in the future.</p>
<b>Operation list</b>	<p>Contains all operations that have been created. Any desired operations (matrix or PFC) can be selected for both editors (matrix and PFC).</p> <p>Entries can:</p> <ul style="list-style-type: none"> <li>▶ be sorted by clicking on the column title; another click inverts the sorting</li> <li>▶ be filtered into filter rows by alphanumeric entries (wildcards can be used)</li> </ul>
<b>Cursor keys</b>	<p>Move the selection:</p> <ul style="list-style-type: none"> <li>▶ Up</li> <li>▶ Down</li> </ul>
<b>Status line</b>	<p>Display:</p> <ul style="list-style-type: none"> <li>▶ Number of entries</li> <li>▶ Number of filtered entries displayed</li> <li>▶ Number of selected entries</li> </ul>
<b>No selection</b>	Cancels existing selection for existing operation and closes the dialog.
<b>OK</b>	Accepts selection, closes dialog and inserts operation.
<b>Cancel</b>	Discards selection and closes dialog without selecting or amending a template.

\*workspace minimized:



### 9.10.3 Status operation

The status shown for an operation always represents the status of the object with the highest priority in the operation.

The following applies to the colored identification:

- ▶ Non-transient status: the whole operation is colored
- ▶ Transient status: Original status and target status are displayed
- ▶ Holding as original status: Running (green) is always displayed

### PRIORITY

Priority of the objects in an operation, starting with the highest priority:

1. ABORTING
2. ABORTED
3. STOPPING
4. STOPPED
5. RESTARTING
6. HOLDING

- 7. HELD
- 8. PAUSING
- 9. PAUSED
- 10. RUNNING
- 11. COMPLETE
- 12. IDLE

#### 9.10.4 Symbol for execution

The symbols correspond to the symbols (on page 224) generally used the REE. Operations are symbolized in the REE by triangles in the left and right corner.

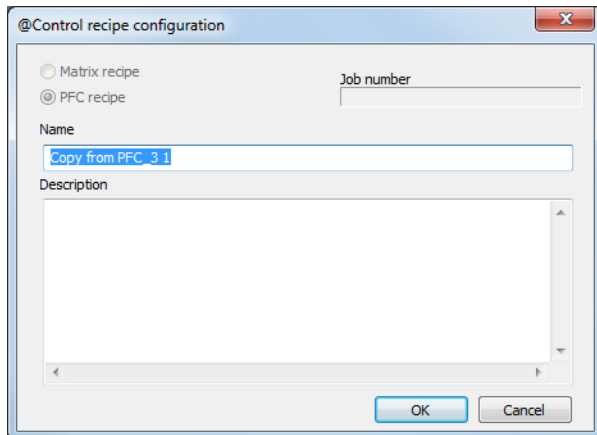
Symbol	Description	Tooltip
No symbol	Idle: No object is active in the operation.	Idle
	In execution: A symbol is always displayed whilst an operation is running. If objects are still running in the operation, the progress symbol is shown.	Execution of the internal objects.
	Phase finished: If objects are already active after the operation, the symbol for <b>following condition</b> is displayed.	Waiting for <b>Following conditions</b> .
	If an object has a different status within the operation, then a symbol is also displayed on the right.  There is a different status if an object has an empty status within the operation that does not correspond to the status of the operation and is not <code>idle</code> or <code>complete</code> .	

## 9.11 Control recipe

Control recipes control the progress of a recipe in the Runtime.

### 9.11.1 Create control recipe

You can create control recipes only based on released master recipes. Select the released master recipe in the list of the master recipes, which should serve as basis for your control recipe and click on button **New control recipe....**



Parameter	Description
<b>Matrix recipe</b>	Active: A matrix control recipe is created. Display only.
<b>PFC recipe</b>	Active: A PFC control recipe is created. Display only.
<b>Job ID</b>	The job ID provided by the job variables (on page 16).
<b>Name</b>	Unique name for the control recipe. The name must not contain a dot (.), a question mark (?), a @ or an asterisk (*).  The uniqueness is checked in the entire network. Therefore it can happen that a name is not accepted as another user already used the same name on another computer in the zenon network at the same time.  You can change the name afterwards as long as the recipe has status <i>Prepared</i> .
<b>Description</b>	Optional description of the recipe.
<b>OK</b>	Applies configuration and closes the dialog.
<b>Cancel</b>	<i>Discards entries and closes the dialog.</i>

If the control recipe was created using the symbol in the toolbar, then it is automatically opened in a new tab in the recipe editor provided this has been configured in the screen. If the control recipe has been created using the context menu, it is not opened in the recipe editor.

The newly created recipe is also displayed in the list of control recipes even if it does not match the set filter criteria.

## 9.11.2 Toolbar and context menu for control recipe list view

### TOOLBAR

Symbol	Description
<b>Open control recipe in Editor</b>	Opens the selected recipe in the recipe editor.
<b>Rename control recipe</b>	Opens dialog to rename the selected recipe.
<b>Duplicate control recipe</b>	Creates a copy of the selected recipe and opens the dialog to rename the duplicate.
<b>Delete control recipe</b>	Deletes selected recipes.
<b>Export selected XML</b>	Exports the selected control recipe as an XML file.
<b>Import XML</b>	Imports the selected XML file as operation(s).
<b>Start control recipe</b>	Starts selected control recipe.
<b>Column selection</b>	Opens the dialog for selecting the columns which should be displayed.
<b>Column Format</b>	Opens the dialog for configuring the column formats.

### CONTEXT MENU

Command	Description
<b>Open in Recipe Editor</b>	Opens the selected recipe in the Editor.
<b>Rename</b>	Opens dialog to rename the selected recipe.
<b>Duplicate</b>	Creates a copy of the selected recipe and opens the dialog to rename the duplicate.
<b>Delete</b>	Deletes selected recipes.
<b>Export selected XML...</b>	Exports the selected control recipe as an XML file.
<b>Import XML...</b>	Imports the selected XML file as operation(s).
<b>Starting</b>	Starts selected control recipe.

### 9.11.3 Control recipe edit mode toolbar



Parameter	Type	Description
<b>Start recipe</b>	Command	Starts the recipe process.
<b>Recipe pausing</b>	Command	Pauses the recipe.
<b>Recipe resuming</b>	Command	Resumes paused recipe.
<b>Recipe holding</b>	Command	Holds recipe.
<b>Restart recipe</b>	Command	Restarts held recipe.
<b>Recipe stopping</b>	Command	Stops the recipe.
<b>Recipe aborting</b>	Command	Aborts the recipe process.
<b>Phase pausing</b>	Command	Pauses the phase.
<b>Phase resuming</b>	Command	Resumes the process of a paused phase.
<b>Phase holding</b>	Command	Holds phase.
<b>Restart phase</b>	Command	Restarts held phase.
<b>Escape phase</b>	Command	Starts process to exit (on page 243) from the phase.
<b>Check recipe for errors.</b>	Action	Starts recipe validation (on page 194).
<b>Edit element</b>	Action	Opens the corresponding dialog for editing the selected element.
<b>Graphical design</b>	Action	Opens the dialog (on page 131) to configure background colors, grid, and display of element ID.
<b>Switch to automatic mode</b>	Command	Switches process to <code>automatic</code> mode.
<b>Switch to semi-automatic mode</b>	Command	Switches process to <code>semi-automatic</code> mode.
<b>Switch to manual mode</b>	Command	Switches process to <code>manual</code> mode.
<b>Continue recipe only on selected active elements</b>	Command	Continues a recipe at the selected position.
<b>Continue recipe at all execution positions</b>	Command	Continues a recipe on every available position.
<b>Skip active condition</b>	Command	Skips an active condition. Only possible in the <code>manual</code> mode.
<b>Duplicate recipe</b>	Action	Only active if precisely one recipe was selected. Creates a copy of the selected recipe. At the creation of the copy, the version of the recipe saved on the hard disk is used. If the recipe is



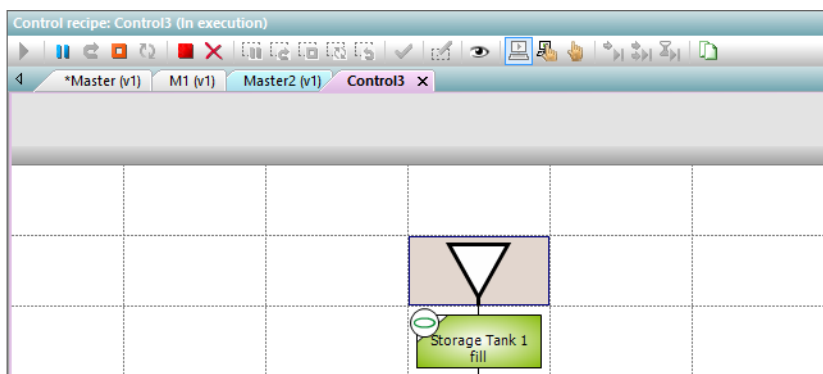
		<p>just edited in another computer and the changes have not yet been saved, the changes are not applied. The dialog for the input of a unique name and the description is opened.</p> <p>The copy of the recipe automatically gets the status <i>Prepared</i> and can therefore be edited and started. The execution status (on page 231) of the duplicate is set to <i>automatic</i>.</p> <p>When duplicating a recipe, a CEL entry is created.</p>
--	--	--

### 9.11.4 Execute control recipe

Control recipes can be started:

- ▶ after selecting a control recipe in the list of the control recipes:
  - via click on button **Start control recipe**
  - via click on menu item **Start** in the context menu
- ▶ via click on symbol **Start control recipe** in the toolbar if the control recipe is opened

PFC control recipe:



Matrix control recipe:




	TestUnit Test4	TestUnit Test2
1 @Step	TestUnit Test4	TestUnit Test2
2 @Step		TestUnit Test2
3 @Step		

**CHANGE VALUES**

If in the master recipe property **Changeable in the control recipe** was activated, certain values can be adapted in the control recipe as long as the phase is not active yet. In this case a button for synchronization is displayed next to the value. A click on this button take over the defined value from the master recipe.

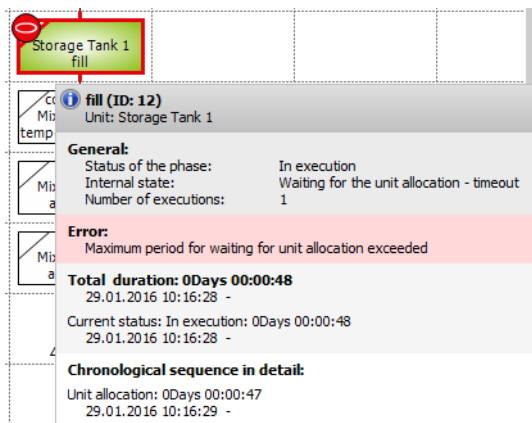
No more values can be changed in control recipes that have already been executed.

### MEANING OF THE SYMBOLS NEXT TO THE VALUES:

Symbol	Description
	Value in the control recipe and in the master recipe match.
	Value in the control recipe and in the master recipe are different.
	Click on button to apply the value from the master recipe. It overrides the value in the control recipe. Only active if the values in the control recipe and the master recipe do not match.

### TOOLTIP

A tool tip informs you about the current and historical events of an element.



The following is displayed:

- ▶ Element name
- ▶ General information about status and number of executions.  
Note: The counter becomes active if the phase is reactivated, including if the status is *restarting*, but the restart condition has not yet been run through.
- ▶ Error
- ▶ Total duration including times in status *Paused*.
- ▶ Timing

## 9.12 Synchronization

### CHECKING FOR CHANGES IN THE EDITOR

When loading, opening, duplicating or approving a recipe, a check is made to see if the configuration of the unit, phase, etc. was changed in the Editor. If a change is detected, it is taken over in the object to which the function concerned is assigned. Settings that were overwritten in Runtime are retained. When the recipe was changed, it is shown via an asterisk (\*).

At reloading the recipe is also checked.



#### Information

*Only recipes in edit mode or test mode are updated. Recipes in test mode that are currently running are only updated after execution has ended.*

If a changed recipe should be release which was not yet saved, a prompt is displayed with the following possibilities:

- ▶ to approve the current recipe
- ▶ to approve the saved recipe
- ▶ cancel the release process



#### Attention

*Changes to phases in the Editor are taken over without warning message when the recipe is releases in the Runtime. For all data which are not overwritten in the Runtime, the Editor is the leading system.*

### SYNCHRONIZATION OF PHASES

Editing of phases and their parameters is possible at four consecutive levels:

- ▶ in the zenon Editor
- ▶ In the template of the operation
- ▶ in the master recipe
- ▶ in the control recipe

When instancing, the data form the level above is always used. Synchronization and comparison in the editing dialogs also always relates to the previous level.

## ORIGIN OF DATA FOR COMPARISON OR SYNCHRONIZATION:

Position of phase	Phase that provides comparison data
in the zenon Editor	No data to compare.
in the master recipe	Phase configured in the zenon Editor.
in the control recipe	The phase from the master recipe.
In the template of the operation	Phase configured in the zenon Editor.
In the instance of the operation in the master recipe	The phase from the template of the operation.
In the instance of the operation in the control recipe	The phase from the instance of the operation in the master recipe.

You can find information on the origin of the comparison data in the tooltips of the control elements for synchronization.

## RECIPES AND OPERATIONS

The following is always the case when synchronizing operations:

- ▶ Master recipes are always synchronized with the editor data
- ▶ Then the instances of operations are synchronized with the data from their templates

Synchronization is carried out if:

- ▶ A recipe is opened,
- ▶ loaded,
- ▶ An operation template is saved and a recipe is opened that has an instance of this template

Values in phases and parameters correspond to the values in the templates as standard and can be overwritten locally.

## CHANGES TO CONTROL STRATEGIES AND CONTROL STRATEGY ACTIVATION

The following applies for control strategies (on page 136) during synchronization:

- ▶ When synchronizing recipes, phases with active control strategies are updated with the amended information (name change, new tags added, etc.).
- ▶ If control strategies were activated for a phase in the Editor, the phase in Runtime is set to the status where no control strategy is active. It contains only return parameters.

- ▶ If control strategies are deactivated for a phase in the Editor, the phase contains the complete current configuration of the phase after synchronization. With this switch, all changes that have been made to the phase in the recipe are discarded.

## 9.13 Manage recipes

The entire recipe management is done in the Runtime with the help of one or several screens of type `Batch Control`. Due to suitable filter settings you can achieve already filtered views for master recipes or control recipes.

In the screen different control elements (buttons, lists, editors) are available for different tasks. The screen is separated in three main areas which can be used and engineered in part completely independent of each other:

- ▶ Area master recipes (on page 215): Consists of a list and buttons for managing. The area can be used completely independently.
- ▶ Area control recipe (on page 219): Consists of a list and buttons for managing. The area can be used only together with the **list of the master recipes** as a master recipe must be selected first before the appertaining control recipes are displayed in the list.
- ▶ Recipe editors: Depending on the set recipe type the PFC editor (on page 147) or the matrix editor (on page 183) is used. The recipe editor needs either **List master recipe** or both lists for a recipe to be opened in it.

### 9.13.1 Manage master recipes

#### LIST AND LIST FORMATING

List/action	Description
<b>Master recipes list</b>	<p>In this list all master recipes can be displayed. The display can be limited by filters to an individual selection.</p> <p>The filtering can be preset in the zenon Editor in the screen switch function (on page 59). Online filtering is also possible. These filters are discarded when the screen is called up again. A permanent definition is only possible in the zenon Editor.</p> <p>All commands are also possible in the context menu of the list. The commands for list management can be called from the header of the list. The commands for recipe management can be called at editing one or more recipes.</p> <p>The recipes in the list cannot be edited directly in the list. Renaming, changing the description or changing the recipe status is only possible with the corresponding commands.</p> <p>Hint for ideal configuration of the list:</p> <p>The list can be designed diversely concerning content and look:</p> <ul style="list-style-type: none"> <li>▶ <b>Content:</b> The displayed columns (on page 65) can be selected, the column format (on page 67) (column width, alignment, label) can be changed and you can define filters (on page 71). These settings can be edited in Editor and Runtime.</li> <li>▶ <b>Look:</b> At the settings of the list in the Editor you can find diverse setting possibilities in areas <b>Representation</b>, <b>Scroll bars</b> and <b>Fill</b>. With these properties you can even design the list ready for touch operation.</li> </ul>

#### ACTIONS FOR LIST MANAGEMENT

Action	Description
<b>Column selection master recipe...</b>	<p>Opens a dialog in order to determine which columns should be displayed (on page 65).</p> <p><b>Attention:</b> These changes are discarded when the screen is called up again. A permanent definition is only possible in the zenon Editor.</p>
<b>Format columns master recipe...</b>	<p>Opens a dialog to edit the column settings (on page 67).</p> <p><b>Attention:</b> These changes are discarded when the screen is called up</p>

	again. A permanent definition is only possible in the zenon Editor.
--	---

## ACTIONS FOR RECIPE MANAGEMENT

Action	Description
<b>New master recipe...</b>	<p>Creates a new and completely empty master recipe in status <code>Editable</code>. The dialog (on page 140) for entering a unique name and a free description is displayed. The uniqueness of a name is also checked in the zenon network.</p> <p>Depending on the licensing, there may also be the possibility to select the recipe type: Matrix recipe (on page 182) or PFC recipe (on page 146). If only one of these recipe types is licensed, the licensed recipe type is fixedly set. The selected recipe type cannot be changed afterwards.</p> <p>When creating a master recipe, a CEL entry is created.</p>
<b>Create new version</b>	<p>Master recipes can also be versioned. In doing so, a copy of an approved or obsolete report is created. This copy is in edit mode and contains a unique version number. The new recipe can be edited, but not renamed. Individual versions, including the source recipe itself, can be deleted.</p>
<b>Duplicate master recipe</b>	<p>Only active if precisely one recipe was selected.</p> <p>Creates a copy of the selected recipe. At the creation of the copy, the version of the recipe saved on the hard disk is used. If the recipe is just edited in another computer and the changes have not yet been saved, the changes are not applied. The dialog for the input of a unique name and the description is opened.</p> <p>The copy of the recipe automatically receives status <code>Editable</code> and can be edited further.</p> <p>When duplicating a recipe, a CEL entry is created.</p>
<b>Delete master recipe</b>	<p>Deletes the selected recipes irrevocably. If the recipe is opened on another computer for editing, it is automatically closed there.</p> <p>Deleting is only possible if there are no control recipes which are based on the master recipe. First you must delete all control recipes.</p> <p>Recipes which are currently executed in test mode (master recipe status: Test in execution) cannot be deleted. First they must be <i>finished</i>, <i>stopped</i> or <i>canceled</i>.</p> <p>If recipes must not be deleted - e.g. in an FDA-regulated environment - it is recommended that this button is not configured or that it is given an appropriate <b>Authorization level</b>.</p> <p>A CEL entry is created when a recipe is deleted.</p>
<b>Export selected XML</b>	Exports the selected master recipe as an XML file.



<b>Import XML</b>	Imports the selected XML file as a master recipe.
<b>Rename master recipe</b>	<p>Only active if exactly one master recipe was selected. The dialog for the input of a unique name and the description is opened. Recipes can only be renamed if they are in status <code>Editable</code>. Also use this function in order to changed the description of the control recipe.</p> <p>When renaming a recipe, a CEL entry is created.</p>
<b>Open master recipe</b>	Opens the selected master recipe in the recipe editor if screen element <code>Recipe editor</code> exists in the screen. Each selected master recipe is opened in a separate tab of the recipe editor.
<b>Release master recipe</b>	<p>Changes the master recipe status of the selected recipes to <code>Released</code>. Only recipes without errors can be released. If error occur during the validation (on page 194), you must first fix them. Only recipes in status <code>Test mode</code> and <code>Editable</code> can be released.</p> <p>Released recipes can no longer be edited. Control recipes can only be created from released recipes. For details about the states see chapter Recipe types and recipe states (on page 135).</p> <p>When releasing a recipe, a CEL entry is created.</p>
<b>Test master recipe</b>	<p>Changes the master recipe status of the selected recipe to <code>Test mode</code>. Only faultless recipes can be switched to test mode. If error occur during the validation (on page 194), you must first fix them.</p> <p>Recipes in the test mode can be executed but no longer reengineered. For details about the states see chapter Recipe types and recipe states (on page 135).</p>
<b>Edit master recipe</b>	<p>Changes the master recipe status of the selected recipes to <code>Editable</code>. In this status, recipes can again be edited completely. Only recipes in <code>Test mode</code> can be set back to <code>Editable</code>.</p>

<b>Highlight master recipe as outdated</b>	Changes the status of the recipe to <code>outdated</code> . The recipe can no longer be edited or approved. No control recipe can be created on the basis of this recipe.
<b>New control recipe...</b>	<p>Opens the dialog (on page 205) for entering a unique name and a description for the control recipe. The uniqueness of the name is also checked in the zenon network. The name must only be unique within the master recipes. Control recipes which are based on other master recipes may have the same name. The uniqueness within module Batch Control is achieved by always referencing the master recipe name and the control recipe name.</p> <p>When creating a control recipe, a CEL entry is created.</p>

## ACTIONS FOR FILLING THE CONTROL RECIPE LIST

As each control recipe can be executed only once, we assume that there are very many control recipes. As during the loading of the list of the control recipes each control recipe is opened on the hard disk, it makes sense to not display all control recipes. Therefore control recipes cannot be opened automatically. They must be called up manually and via filters:

1. Provide the fitting filter options.
2. Select the desired master recipes.
3. Click on button **Display associated control recipes in list**.
4. All control elements complying with the filters and the selection are displayed in the list of the control recipes.

Action/filter	Description
<b>currently executed control recipes</b>	Opens only control recipes that are currently being executed. Control recipe status: <code>Running</code>
<b>prepared control recipes</b>	Opens only control recipes which are prepared for execution. Control recipe status: <code>Prepared</code>
<b>finished control recipes</b>	Opens only control recipes which have already been executed. Control recipe status: <code>Executed</code>
<b>Display associated control recipes in list</b>	Displays all control recipes that are based on the selected master recipe and that comply with the set filter criteria.

## 9.13.2 Manage control recipes

### LIST AND LIST FORMATING

List/action	Description
<b>Control recipe list</b>	<p>In this list all control recipes can be displayed. The display can be limited by filters to an individual selection.</p> <p>Per default the list is empty. To fill the list, you must:</p> <ul style="list-style-type: none"> <li>▶ select master recipes</li> <li>▶ Set the <b>currently-executed control recipes</b>, <b>prepared control recipes</b> and <b>completed control recipes</b> filters</li> <li>▶ click button <b>display associated control recipes in list</b></li> </ul> <p>In addition to the filters mentioned above, you can filter the list itself. The filtering can be preset in the zenon Editor in the screen switch function (on page 59). Online filtering is also possible. These filters are discarded when the screen is called up again. A permanent definition is only possible in the zenon Editor.</p> <p>All commands are also possible in the context menu of the list. The commands for list management can be called from the header of the list. The commands for recipe management can be called at editing one or more recipes.</p> <p>The recipes in the list cannot be edited directly in the list. Renaming, changing the description or starting the recipes is only possible with the corresponding commands.</p> <p><b>Hint for ideal configuration of the list</b></p> <p>The list can be designed diversely concerning content and look:</p> <ul style="list-style-type: none"> <li>▶ <b>Content:</b> The displayed columns (on page 65) can be selected, the column format (on page 67) (column width, alignment, label) can be changed and you can define filters (on page 71). These settings can be edited in Editor and Runtime.</li> <li>▶ <b>Look:</b> At the settings of the list in the Editor you can find diverse setting possibilities in areas <b>Representation</b>, <b>Scroll bars</b> and <b>Fill</b>. With these properties you can even design the list ready for touch operation.</li> </ul>

### ACTIONS FOR LIST MANAGEMENT

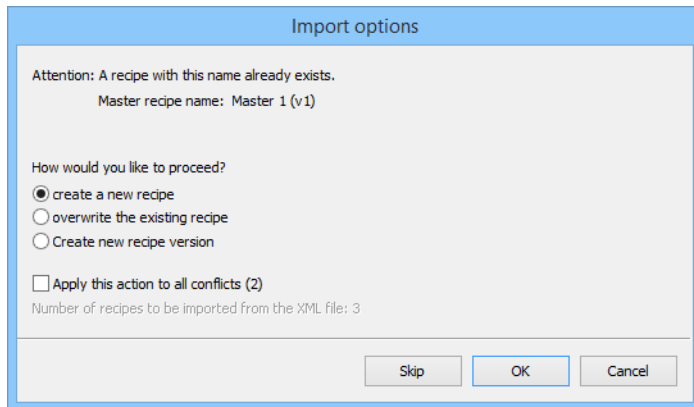
Action	Description
<b>Column selection master recipe...</b>	Opens a dialog in order to determine which columns should be displayed (on page 65). <b>Attention:</b> These changes are discarded when the screen is called up again. A permanent definition is only possible in the zenon Editor.
<b>Format columns master recipe...</b>	Opens a dialog to edit the column settings (on page 67). <b>Attention:</b> These changes are discarded when the screen is called up again. A permanent definition is only possible in the zenon Editor.

## ACTIONS FOR RECIPE MANAGEMENT

Action	Description
<b>Duplicate control recipe</b>	<p>Only active if precisely one recipe was selected.</p> <p>Creates a copy of the selected recipe. At the creation of the copy, the version of the recipe saved on the hard disk is used. If the recipe is just edited in another computer and the changes have not yet been saved, the changes are not applied. The dialog for the input of a unique name and the description is opened.</p> <p>The copy of the recipe automatically gets the status <code>Prepared</code> and can therefore be edited and started. The execution status (on page 231) of the duplicate is set to <code>automatic</code>.</p> <p>When duplicating a recipe, a CEL entry is created.</p>
<b>Delete control recipe</b>	<p>Deletes the selected recipes irrevocably. If the recipe is opened on another computer for editing, it is automatically closed there.</p> <p>Deleting is only possible if all selected recipes are not executed (control recipe status: <code>In execution</code>). In <code>execution</code>: First they must be finished, stopped or canceled.</p> <p>If recipes must not be deleted - e.g. in an FDA-regulated environment - it is recommended that this button is not configured or that it is given an appropriate <b>Authorization level</b>.</p> <p>A CEL entry is created when a recipe is deleted.</p>
<b>Rename control recipe</b>	<p>Only active if exactly one control recipe was selected.</p> <p>The dialog for the input of a unique name and the description is opened.</p> <p>Recipes can only be renamed if they are in status <code>Prepared</code>.</p> <p>Also use this function in order to changed the description of the control recipe.</p>
<b>Open control recipe</b>	Opens the selected control recipe in the recipe editor if screen element <code>Recipe editor</code> exists in the screen. Each selected control recipe is opened in a separate tab of the recipe editor.
<b>Start control recipe</b>	Starts the selected control recipe in the set execution mode. The recipes are executed invisibly at the Server. It is not necessary that the recipe is opened in the recipe editor.

### 9.13.3 Import recipes

When importing a recipe from an XML file, the following dialog is displayed in the event of naming conflicts:



Select how you want to proceed in the dialog:

Parameter	Description
<b>Master recipe name/control recipe/operation</b>	Name of the recipe that is already in use.
<b>Create a new recipe</b>	Creates a new recipe with the name and increments the number at the end of the name.
<b>Overwrite the existing recipe</b>	Overwrites the existing recipe with the name.
<b>Create a new recipe version</b>	Creates a new version of the recipe with the name. <b>Note:</b> Only active if <b>Versioning active</b> has been selected in the Editor
<b>Apply this action to all conflicts</b>	The selected option is applied to all pending conflicts. The number of conflicts is shown in brackets.
<b>Number of recipes to be imported from the XML file</b>	Number of recipes that are to be imported in the selected XML file.
<b>Skip</b>	Skips the importation of the currently-displayed recipe.
<b>OK</b>	Accepts the selection and shows the options for the next naming conflict or closes the dialog.
<b>Cancel</b>	Cancels the action and closes the dialog.

If there is a naming conflict there may, under certain circumstances, regardless of the status of the recipe that already exists, not be all important options available:

Recipe type	Status of the existing recipe.	Possible options
<b>Master recipe</b>	Edit mode	<ul style="list-style-type: none"> <li>► Create new recipe</li> <li>► Overwrite existing recipe</li> <li>► Create new version</li> </ul>
	Released	<ul style="list-style-type: none"> <li>► Create new recipe</li> <li>► Create new version</li> </ul>
	Outdated	<ul style="list-style-type: none"> <li>► Create new recipe</li> <li>► Create new version</li> </ul>
<b>Control recipe</b>	Prepared	<ul style="list-style-type: none"> <li>► Create new recipe</li> <li>► Overwrite existing recipe</li> </ul>
	In execution	<ul style="list-style-type: none"> <li>► Create new recipe</li> </ul>
	Executed	<ul style="list-style-type: none"> <li>► Create new recipe</li> </ul>
<b>Operation</b>	Edit mode	<ul style="list-style-type: none"> <li>► Create new recipe</li> <li>► Overwrite existing recipe</li> </ul>
	Released	<ul style="list-style-type: none"> <li>► Create new recipe</li> </ul>

**Note:** Control recipes cannot be imported to existing master recipes if the version number of the master recipe is different. The attendant master recipe from the control recipe to be imported must have the same version number as the existing master recipe.

### 9.13.4 Saving on the hard disk and backup scenarios

#### MASTER RECIPES

Each master recipe has a unique ID under which it is saved on the hard disk with file extension `.MR`; e.g. `9.MR`

Each recipe conforms to one file. The ID of the recipe can be read from the list of the master recipes. For this column **Master recipe ID** must be visible.

The folder for the master recipes is a sub folder of **Runtime folder**:

`\RT\FILES\zenon\system\BatchRecipes`

For the recipe management file `Recipe.unique` is responsible which is located in the same folder. It makes sure that the recipe names are unique.

**Note:** If you delete a recipe manually via the file explorer and therefore outside of the Runtime and the module Batch Control, you must delete file `Recipe.unique` for its content to be correct again. For example if you delete a control recipe manually, you cannot delete the respective master recipe in module Batch Control as the control recipe still exists according to module Batch Control. Only after a reinitialization of file `Recipe.unique` can the master recipe also be deleted.

## BACKING UP MASTER RECIPES

The `.MR` files - and with this all master recipes - can be backed up at any time. For example you can use function File operations.

## RESTORING MASTER RECIPES

The restoring should only be done if absolutely necessary as more current data is overwritten. Proceed as follows:

1. Exit the Runtime.
2. Save all existing master recipes.
3. Rename file `Recipe.unique` or delete it. It automatically re-created at the Runtime start from the `.MR` files.
4. Restore the `.MR` files from an earlier backup.
5. Restart the Runtime.

## CONTROL RECIPE

Each control recipe has a unique ID under which it is saved on the hard disk with file extension `.CR`; e.g. `9.CR`

Each recipe conforms to one file. The ID of the recipe can be read from the list of the control recipes. The **Control recipe ID** column must be visible to do this. Control recipes are always based on a master recipe and are therefore always assigned to it. The ID number circles are therefore only unique with regard to the underlying master recipe.

**Example:** The master recipe with ID 9 has the control recipes with IDs 1 and 2. The master recipe with ID 10 also has the control recipes with IDs 1 and 2.

Therefore each master recipe has a sub folder in which the control recipes are saved. The name of the folder is always: `<Master recipe ID>.crd`. In our example there is the folder `9.crd` with files `1.CR` and `2.CR` and the folder `10.crd` with files `1.CR` and `2.CR`.

The folder for the command recipes are sub folder of **Runtime folder**:

`\RT\FILES\zenon\system\BatchRecipes\`. In this folder the individual control recipe folders have been created. In each control recipe folder there is the file `Recipe.unique`. It makes sure that the recipe names are unique.

## BACKING UP AND RESTORING CONTROL RECIPES

Proceed in the same way as for the master recipes only that you now need to backup all `.CR` files and the appertaining folder structure. At restoring you must delete all `Recipe.unique` files. They are also restored automatically.

## 9.14 Recipe Execution Engine (REE)

















The REE (Recipe Execution Engine) executes recipes in the Runtime. You can start any number of recipes.

### 9.14.1 Symbols and Color

The states during the process of a phase are displayed with the help of different symbols. Some symbols are also used for transitions and end parallel branch.








## SYMBOLS FOR THE INNER STATUS:

Symbol	Meaning
	Phase starts.
	Waiting for communication with the controller
	Waiting for unit allocation. The unit of the phase is already being used in another recipe.
	Waiting for unit allocation expired.
	Waiting for exclusive execution. Another instance of the phase is active in the same recipe.
	Writing of the initial tags.
	Writing of the initial tags failed.
	Checking the input interlocking.
	Waiting for input lock expired.
	Writing the value tags.
	Writing the value tags has expired.
	Waiting for minimum execution period.
	Phase: Waiting for phase done condition. Transition: Waiting for transition condition. End parallel branch: Wait until execution is finished in all branches.
	Maximum waiting period expired.
	Waiting for following condition.
	Waiting time for following condition expired.
	After the <b>Restart</b> command when waiting for the restart condition.
	After the <b>Escape phase</b> command when waiting for the escape condition.



## SYMBOLS FOR ERRORS:

The symbols in the bottom left corner of the phase provide information on the error status of the phase. They can occur in any desired combination with the symbols for the inner status.

Symbol	Meaning
	There is a communication error.
	Communication error rectified, wait for acknowledgment of the communication error.
	Communication error rectified (and acknowledged if required)
	PLC error.
	PLC error rectified.

#### SYMBOLS FOR "ESCAPE PHASE":

The symbols in the bottom right corner of the phase provide information on the status when escaping from the phase.

Symbol	Meaning
	Waiting for escape condition.
	Escape condition met.

If an error occurs during a phase, the phase is marked as faulty until it is restarted.

#### STATUS

The execution status (on page 231) of **phases**, **transitions** and **end simultaneous sequence** is signaled in color:

Status	Color
Idle:	White
In execution:	green
Finished:	blue
Stopping:	Two colors: ‣ yellow ‣ Original color
Stopped:	yellow
Pausing:	Two colors: ‣ orange ‣ Original color
Paused:	orange
Holding:	Two colors: ‣ gray ‣ Original color
Held:	gray
Aborting:	Two colors: ‣ red ‣ Original color
Aborted:	red
Restarting:	Two colors: ‣ green ‣ Original color
Force: (phases or transitions only)	Violet border
Timeout:	red border

### ACTION ON STOP COMMAND

After a `stop` command, the **phases**, **transitions** and end **simultaneous sequence** immediately go to stopped status, even if other elements are still waiting for a condition for stopping. Further subsequent commands such as `cancel` are ignored. The stopped status remains displayed.

### 9.14.2 Create recipe image

During batch operation, all recipes that are currently being executed can be saved with all recipe data in a recipe image (Image) . Saving is possible as follows:

- ▶ `When Runtime is closed`: a recipe image of the recipe that is running is created automatically.
- ▶ `Cyclical`: in a freely-definable time period between 30 and 4294967295 seconds, an optional recipe image is written
- ▶ `When activating a phase`: if a phase is activated, an optional recipe image is written

These methods can also be combined as desired. It can thus be ensured that Batch operation can also be continued correctly at any time in the event of errors.

#### ENDING THE RECIPE IMAGE IN RUNTIME

If the `Trigger end of Runtime` event is triggered, the ending of Runtime is prevented until the Batch Control module has backed up all data. A process screen is created that represents the initial status for the restart. Likewise, it is ensured that the parameters of the **Write set value** action arrive at the control unit securely. Internally the phase is paused only when the writing confirmation from the driver ensued. This recipe image contains the images of the REEs, the order of the allocations and all reactions that are needed when restarting. Find out more information in the chapter Exit and restart Runtime.

#### CYCLIC WRITING OF A RECIPE IMAGE

A recipe image can also be written cyclically during Runtime. As soon as **Activate cyclic writing** is activated, a recipe image is created in the given cycle.

To activate cyclical writing:

1. Go, in the properties of the Batch Control module, to the **Recipe image** group.
2. Activate the property **Activate cyclic writing**
3. In the **Cycle time [s]** property, create the time period for the writing of the recipe image; the minimum time period is 30 seconds  
Note: If the cycle time is changed and reloaded, then the next time to which an image is written is recalculated.

With cyclical writing, the last two image files are always retained. Older ones are deleted. Writing is a two-stage process:

1. The recipe image is written to the **TemporaryImg.REE** file.
2. If this was successful, the next version number is issued and the temporary file is renamed to the new version.

With cyclically-created images, all recipes that are currently running are saved in the recipe image. This ensures that the recipe is appropriate to the execution status on restarting. To do this, the recipe from

the recipe image overwrites the recipe in Runtime when restarting. It is only possible to restart if the recipe in question still exists and is still in an execution status. The recipe is not restarted if the recipe has already ended after the recipe image has been created.

## RECIPE IMAGE WHEN ACTIVATING A PHASE

A recipe image can also be created when activating a phase. To create a recipe image each time a phase is activated:

1. Go, in the properties of the Batch Control module, to the **Recipe image** group.
2. Activate the property **Write recipe image when activating a phase**

A recipe image is written in Runtime each time a phase is activated.

## SAVING AND RESTORING

REE images are stored in the project folder with the following naming convention:

**Batch[Version-Hex].REE**

With the file extension **.REE**, image files are read in when Runtime is started and the most recent version is identified. The most recent version is the recipe image to be loaded and remains as a file after restarting. All other files with the extension **.REE** are deleted.

### 9.14.3 Behavior of elements in Runtime

The basic principle is: Phases and all elements that follow them (transition, end simultaneous sequence, start simultaneous sequence, allocation) remain active until the next phase becomes active. (Exception: a manual skip is carried out.)

## PHASES

### ALL PHASES PAUSED

In `manual` mode, it is possible to assign all phases the status `paused`.

**For example:** The branch continues to be gone through before the `end simultaneous sequence` after the end of this. The active element is thus after the `end simultaneous sequence` and before the next phase. All phases before the `end simultaneous sequence` have the status `paused`. These phases are now only set to `hold` and then to `restart`. The other phases remain `paused`. If the restarted phase is ended, there is only one active element. If this is activated with the next step, then no active element is present any more. The `end simultaneous sequence` element remains `paused` however and does not switch through.

**Solution:** Continuation of the paused phases.

## PAUSE AND RESUME

The following applies to pausing and resuming:

- ▶ Pausing and resuming with active element: A paused phase that is active for an active element is not continued.
- ▶ Switching from manual mode to automatic: All phases that are active for an active element are resumed.
- ▶ If a phase is `paused` in manual mode and the REE is switched to `automatic`, the phase remains paused. Global continuation would also not put this phase back to the status of `continue`, because the phase was paused in `manual` mode.
  - A `Pause phase` command, followed by a global pause and a global resumption, sets phases that were paused using the `Pause phase` command to the status `resume`.
  - Phases that were paused in `manual` mode can be set to the status of `resume` with the `Resume phase` command.
- ▶ The recipe status changes to `running` after global pausing and resumption. However phases with active elements remain paused and the active element remains unchanged.
- ▶ The recipe status remains as `running` after local pausing and resumption. The recipe is resumed at the active elements. This also applies if the active element has been moved.

## BRANCHES

The following applies in branchings:

- ▶ As long as the left transition does not have a value, the right transition is ignored.
- ▶ If the transition condition is met for both transitions, the left branch is selected.

## STOPPING ELEMENTS

If an **end simultaneous sequence** is stopped, it becomes inactive immediately and does not react to any more REE commands such as `cancel`. Therefore it also does not change to the status `aborted` after a `cancel` command. The same applies to transitions. In contrast, phases may wait after stopping for further conditions to be met.

## TRANSITIONS

- ▶ A transition before an end parallel branch remains active, including the phase before the transition after it has been run through, if the transition is active after the end parallel branch but has not yet been run through.

- ▶ In branches, impulses for a transition are ignored for as long as the transition to the left of the transition concerned does not have a valid value.
- ▶ Transitions are deactivated with `hold` and `restart`. The active element is activated again in manual or semi-automatic mode.

#### 9.14.4 Mode and mode change

The REE can run in three modes:

- ▶ `Automatic`: The recipe runs entirely automatically.
- ▶ `Semi-automatic`: The recipe is executed manually. Conditions cannot be jumped.
- ▶ `Manual`: Each step in the recipe or operation is executed manually; conditions that are being waited for can be skipped.

To execute a recipe manually or semi-automatically, the operation types (on page 233) **Step-by-step execution of the recipe** and **Jump** can be used.

When switching to `automatic` mode, all execution positions are removed. Global commands are only executed in branches that have no execution positions.



##### Information

*To react on serious events, you can change the mode during the running process via reaction type (on page 37) influencing the recipe.*

#### 9.14.5 The execution status

The following states are possible:

Status	Description
Idle	The REE is in idle state.
In execution	When starting a control recipe, it changes to the status <code>running</code> .
Executed	As soon as the execution is finished, the recipe changes to status <code>Finished</code> . In this status execution is not possible.
Pausing	The recipe changes to status <code>Paused</code> .
Paused	<p>Within the phase the process stops at:</p> <ul style="list-style-type: none"> <li>▶ Waiting for <code>Finished</code></li> <li>▶ Waiting for <b>Allocation</b></li> <li>▶ Waiting for <b>Interlocking condition</b></li> <li>▶ Waiting for <code>Phase finished</code></li> <li>▶ Check for parallel execution</li> </ul>
Holding	The object changes to <code>Held</code> and does not carry out any allocations anymore. When restarted, the object is restarted and changes to <code>running</code> .
Held	<p>Within the phase the process stops at:</p> <ul style="list-style-type: none"> <li>▶ Waiting for <code>Finished</code></li> <li>▶ Waiting for <b>Allocation</b></li> <li>▶ Waiting for <b>Interlocking condition</b></li> <li>▶ Waiting for <code>Phase finished</code></li> <li>▶ Check for parallel execution</li> </ul>
Restarting	Phase is restarting.
Restarted	Phase is completely restarted.
Stopping	Stops the process and changes to <code>Stopped</code> .
Stopped	The object was stopped.
Aborting	Aborts the process and changes to <code>Aborted</code> .
Aborted	<p>Recipe process was aborted.</p> <p>If a recipe cannot be restarted in the image at the restart, its status automatically changes to <code>Aborted</code>.</p>
Prepared	Prepared for execution.

## ACTIVE ELEMENT AND JUMP TARGETS

Status	Description
<b>Continue</b>	If an object is paused and an active element is located after it, <code>continue</code> has the same effect as <code>Next step</code> . This also includes jumps.



	At a phase command the command only effects a jump in the same branch.
<b>Held</b>	Removed <ul style="list-style-type: none"> <li>▶ With <b>phase</b>: execution positions in the branches</li> <li>▶ With <b>global</b>: all execution positions</li> </ul>
<b>Break</b>	Has now effects for jump targets. Already defined targets remain.
<b>Others</b>	Always causes the deletion of the jumps. For a phase command only the jump in the area of the phase is deleted.

### 9.14.6 Step-by-step execution of a recipe and jumps in the recipe

#### STEP-BY-STEP EXECUTION OF A RECIPE

A recipe can be executed step-by-step if:

- ▶ The recipe is in either semi-automatic or manual mode and
- ▶ The status of the recipe is `running`.

For the step-by-step execution the execution is held as soon as an element is finished with its execution. The holding is done via command `Pause` to the concerned execution path. As soon as all active elements in this path have reached the status of `paused`, the active element is marked by a red arrow. Operations are treated like all other objects of a recipe.

The execution is resumed with:

- ▶ a selective step: selection of the corresponding arrow (green)
- ▶ a global step: all positions with arrows for possible resuming are started

#### COMMUNICATION ERROR WITH STEP-BY-STEP EXECUTION

If an communication error occurs with a phase whilst this is waiting for an active element, the active element is no longer displayed until the problem is rectified. However, if in the meantime, the execution is switched to `automatic` mode, then the phase is no longer paused and must be continued. This also applies if a switch to manual or semi-automatic mode is made again.

## COMMANDS

### GLOBAL COMMANDS

For global commands all execution positions are deleted as the execution cannot be resumed from there.

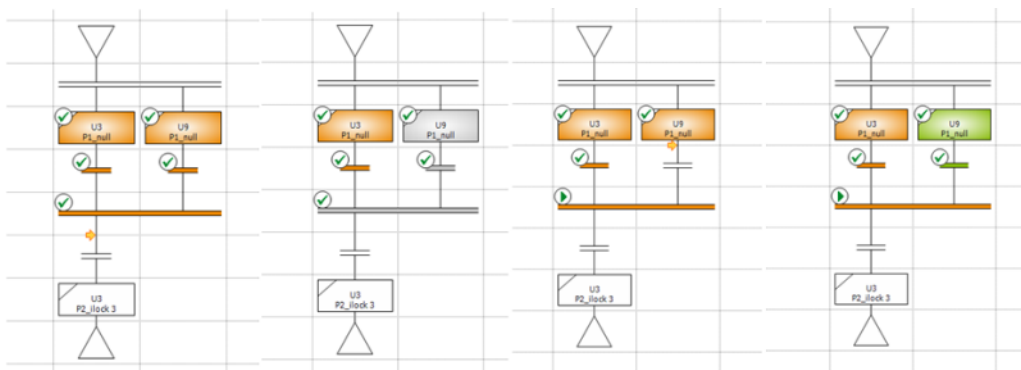
**Exceptions:** `Pause` and `Resume`. The execution positions remain as they are.

### PHASE COMMANDS

- **Hold for a phase:** The active element is deleted from the execution path of the phase.
- **Resume:** If there is an active element, a selective next step is executed in this execution path.

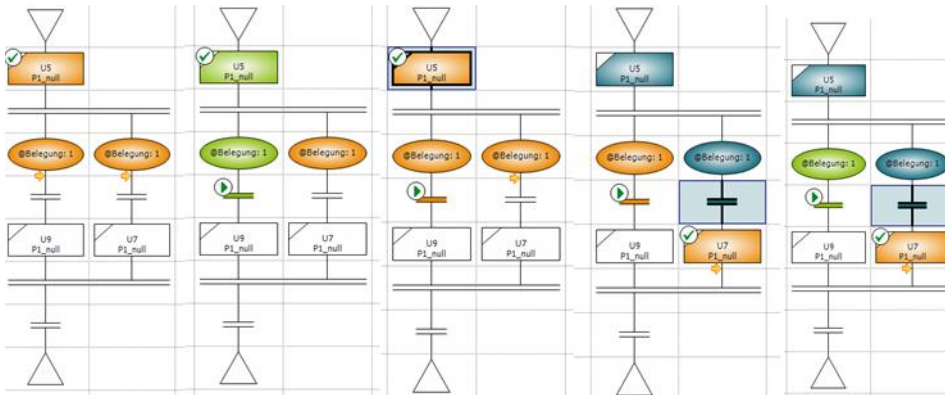
### SPECIAL CASES:

#### PHASE COMMAND **HOLD** AND **RESTART** IN A PARALLEL BRANCH FOR ACTIVE END PARALLEL BRANCH



If you hold in this example (images from left to right) a parallel branch and restart it, then you will reach after a step-by-step execution the already paused **end parallel branch**. To resume the execution from here, the left phase must receive command `Resume`.

## ONLY ONE PATH IN A PARALLEL BRANCH WITH AN ACTIVE PHASE BEFORE STEP-BY-STEP EXECUTION



If, in a parallel branch with a phase before, only one path is executed completely and waiting is taking place in one of the other parallel branches (phase before is *running*), no active element is displayed in the parallel branch. To get them in the other path, the phase must be paused with a phase command. After that it is possible to execute the path completely.

However, if the parallel branch with the active element continues to be executed, the phase before the parallel branch is deactivated. The left path thus remains *paused* and without an active element. As there is not active phase, the execution can only be resumed with a global *Pause* and *Resume*.

## JUMP

Jump means to move from one position to another, distant position during execution in order to continue the execution there.

To jump:

1. Select an active element with the mouse cursor
2. move it to one of the offered targets
3. execute the next step

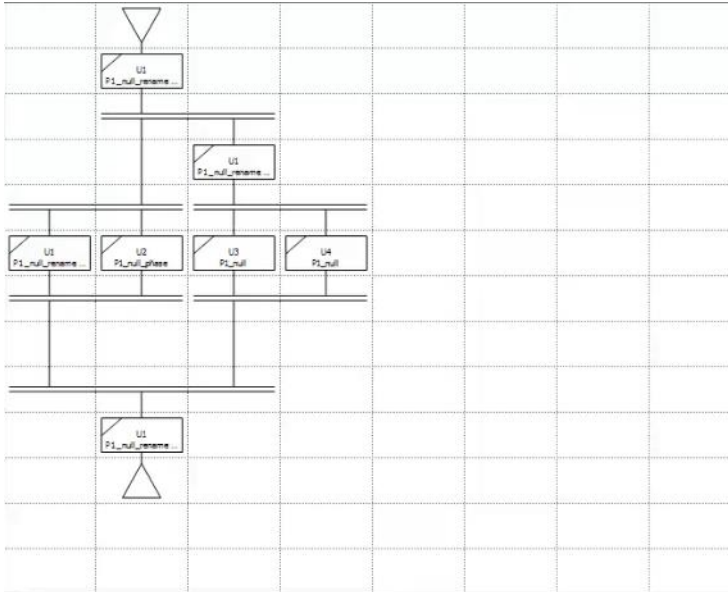
All active elements in front of the source pointer are deactivated and the object after the target is activated.

If a jump is registered for which source and target are analogously the same (jumps over lines, jump targets or end branch objects), this jump is ignored and a simple step is executed.

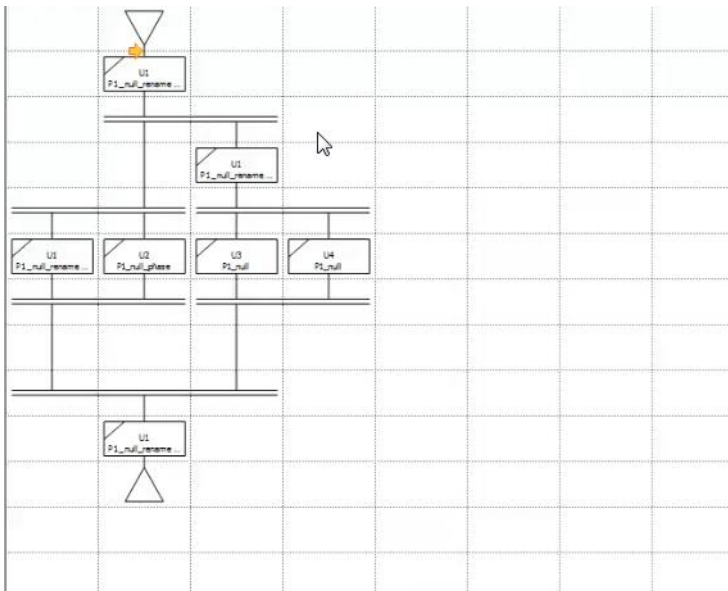
## JUMPING IN SIMULTANEOUS SEQUENCES

If, in parallel sequences, a jump is made in first parallel branch via the phase, then the first phase before the parallel branch is deactivated. Therefore no phase is active. Phases cannot be skipped if this means that no phase would be active in the recipe. The following behavior when skipping leads to an error:

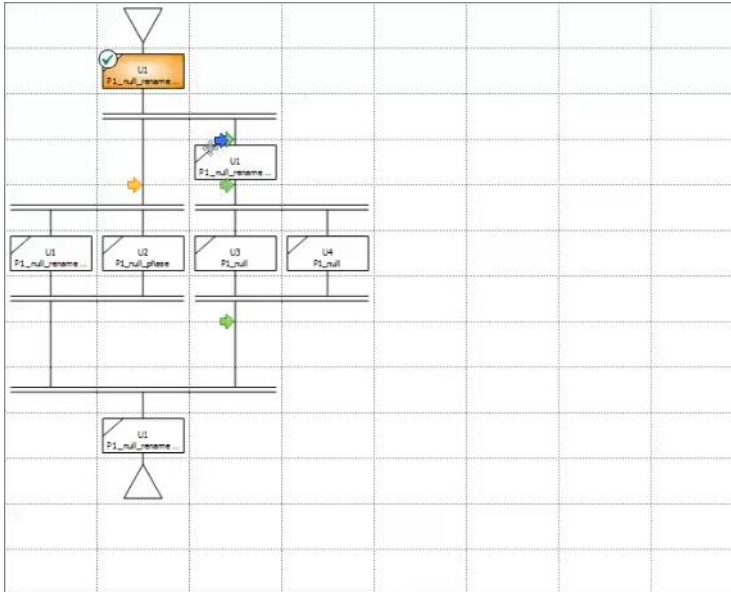
- Recipe with simultaneous sequence



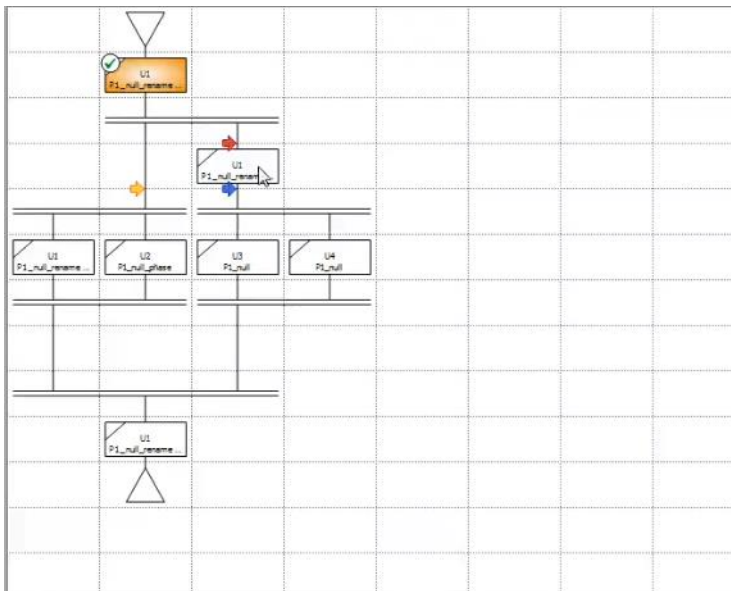
- A global step activates the phase before begin parallel sequence.



- After the next step, the active element is before the phase in the simultaneous sequence.



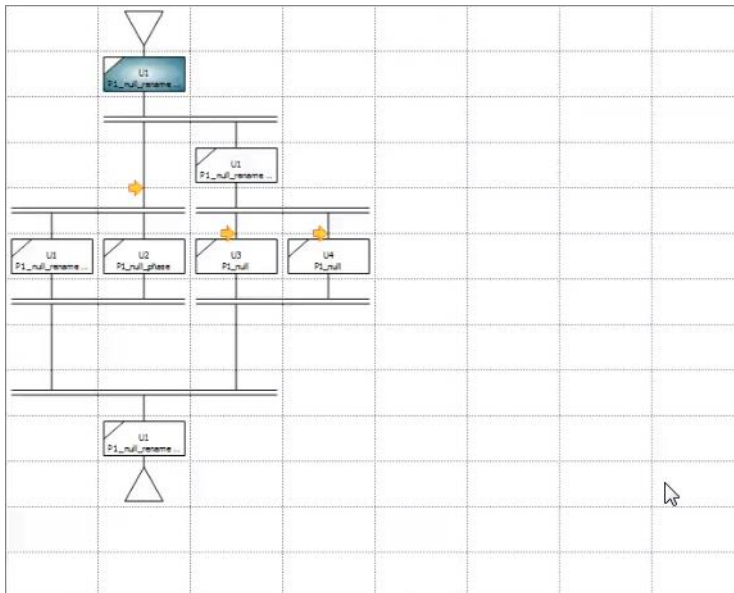
- ▶ The active element is moved behind this phase.



**Rule when jumping:** Objects before a begin parallel sequence become inactive as soon as objects become active after this.

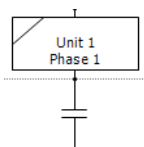
Thus the first phase becomes inactive when jumping. The skipped phase in the parallel branch never became active, nor did the following phases.

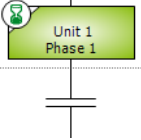
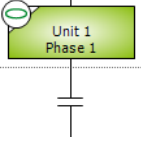
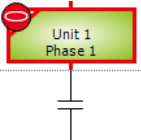
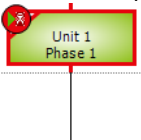
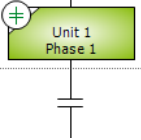
Thus no phase is still active in the recipe:

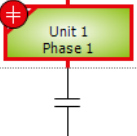
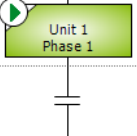
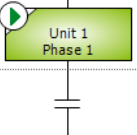
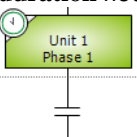
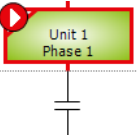
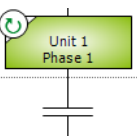


### 9.14.7 Process of a phase in detail

A phase is always processed sequentially after the same pattern. To break down the exact process, you also need a following condition. For this display we use a transition as following condition. We give the phase the name **Phase 1**. You can find additional special process behavior with following condition in chapter Following condition (on page 242).



Phase	Transition	Event
<p>Phase is activated</p> 		Phase activated
<p>All variables of the phase are registered at the drivers.</p>		
<p>Unit allocation is started and waiting period unit allocation is started.</p> 		If the unit allocation was not successful in the first try: Unit allocation not possible
<p>Optional: Unit allocation possible within the waiting period.</p> 		Waiting period unit allocation exceeded
<p>Check is started whether phase isn't already executed. This can happen if the phase is already active in a parallel branch or if the unit allocation was skipped manually and the phase is already executed in another recipe.</p> 		Phase started multiple times
<p>Start phase</p>		Phase started
<p>Check of the input interlocking is started and Waiting period input interlocking is started.</p> 		If the checking of the input interlocking was not successful in the first try: Input interlocking blocked
<p>Optional: Condition of the input interlocking not TRUE within the waiting period.</p>		Waiting period input interlocking exceeded

		
<p>Writing the command tag</p> 		<p>When all command tags were written: Finished writing command tags</p>
<p>Checking of the phase-done condition is started and time for <b>Minimum execution duration</b> and <b>Maximum execution duration</b> is started.</p> 		
<p>Optional: Phase-don condition fulfilled but <b>Minimum execution duration</b> not reached.</p> 		
<p>Optional: Condition of the <b>Phase-done condition</b> not TRUE within the <b>Maximum execution duration</b>.</p> 		<p>Maximum execution period exceeded</p>
<p>Optional: Waiting for restart of the whole execution. If the execution is still restarting (for other phases in the recipe the restart condition is not yet fulfilled), it is waited here. This guarantees that the following element is activated after the recipe changes to <b>running</b>.</p> 		



<p><b>Phase-done condition</b> is <code>TRUE</code> and minimum execution duration is reached or exceeded.</p>  <p>Waiting period following condition is started.</p>	Transition is activated	Phase finished
	All variables of the transition are registered at the drivers.	Waiting period following condition exceeded
<p>Optional: <b>Following condition</b> not within waiting period <code>TRUE</code></p> 	The transition condition is checked.	
The next phase is activated. The following condition can be composed from several objects (e.g. transition + unit allocation). No till the next phase is reached (or the end of the recipe), the following condition counts as fulfilled.		
<p>Phase is informed that the following condition is fulfilled</p> 	Transition condition is <code>TRUE</code> .	Phase deactivated
All events of the phase are deactivated.		
All variables of the phase are signed off from the drivers.	All variables of the transitions are signed off from the drivers.	
The phase is deactivated.	The transition is deactivated.	

## RULE FOR VALUES OF TRANSITIONS

If a transition has the value `TRUE` for the phase-done condition during the waiting period, it is marked as finished. If its value should later change to `FALSE`, the execution of the recipe is not influenced.

## GLOBAL PAUSING AND CONTINUATION IF THERE IS A COMMUNICATION ERROR

If a phase is paused and there is a communication error, this cannot be simply continued. Phase commands are prevented, recipe commands are circumvented. If the recipe is paused, a `Continue recipe` recipe command can be sent. As a result of this, the recipe changes to the status `running`, but the execution path of the phase concerned remains unaffected.

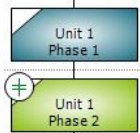
### Following condition

A phase is active as long as the following condition is fulfilled. Only once the following condition is fulfilled is the phase deactivated. The `phase completed` event is triggered and the phase is deactivated. Before it is deactivated, the event reactions are executed. See also Process of a phase in detail (on page 238)

The following condition can be very different. Here some examples:

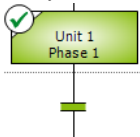
- ▶ Phase 1 followed by a phase 2:

As soon as the phase done condition (and optional the **Minimum execution duration**) is fulfilled for phase 1, phase 1 is completed and phase 2 is activated.



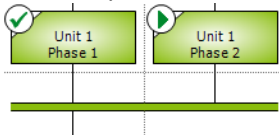
- ▶ Phase 1 followed by a transition:

Only when the transition condition is fulfilled, phase 1 is completed.



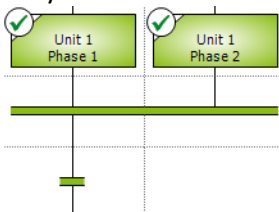
- ▶ Phase 1 and phase 2 parallel followed by an end simultaneous sequence:

Only once the phase done condition (and optionally the **minimum execution duration**) is fulfilled for both phases are both phases completed.



- ▶ Phase 1 and phase 2 parallel followed by an end parallel branch followed by a transition:

Only when the transition condition is fulfilled, both phases are completed.



### 9.14.8 Escape phase

It is possible to exit a phase during execution. Execution is then continued after the phase without having to run through the outstanding steps of the phase. It is only possible to exit a phase in manual mode. In the Editor, it is also possible to configure a condition in the **Condition transient status/Escape condition** property.

It is also possible to exit if the phase is in this status:

- ▶ Running
- ▶ Pausing
- ▶ Paused
- ▶ Holding
- ▶ Held
- ▶ Restarting

If the exit command has been reached, the current execution step is canceled and checking of the **Escape condition** starts immediately. If the condition exiting is met, all necessary steps are instigated so that the object changes back to the status of `running`:

- ▶ Restarting: The normal procedure is continued from the Wait until the recipe has the status "running". That means: All steps between instigating the exit and Wait until the recipe has the status "running" are not executed.
- ▶ Running: The normal procedure continues to be executed after the phase or the phase waits for the `pause` condition to be met.

No more transient conditions are checked from the start of the check for the exit of a phase. Only after a switch to `stopping` or `aborting` is the checking of **Escape condition** interrupted.

Behavior when exiting from the phase depending on the execution status:

- ▶ Exit from the phase from `holding` or from `held`:  
A restart is only carried out internally when switching back to `running`. The execution data is retained with the object. This also applies if a restart is carried out during the checking **Escape condition**.
- ▶ Exit from the phase if the recipe status is not `running` or `restarting`:  
If there is a different recipe status, once the condition has been met, waiting continues until the recipe status changes to `running` or `restarting`. This also makes it possible to exit with a different recipe status.  
For example: The recipe has the status `held`. The `exit` action is carried out for a phase. Then the end of the exit is only reached if the user executes a global `restart` command. This also makes it possible to exit if the recipe has the status `held`, without using the normal mechanism for a restart.



### Information

*If an error occurs with communication to the PLC, the check of the exit conditions is postponed until the error has been rectified.*

## SUPPRESSION OF REACTIONS WHEN EXITING

Most events for the status change and mode change are suppressed whilst a phase is exited. The checking of the **Escape condition** has priority and the status change is also carried out without the condition being checked. Exceptions to this are `stopping` and `aborting`, as well as `End Runtime triggered` and `Runtime restart`, because these have a higher priority than `Exit without phase`. These events are triggered, as well as the event for `Linked variable invalid`.

### 9.14.9 Restart phase

Phases can be restarted. In doing so, all connected active objects are deactivated, including connected simultaneous sequences. A phase always carries out a series of event for activation `activation`, `start` and `finished`, regardless of the number of restarts. Events that have already been carried out are skipped on restarting. Transient conditions are only checked after the `start` event.

## DEACTIVATION

At the restart of a phase, all active objects are deactivated in principle, however only if they are linked. Isolated active objects are not deactivated.

## SEQUENCE OF DEACTIVATION

The deactivation starts from the object which has been active the longest. After the restart this object is also going to be activated. If this object is deactivated, it also deactivates all branches to which it can establish a connection via an active object.

## ISOLATED BRANCH

A branch is isolated if it is not connected to another branch via an active object. The deactivation only takes place between connected branches. As long as an object does not have status `Completed`, there is not active connection to the following object.

## RESTART OF SELECTED PHASES

One or more phases can be selected and restarted. For the selection of several phases, they must be in separate branches.

The oldest object is restarted in the selected branch. With this all active objects in the connected branches are deactivated.

## GLOBAL RESTART

The global restart carries out a restart for all phases. The restart is done for the oldest active object and with this all connected, active objects are deactivated. All remaining active objects are in an isolated branch. Here also the oldest object is restarted until all active objects were dealt with.

### 9.14.10 Secure writing of the command parameters

The command parameters (initial parameters and value parameters) are written to the PLC securely. The waiting time can be configured in the Editor.

#### PROCEDURE

The following applies when writing command parameters:

1. Waiting occurs until all parameters to be written as inverted have a value
  - If this is not possible within the configured waiting period, 3 attempts are made.
  - If there are still parameters with no value, no parameter is written, not even those with a value.
2. Command parameters are written.
3. The actual values are compared to the written values.
  - After a positive write confirmation, waiting occurs until the variables to which they were written can be read again. All written values must be active at the same time. If the waiting period has expired, writing starts again. There is a maximum of 3 repetitions.
4. If all attempts have been unsuccessful, the `Command parameter without value event` is triggered.
5. If secure writing is unsuccessful, a `communication failure event` is triggered. That means:
  - The error must be acknowledged.
  - The phase must be put back to the status "running".
  - Writing is restarted once continued.
  - The execution of the function is restarted in the event of a restart.

6. The procedure can be paused, held etc. using commands whilst secure writing is being carried out. If the phase has the status `paused` and it is then continued, the writing is also restarted. In doing so, the values to be inverted are recalculated for a command parameter to be toggled.
7. Writing of the command parameters can also be skipped.
8. If Runtime is ended whilst writing the command parameters, these are rewritten when Runtime is started. Parameters to be toggled are recalculated.

Duration, start time and end time of the writing are displayed in the tooltip of the phase.

## CONFIGURING THE WAITING PERIOD

To configure the waiting period for secure writing:

1. Navigate to the **General/Protected writing** group in the properties of the Batch Control module.
2. Enter the desired waiting period in the **Time out for protected writing** property



### Information

*The waiting period includes all waiting processes in the whole write process:*

- ▶ Waiting for values for all parameters to be written
- ▶ Waiting for confirmation of the written parameters
- ▶ Waiting for reading of the variables

If the standard value of 20 seconds is used, all wait processes within 20 seconds must be concluded positively. If the waiting period has expired without a positive result, writing is started over.

## 9.14.11 Exit and restart Runtime

### ACTIONS ON RESTARTING

Actions can be predefined for restarting Runtime after closing. These can be defined for:

- ▶ **Restart after normal shutdown**
- ▶ **Restart on system failure**

One of the following actions can be selected for each of the two properties:

- ▶ **Hold recipe:** The complete recipe is held after restarting.
- ▶ **Recipe pausing:** The complete recipe is paused after restarting.

- **Retain recipe status:** After restarting, the recipe is set to the same status as before closing.

## STATUS CHANGE

After restarting, an attempt is made to execute the configured status change. To do this, the corresponding command must also be executable. The status `Restarting` for recipes and phases is handled in the same way as `in execution`. That means:

- **Paused is set for:** `In execution`, `pausing` and `restarting`.
- **Held is set for:** `In execution`, `pausing`, `paused`, `holding` and `restarting`.

Transient conditions are not checked and events are not set. Therefore the status in the recipe can be brought in line with the status of the equipment, without sending events to the equipment for the status change.

## INFORMATION IN RECIPES AND UNIT

When restarting after Runtime has been restarted, the respective status is stored with the information in the recipes. For example: **Paused after normal shutdown** or **Held after incorrect shutdown**.

The execution status is also displayed in the unit information. The execution status (numerical and text) in the unit information contains a number and text that corresponds to that of the variables in the screen. Including information on whether triggered by a restart, information on objects with a different status and objects that delay a status change.

**Caution:** The content of these variables is not compatible between zenon 7.10 and 7.11.

## IMAGE FILE

At closing the Runtime an image file (on page 228) of the running recipe is created. It contains the images of the REEs, the order of the allocation and all reactions which are needed for the restart.

## ALLOCATIONS

After the restart the allocations match the state before the finishing. It is saved who allocated a unit and who forced an allocation in which order. If a recipe cannot be restarted (e.g. because of failed validation), the allocations for this recipe are removed.

## REACTIONS

Reactions which were triggered by the process are also incorporated in the image if they are active. They are then executed after the REE is restarted. This guarantees that the reaction is always executed as a whole either before the image file is created or after the restart.

The `Exit Runtime` reaction is always executed and can never be incorporated in the image.

## SYNCHRONOUS WRITING

The REE manages the confirmation for all variables whose write set value should be executed synchronously. The time out for this is defined by the time within which the Runtime must be closed. For each write acknowledgment the time out is restarted. A time out is written in the log file.

Variables which don't access a driver are always written without an acknowledgment even if an acknowledgment is requested. Internal drivers do not support acknowledgments.

## ALLOCATE TAG

As during the start of the Runtime all drivers are also started, it is possible that they do not provide valid values if they are needed at the restart. During the restart it is not waited for the value update. This does not ensure that the value is written as expected. If no value is available, the alternate value is used.

There is a wait for the values of internal drivers if they are available within 2 cycles.

## CHECKING FOR A COMMUNICATION ERROR

A check is also made for communication errors when restarted if this has been configured (on page 291).

# 10. Behavior in the network

The module **Batch Control** is fully capable of using a network in terms of Client/Server technology. This means that Batch recipes can be created, duplicated, edited, deleted, etc. on a Client. The whole recipe management remains always on the server. Likewise the whole process control such as **start** recipe, **pause** recipe, **stop** recipe, etc. can be done from the Client. Also mode changes and manual operations such as **jump** are possible.



### Attention

*Module Batch Control does not support redundancy. There is no synchronization between Standby Server. When the Server breaks down, the executed Batch recipes are not continued seamlessly on the Standby!*

For using Batch Control in a network the following is true:

## ALLOCATION

- The forcing of allocations can be carried out from the Server or Client.



## FUNCTIONS

Functions are always carried out at the Server.

## PHASES

- ▶ Editing phases in the master recipe:
  - Edit mode: Changes a done locally at the Client.  
If during the editing the recipe is saved on another computer in the network, the current configuration is lost. An appropriate message is displayed and the editing dialog is closed. The new data from the server are displayed.
  - Test mode: Changes a done at the Server.
- ▶ Control recipe: Changes a done at the Server.
- ▶ If a recipe is saved in the network, all Clients using this recipe are updated.
- ▶ If a recipe is opened on a client, the current version on the server is always displayed, even if it has not yet been saved there.
- ▶ If a recipe is deleted on a computer, a message is displayed on all computer on which the recipe is opened that the recipe has been deleted.

## MODE

- ▶ The mode (automatic, semi-automatic, manual) can be switched by the server and the client.
- ▶ Jumps in the recipe and step-by-step progress of a recipe can be done from Server and Client.

## RECIPES

- ▶ Recipes can be started and controlled by the zenon server and by zenon clients.
- ▶ If parameters in a recipe are changed whilst the recipe is saved on a different zenon client, the change to the parameters is refused and not carried out.
- ▶ A master recipe can be changed on the zenon client whilst it switches to test mode on the zenon server and is sent to the zenon client. The changes that were last saved are transferred. This means: If the zenon client saves last, the recipe is switched to editing mode again. If the zenon server saves last, the change to the zenon clients is discarded and the recipe is in test mode.
- ▶ If a communication error occurs when deleting a recipe or an operation template, the deletion is refused with an error message.

## WEB CLIENT

With a standard web client:

- ▶ The settings for grid and color can be changed
- ▶ No recipes can be created or edited
- ▶ The size of the editing area cannot be changed
- ▶ In the toolbar, all symbols that are not permitted are deactivated; it is not possible to select the corresponding objects.

Web client PRO is not affected by these restrictions.

## 10.1 Redundancy

zenon Batch Control does not support redundancy. In networks that have been set up as redundant, this means:

- ▶ If the server fails, the recipe is in an undefined status.  
It is not passed on to the standby server.
- ▶ On the standby server, once it has stepped up to become the server:
  - Master recipes can be switched back to edit mode and deleted.  
These changes are carried over to the server once it is working again.
  - Control recipes cannot be edited or executed.
- ▶ Starting, pausing or other commands are not possible on the Standby Server.
- ▶ If the recipe is not edited in the Standby Server, it runs normally again as soon as the server is online again.

## 11. Reporting

Reports for configuration of the recipes can be created with the Report Viewer integrated into zenon.

When switching to a Report Viewer screen:

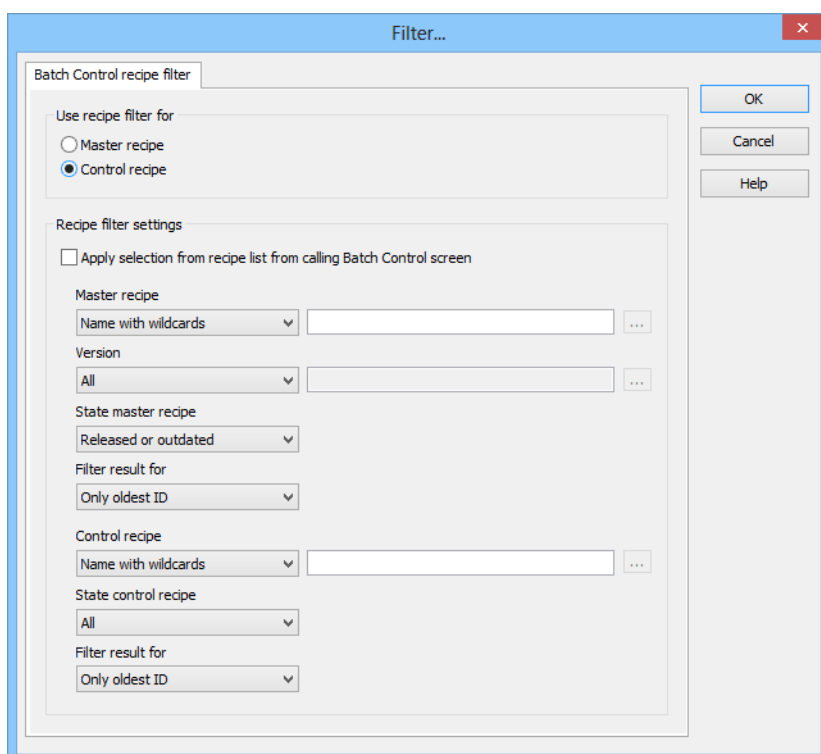
- ▶ it is possible to filter for recipes (on page 251)
- ▶ Datasets for Batch Control reports can be created:
  - Master recipe
  - Control recipe
  - Recipe screens
  - Matrix cells
  - PFC structure

- Basic functions
- Parameters
- Transitions
- Unit allocations
- Operation instance

## 11.1 Batch Control recipe filter

When screen switching to a report viewer type screen, a filter can be set for recipes from the Batch Control module. To filter according to recipes:

1. Open the Report definition tab for screen switching
2. go to area **Filter**
3. In the **Module-specific filter** tab, click on **Batch Control: Recipe Filter**
4. the dialog for configuring the filter is opened



The screenshot shows a dialog box titled "Filter..." with a close button (X) in the top right corner. Inside the dialog, there is a tab labeled "Batch Control recipe filter". Below the tab, there are two radio buttons under the heading "Use recipe filter for": "Master recipe" (unselected) and "Control recipe" (selected). To the right of these buttons are three buttons: "OK", "Cancel", and "Help". Below the radio buttons is a section titled "Recipe filter settings". This section contains a checkbox labeled "Apply selection from recipe list from calling Batch Control screen" which is currently unchecked. Under this checkbox, there are two groups of settings. The first group is for "Master recipe" and includes a dropdown menu set to "Name with wildcards", an empty text input field, and a button with three dots. Below this is another dropdown menu set to "All", another empty text input field, and a button with three dots. The second group is for "State master recipe" with a dropdown menu set to "Released or outdated". Below this is a dropdown menu set to "Only oldest ID". The second group is for "Control recipe" and includes a dropdown menu set to "Name with wildcards", an empty text input field, and a button with three dots. Below this is a dropdown menu set to "All", another empty text input field, and a button with three dots. Finally, there is a dropdown menu set to "Only oldest ID".

## USE RECIPE FILTER FOR

Parameters	Description
<b>Use recipe filter for</b>	Selection of the recipe type that is applied to the filter: <ul style="list-style-type: none"> <li>▶ Master recipe</li> <li>▶ Control recipe</li> </ul>
<b>Master recipe</b>	Active: It is filtered on Master recipes.
<b>Control recipe</b>	Active: It is filtered on control recipes.  <b>Note:</b> The attendant master recipes must also be selected. If no master recipe has been selected for the control recipe, the filter cannot find the recipe being searched for in Runtime.  <b>Hint:</b> If the master recipe is not known, filtering of all master recipes with a placeholder is recommended.

## RECIPE FILTER SETTINGS

Parameters	Description
<b>Recipe filter settings</b>	Options for the recipe filter
<b>Apply selection from recipe list from calling Batch Control screen</b>	Active: In Runtime, the first selected recipe of the batch screen from which the report viewer screen is called up is used. Individual settings in this dialog are then not available.  Inactive: The filter settings are changed individually using this dialog.
<b>Master recipe</b>	Parameters for the selection of the master recipe. Selection from drop-down list: <ul style="list-style-type: none"> <li>▶ Name with wildcards: A name with placeholder can be entered into the input field. Filtering according to this name is carried out.</li> <li>▶ Name from variable:_ The name of the master recipe is defined by a variable in Runtime. A click on button ... opens the dialog for selecting variables.</li> <li>▶ ID from variable:_ The ID of the master recipe is defined by a variable in Runtime. A click on button ... opens the dialog for selecting variables.</li> </ul>
<b>Version</b>	Selection of the version (on page 192) from the drop-down list: <ul style="list-style-type: none"> <li>▶ All: The version stated is ignored and each version found is used.</li> </ul>

	<ul style="list-style-type: none"><li>▶ Fixed version: This filters for versions that are entered in this field. Highest possible version: 4294967295</li><li>▶ Version from variable: The recipe that was in the linked variables at the time of execution is filtered for. Click on button ... in order to open the dialog for selecting a variable.</li><li>▶ Only oldest version: Only the recipe with the oldest version number is used.</li></ul> <p>Only newest version: Only the recipe with the newest version number is used.</p>
--	---

<b>State master recipe</b>	<p>Status of the recipe Select from drop-down list:</p> <ul style="list-style-type: none"> <li>‣ Released or outdated</li> <li>‣ Released</li> <li>‣ Outdated</li> </ul>
<b>Filter result for</b>	<p>Define which ID is to be selected when filtering for names by selecting from the drop-down list:</p> <ul style="list-style-type: none"> <li>‣ Only oldest ID</li> <li>‣ Only newest ID</li> </ul> <p>Because a report can only be used for one recipe, it is not possible to filter for "all recipes".</p>
<b>Control recipe</b>	<p>Parameters for the selection of the control recipe. Selection from drop-down list:</p> <ul style="list-style-type: none"> <li>‣ Name with wildcards: A name with placeholder can be entered into the input field. Filtering according to this name is carried out.</li> <li>‣ Name from variable:_ The name of the control recipe is defined by a variable in Runtime. A click on button ... opens the dialog for selecting variables.</li> <li>‣ ID from variable: The ID of the master recipe is defined by a variable in Runtime. A click on button ... opens the dialog for selecting variables. Precisely one recipe can be found if the variable value at the time of execution is a valid ID of a control recipe.</li> <li>‣ Job ID from variable: Finds control recipes that belong to the master recipes already found and which have the given job ID. Any type of variable can be linked. The value is automatically converted into STRING. <b>Note:</b> If the variable does not have a value, no recipe is sent to the Report Viewer.</li> </ul>
<b>State control recipe</b>	<p>Selection of the recipe status from the drop-down list:</p> <ul style="list-style-type: none"> <li>‣ All</li> <li>‣ Prepared</li> <li>‣ Running</li> <li>‣ Executed</li> <li>‣ Terminated with error</li> <li>‣ Outdated</li> </ul>

<b>Filter result for</b>	Define which ID is to be selected when filtering for names or job ID by selecting from the drop-down list: <ul style="list-style-type: none"> <li>▶ Only oldest ID</li> <li>▶ Only newest ID</li> </ul>
<b>OK</b>	Applies all changes, creates filter and closes the dialog.
<b>Cancel</b>	Discards all changes in all tabs and closes the dialog.
<b>Help</b>	Opens online help.

**Note for variable selection using name or ID:** For the selection of variables according to name or ID, numerical variables and string variables can be selected respectively. The data types are converted to the respective correct form.

## 12. Formula editor

The formula editor is automatically opened if you need to enter or edit a formula. Above all:

### Editor:

- ▶ Phases
- ▶ Interlocking conditions
- ▶ all conditions for transitions
- ▶ Phase done condition

### Runtime:

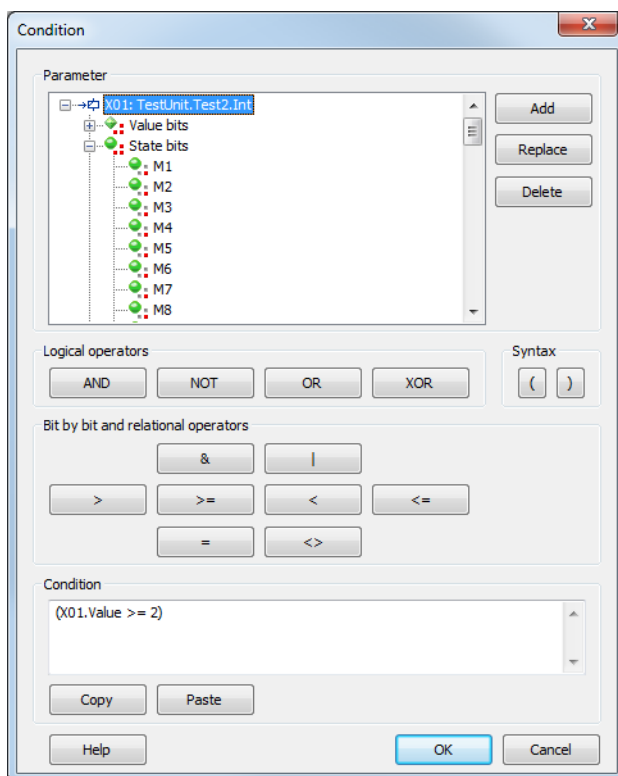
- ▶ Phase done condition and interlocking when editing a phase
- ▶ Editing transitions

**Note:** If the phase referenced in the formula is removed and a new phase is added, the operands are reassigned in the case of operands. To do this, the same phase must be reinserted. Parameters from a different phase are not automatically linked.

## ENTER FORMULA

The following input is accepted:

- ▶ Constant as decimal number
- ▶ Hexadecimal number if it is preceded by an **x**
- ▶ Dot as decimal separator; the following is true:
  - Comma is automatically converted into a dot: 23,000 to 23.000
  - Decimal places which are only zeros are removed: 23.000 to 23





Parameter	Description
<b>TAG list</b>	<p>List of the tags which can be used for the formula.</p> <p>Each entry contains of:</p> <ul style="list-style-type: none"> <li>▶ a basis node for the label</li> <li>▶ a value</li> <li>▶ a status</li> <li>▶ the bits for value and status</li> </ul> <p>A symbol at the first node displays whether it is a command or return tag.</p> <p>The short identifier at the beginning of the name is used for the formula.</p>
<b>Add</b>	<p>Opens the dialog for adding a parameter (on page 260). For this, the following applies:</p> <ul style="list-style-type: none"> <li>▶ The following can be added: numeric and binary tags and tags for time duration. Values for duration are converted to seconds</li> <li>▶ For conditions of the phase only the tags created for it can be added.</li> <li>▶ Tags can be added multiple times.</li> </ul>
<b>Replace</b>	<p>Makes it possible to replace a tag. Clicking on the button opens the dialog to add a parameter (on page 260).</p> <ul style="list-style-type: none"> <li>▶ Selection of a new parameter replaces the highlighted parameter.</li> <li>▶ Clicking on the <b>no selection</b> button deletes the highlighted parameter from the list.</li> </ul> <p>The short identifier remains the same at replacing.</p>
<b>Remove</b>	<p>Removes the highlighted tag. For a tag to be deleted:</p> <ul style="list-style-type: none"> <li>▶ the formula must be correct</li> <li>▶ the selected tag must not be used in the formula</li> </ul>
<b>Logical operators</b>	Via the buttons for operators, operators are added to the formula.
AND	logical 'AND'
OR	logical 'OR'
XOR	logical 'EXCLUSIVE OR'
NOT	Negation
<b>Syntax</b>	The operator buttons add the string shown on them to the formula.
(	Open parenthesis

)	Close parenthesis
<b>Bit by bit and relational operators</b>	
&	And
	Or
>	greater than
>=	greater or equal
<	less than
<=	Less than or equal
=	equal
<>	less or greater
<b>Condition</b>	Configuration and display of the formula.
<b>Copy</b>	<p>Copies the whole formula:</p> <ul style="list-style-type: none"> <li>▶ All configured tags from the tag tree</li> <li>▶ Formula from the field</li> </ul>
<b>Paste</b>	<p>Pastes a formula from the clipboard. At this all already configured elements are deleted and replaced by the copied formula.</p> <p>When copying formulas between phases, it is tried to resolve the operands via their names. For tags which are not found invalid entries are created in the operands list. Their point of use in the formula remain the same.</p>
<b>OK</b>	<p>Applies formula and closes the dialog.</p> <p>For this the formula must be correct.</p>
<b>Cancel</b>	Discards all changes and closes the dialog.



### Information

*You can link up to 99 tags in a formula. X01 to X99. The length of the formula must not exceed 4096 characters.*

### THE MEANING OF THE BITS:

Parameters	Description
<b>value bits</b>	32 value bits (from 0 -31) are available. They describe the tag value bit by bit. For binary tags only bit 0 is of importance, for SINT and USINT only the bits from 0 – 7, etc.
<b>State bits</b>	Here you find the most commonly used status bits. You find the exact definition and use of the status bits in the Status Bits List.
<b>value and status</b>	<p>In the formulas, all values (value bits and status bits) are treated as binary values and can be logically linked with AND, OR, etc.</p> <p>The total value and overall status are an exception to this. In order to get a Boolean result this total value has to be ORed with a constant bitwise. For this, we use the operator &amp;.</p> <p>For the result 0 (false) of this logical ORing we get the binary value 0 (false), otherwise 1 (true).</p> <p>Example: see chapter Example bit by bit ORing</p>



### Info

*The status bits NORM and N\_NORM are only available in the formula editor and cannot be engineered via the status.*



### Information

Formulas with binary X values and bitwise linking can be used with a maximum of 2 binary values. If more values are required, the linking must be carried out without binary X values.

#### Example:

**X01.Value & X02.Value** -> works

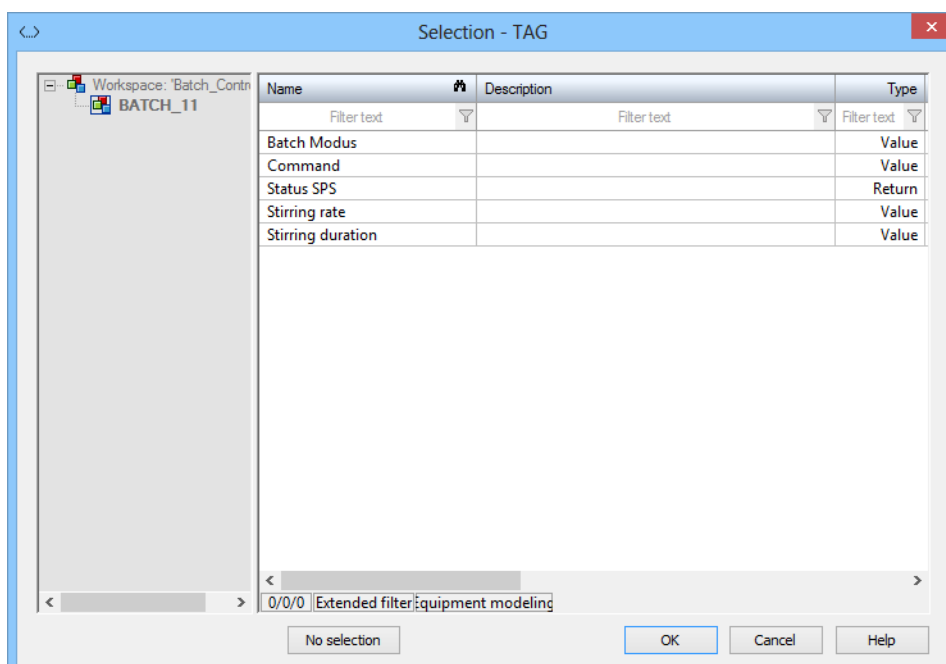
**X01.Value & X02.Value & X03.Value** -> does not work

But:

**X01.00 AND X02.00 AND X03.00 AND X04.00 AND X05.00** -> works

## 12.1 Adding parameters

Clicking on the **Add** button in the formula editor (on page 255) opens the dialog to select parameters that are to be used for a formula.



Parameter	Description
<b>Project list</b>	Display of the active project. Only parameters that have been created in the active project for the phase to be configured can be selected.
<b>TAG list</b>	<p>List of the parameters available for the selected phase.</p> <p>Multiple selection is possible. Apply by selecting and clicking on the OK button or by double clicking on a parameter.</p>
<b>None</b>	<p>Deletes parameters already set. Only effective for replacement of parameters.</p> <p>If a parameter is highlighted in the formula editor and this dialog is opened by clicking on the <b>Replace</b> button, then clicking on the <b>No selection</b> button deletes the parameter from the list in the formula editor.</p> <p>The short identifier remains the same at replacing.</p>
<b>OK</b>	Inserts selected parameters into the parameter list of the formula and closes the dialog.
<b>Cancel</b>	Discards selection and closes dialog.
<b>Help</b>	Opens online help.

## 12.2 List of status bits

Bit number	Short term	Long name	zenon Logic label
0	M1	User status 1; for Command Processing: Action type "Block"; Service Tracking (Main.chm::/IEC850.chm::/117281.htm) of the IEC 850 driver	_VSB_ST_M1
1	M2	User status 2	_VSB_ST_M2
2	M3	User status 3	_VSB_ST_M3
3	M4	User status 4	_VSB_ST_M4
4	M5	User status 5	_VSB_ST_M5
5	M6	User status 6	_VSB_ST_M6
6	M7	User status 7	_VSB_ST_M7
7	M8	User status 8	_VSB_ST_M8
8	NET_SEL	Select in the network	_VSB_SELEC
9	REVISION	Revision	_VSB_REV
10	PROGRESS	In operation	_VSB_DIRECT
11	TIMEOUT	Runtime exceedance	_VSB_RTE
12	MAN_VAL	Manual value	_VSB_MVALUE
13	M14	User status 14	_VSB_ST_14
14	M15	User status 15	_VSB_ST_15
15	M16	User status 16	_VSB_ST_16
16	GI	General query	_VSB_GR
17	SPONT	Spontaneous	_VSB_SPONT
18	INVALID	Invalid	_VSB_I_BIT
19	T_CHG_A	Daylight saving time/winter time announcement	_VSB_SUWI
20	OFF	Switched off	_VSB_N_UPD
21	T_EXTERN	Real time external	_VSB_RT_E
22	T_INTERN	Realtime internal	_VSB_RT_I
23	N_SORTAB	Not sortable	_VSB_NSORT
24	FM_TR	Error message transformer value	_VSB_DM_TR
25	RM_TR	Working message transformer value	_VSB_RM_TR

26	INFO	Information for the variable	_VSB_INFO
27	ALT_VAL	Alternate value  If no value was transferred, the defined alternate value is used otherwise the last valid value is used.	_VSB_AVALUE
28	RES28	Reserved for internal use (alarm flashing)	_VSB_RES28
29	N_UPDATE	Not updated	_VSB_ACTUAL
30	T_STD	Standard time	_VSB_WINTER
31	RES31	Reserved for internal use (alarm flashing)	_VSB_RES31
32	COT0	Cause of transmission bit 1	_VSB_TCB0
33	COT1	Cause of transmission bit 2	_VSB_TCB1
34	COT2	Cause of transmission bit 3	_VSB_TCB2
35	COT3	Cause of transmission bit 4	_VSB_TCB3
36	COT4	Cause of transmission bit 5	_VSB_TCB4
37	COT5	Cause of transmission bit 6	_VSB_TCB5
38	N_CONF	Negative confirmation of command by device (IEC 60870 [P/N])	_VSB_PN_BIT
39	TEST	Test bit (IEC870 [T])	_VSB_T_BIT
40	WR_ACK	Writing acknowledged	_VSB_WR_ACK
41	WR_SUC	Writing successful	_VSB_WR_SUC
42	NORM	Normal status	_VSB_NORM
43	N_NORM	Deviation normal status	_VSB_ABNORM
44	BL_870	IEC 60870 Status: blocked	_VSB_BL_BIT
45	SB_870	IEC 60870 Status: substituted	_VSB_SP_BIT
46	NT_870	IEC 60870 Status: not topical	_VSB_NT_BIT
47	OV_870	IEC 60870 Status: overflow	_VSB_OV_BIT
48	SE_870	IEC 60870 Status: select	_VSB_SE_BIT
49	T_INVAL	Time invalid	not defined
50	CB_TRIP	Breaker tripping detected	not defined
51	CB_TR_I	Breaker tripping detection inactive	not defined
52	OR_DRV	Value out of Range	not defined
53	RES53	reserved	not defined

54	RES54	reserved	not defined
55	RES55	reserved	not defined
56	RES56	reserved	not defined
57	RES57	reserved	not defined
58	RES58	reserved	not defined
59	RES59	reserved	not defined
60	RES60	reserved	not defined
61	RES61	reserved	not defined
62	RES62	reserved	not defined
63	RES63	reserved	not defined



### Information

*In formulas all status bits are available. For other use the availability can be reduced.*

You can read details on status processing in the Status processing chapter.

## 12.3 Logical operators

Logical links: Variables will only be checked for the logical value '0'; if the value does not equal '0', it will be considered as '1'.

In contrast to bit formulas, the technical range can be modified by a stretch factor -> (not equal '0' or '1').

Operator	Meaning
AND	logical 'AND'
NOT	Negation
OR	logical 'OR'
XOR	logical 'EXCLUSIVE OR'

The operators have the following priority in the formula calculation:



Priority	Operator
1	& (operator for bit formulas)
2	NOT
3	AND
4	XOR/OR

**Info**

*Up to 99 variables can be linked in one formula. X01 to X99.*

**Info**

*The status bits NORM and N\_NORM are only available in the formula editor and cannot be engineered via the status.*

## 12.4 Bit formulas

Bit formulas only have a logical high or low state. In contrast to logical formulas, the raw value is already predefined (0,1).

Operator	Description
&	AND
	OR

### 12.4.1 Example: ORing bitwise

You want to find out if one of the user status bits 1-8 (M1 ... M8) of the variable X01 is set.

#### USUAL FORMULA:

**X01.M1 OR X01.M2 OR X01.M3 OR X01.M4 OR X01.M5 OR X01.M6 OR X01.M7 OR X01.M8**

This query can be made much easier by the logical ORing of the overall status.

## LOGICAL ORING:

### **X01.Status & 0xFF**

The constant can be entered in hexadecimals, as described above:

0xFF corresponds to decimal 256; these are the first eight status bits (binary 11111111). If one of these bit is set to 1, the result of this bitwise ORing is 1 (true), otherwise it is 0 (false).

If, for example, all user status bits except the user status bit M7 should be queried, the binary statement for this would be: 10111111. Bit 7 is not of interest and is thus set to 0. This corresponds to 0xBF in hexadecimal. The expression for the formula is then: **X01.Status & 0xBF**.

Instead of ORing bitwise with a constant, the value can also be directly compared to a decimal number. If the comparison is wrong, the binary value is 0 (false) otherwise it is 1 (true).

Example:

You want to find out if the value is equal to the constant 202: The formula is:

### **X01.value = 202**

If the value is equal to the constant 202, the result of the comparison is 1 (true) otherwise it is 0 (false).

Note: The bitwise ORing works with the OR character (|) in a similar manner to this example.

## 12.5 Comparison operators

Comparison operators are for the direct comparison of two numeric values. The result of this comparison is a binary value. "0" if the condition is not fulfilled and „1“ if the condition is fulfilled.

Operator	Description
<	Less
>	greater
<=	Less than or equal
>=	greater or equal
=	Equal
<>	unequal

To the left and to the right of the comparison operator, there has to be a (total) value or a (total) status, single bits cannot be used with these comparison operators.

There can also be a constant to the right of the comparison operator.

These constants are entered as hexadecimal values or decimal values in the combined element.

Hexadecimal figures are automatically converted to decimal values by clicking on **OK** (for example, 0x64 is in decimal figures 100).



### Example

*X01.value >= X02.value*

*The result is 1, if the value of X01 is higher than or equal to the value of X02*

*X01.value = 0x64*

*The result is 1, if the value of X01 is exactly equal to the numeric value 100 (= hex 0x64)*

*(X01.value = 0x64) OR (X01.value = 0x65)*

*The result is 1, if the value of X01 is exactly equal to the numeric value 100 or 101 (= hex 0x64 and hex 0x65)*

## 12.6 Examples for formulas

### SIMPLE LOGICAL AND LINKING BETWEEN TWO BIT VALUES



### Example

*Formula: X01.03 AND X02.03*

This formula has the status TRUE, if both **bit 3** of variable 1 and **bit 3** of variable 2 both have the value 1.

### COMPARISON OF AN ANALOGUE VALUE OR STATUS OF A VARIABLE



#### Example

*(X01.Value > X02.Value)*

### COMPARE ANALOG VALUES WITH EACH OTHER ON A LOGICAL BASIS



#### Example

*(X01.Value > X02.Value) AND (X01.Value = X02.Value)*

### COMPARE WITH VALUE BITS AND STATUS BITS



#### Example

*(X01.Value > X02.Value) AND (X01.Value = X02.Value) OR (X01.03 = X02.03)*

### COMPARE A VALUE WITH A DECIMAL OR HEXADECIMAL VALUE



#### Example

Formula: *(X01.Value = 111)*

Formula: *(X01.Value = 0x6F)*

If a hexadecimal values is used, this is later transferred to decimal by clicking on **OK**. If a decimal value is entered and confirmed, the value continues to be displayed as a decimal value after reopening.



#### Info

*It is not possible to use a comma or a period when entering values.*

## 13. XML export: Units, phases and recipes

The Batch Control module allows the export of units, phases and recipes into XML files. The structure of these files is described in the following chapters.

## 13.1 General recipe properties in the XML file

### PROPERTIES OF MASTER RECIPES

Properties of master recipes are also contained in exported control recipes.

node	Property	Description	Possible values
Recipes			<ul style="list-style-type: none"> <li>▶ Master</li> <li>▶ Control</li> <li>▶ Operation</li> </ul>
	MrId	ID of the corresponding master recipe.	
	MrName	Name of the corresponding master recipe.	
	MrDescription	Description of the corresponding master recipe.	
	MrVersion	Version of the corresponding master recipe.	
	MrSourceVersion	Original version of the master recipe.	
	MrStatus	Status of the corresponding master recipe.	<ul style="list-style-type: none"> <li>▶ 1: Enabled</li> <li>▶ 2: Ready</li> <li>▶ 3: Terminated</li> <li>▶ 4: Terminated</li> <li>▶ 8: Closed</li> </ul>
	RecipeType	Type of the recipe.	<ul style="list-style-type: none"> <li>▶ Matrix</li> <li>▶ Pfc</li> </ul>
	ApprovalTime	Time stamp of the corresponding master recipe.	
	ApprovalUserName	Name of the user who approved the recipe.	
	ApprovalUserID	ID of the user who approved the recipe.	
	OutdatedTime	Outdated time for the recipe.	
	OutdatedUserName	Name of the user who set the recipe to obsolete.	
	OutdatedUserID	ID of the user who set the recipe to obsolete.	
	Structure	Nodes for the structure of the recipe.	

## PROPERTIES OF CONTROL RECIPES

	<b>CrId</b>	ID of the control recipe.	
	<b>CrName</b>	Name of the control recipe.	
	<b>CrDescription</b>	Description of the control recipe.	
	<b>CrStatus</b>	Status of the control recipe.	<ul style="list-style-type: none"> <li>▶ 2: P</li> <li>▶ 3: R</li> <li>▶ 4: E</li> <li>▶ 7: O</li> </ul>
	<b>CrJobID</b>	Job ID of the control recipe	
	<b>CrJobIDVar</b>	Variable for the job number.	
	<b>CreationTime</b>	Time stamp of the creation of the control recipe.	
	<b>CreationUserName</b>	Name of the user who created the recipe.	
	<b>CreationUserID</b>	ID of the user who created the recipe.	
	<b>StartingTime</b>	Time stamp of the start of the control recipe.	
	<b>StartingUserName</b>	Name of the user who started the control recipe.	
	<b>StartingUserID</b>	ID of the user who started the control recipe.	

## PROPERTIES OF OPERATIONS

	<b>OpId</b>	ID of the Operation template.	
	<b>OpName</b>	Name of the Operation template.	
	<b>OpDescription</b>	Description of the operation template.	
	<b>OpType</b>	Type of operation	<ul style="list-style-type: none"> <li>▶ Matr</li> <li>▶ Pfc</li> </ul>
	<b>OpStatus</b>	Status of the operation template.	<ul style="list-style-type: none"> <li>▶ 1: E</li> <li>▶ 2: R</li> </ul>
	<b>ApprovalTime</b>	Time stamp of the approval of the operation template.	
	<b>ApprovalUserName</b>	Name of the user who approved the operation template.	

	<b>ApprovalUserID</b>	ID of the user who approved the operation template.	
	<b>Structure</b>	Nodes for the structure of the recipe.	

## 13.2 Matrix properties in the XML file

### Matrix structure

node	Property	Description	Possible value
Structure			
	Column	Column number in the matrix recipe.	<ul style="list-style-type: none"> <li>▶ Type 1: Phase</li> <li>▶ Type 2: Operation</li> </ul>
	ObjectName	Name of the linked phase or the linked operation.	
	StepInfo	Step number in the matrix recipe.	
	StepDescr	Description of the step in the matrix recipe.	

### Matrix cell properties

node	Property	Description	Possible value
Cell			<ul style="list-style-type: none"> <li>▶ Type 1: Phase</li> <li>▶ Type 2: Operation</li> </ul>
	CellActive	TRUE if the matrix cell is active at this point.	<ul style="list-style-type: none"> <li>▶ TRUE: Matrix cell active</li> <li>▶ FALSE: Matrix cell inactive</li> </ul>
	ChartId	ID of the matrix cell	
	Phase	Properties node for the phase of a matrix cell. You can find details in the General properties (on page 280) chapter.	
	ControlStrategy	Properties node for the control strategy used for a phase in a matrix cell. You can find details in the General properties (on page 280) chapter.	



### 13.3 PFC properties in the XML file

#### Structure of the PFC recipe

node	Property	Description	Possible value
Structure			
	<b>LastObjId</b>	Last-used ID of the PFC recipe.	
	<b>ColCount</b>	Number of columns in the PFC recipe.	
	<b>RowCount</b>	Number of rows in the PFC recipe.	
	<b>CenterColOffset</b>	Start coordinates of the column.	
	<b>CenterRowOffset</b>	Start coordinates of the row.	

#### Element properties

node	Property	Description	Possible value
Chart object		Element for the PFC recipe.	<ul style="list-style-type: none"> <li>▶ TYPE: 1=start element</li> <li>▶ TYPE: 2=end element</li> <li>▶ TYPE: 3=phase</li> <li>▶ TYPE: 4=transition</li> <li>▶ TYPE: 5=start branch</li> <li>▶ TYPE: 6=end branch</li> <li>▶ TYPE: 7=start parallel branch</li> <li>▶ TYPE: 8=end parallel branch</li> <li>▶ TYPE: 9=unit allocation</li> <li>▶ TYPE: 10=operation</li> <li>▶ TYPE: 11=line</li> <li>▶ TYPE: 12=jump target</li> </ul>

#### Begin element

node	Property	Description	Possible value
<b>ChartObject</b>		Element for the PFC recipe.	TYPE: 1
	<b>ChartId</b>	ID of the element in the PFC recipe.	
	<b>ChartRow</b>	Row coordinates of the element.	
	<b>ChartCol</b>	Column coordinates of the element.	

**End element**

node	Property	Description	Possible value
<b>ChartObject</b>		Element for the PFC recipe.	TYPE: 2
	<b>ChartId</b>	ID of the element in the PFC recipe.	
	<b>ChartRow</b>	Row coordinates of the element.	
	<b>ChartCol</b>	Column coordinates of the element.	

**Phase**

node	Property	Description	Possible value
<b>ChartObject</b>			TYPE: 3
	<b>ChartId</b>	ID of the element in the PFC recipe.	
	<b>ChartRow</b>	Row coordinates of the element.	
	<b>ChartCol</b>	Column coordinates of the element.	
	<b>Unit</b>	Unit name of the linked phase.	
	<b>Phase</b>	Properties node for the phase of a PFC element.	

**Transition**

node	Property	Description	Possible value
<b>ChartObject</b>			TYPE: 4
	<b>ChartId</b>	ID of the element in the PFC recipe.	
	<b>ChartRow</b>	Row coordinates of the	

		element.	
	<b>ChartCol</b>	Column coordinates of the element.	
<b>Condition</b>		Condition for the selected transitions.	
	<b>Expression</b>	The conditions are used to inform the REE of the status of the technological function in the controller.	
<b>OperandTag</b>		Addressing of the tag conditions.	
	<b>PhaseChartId</b>	ID of the phase in the condition.	
	<b>TagAddressing</b>	Tag of the phase.	

**Initial branch**

node	Property	Description	Possible value
<b>ChartObject</b>			TYPE: 5
	<b>ChartId</b>	ID of the element in the PFC recipe.	
	<b>ChartRow</b>	Row coordinates of the element.	
	<b>ChartCol</b>	Column coordinates of the element.	
<b>Cell</b>		Number of branches from left to right.	
	<b>Connector</b>	Linking points of the branches.	<ul style="list-style-type: none"> <li>▶ TRUE: Branch downwards.</li> <li>▶ FALSE: No sequence selection.</li> </ul>
	<b>ArrayOffset</b>	Initial cell of the branch.	

**End branch**

node	Property	Description	Possible value
<b>ChartObject</b>			TYPE: 6
	<b>ChartId</b>	ID of the element in the PFC recipe.	
	<b>ChartRow</b>	Row coordinates of the element.	

	<b>ChartCol</b>	Column coordinates of the element.	
<b>Cell</b>		Number of branches from left to right.	
	<b>Connector</b>	Linking points of the branches.	<ul style="list-style-type: none"> <li>▶ TRUE: Branch downwards.</li> <li>▶ FALSE: No sequence selection.</li> </ul>
	<b>ArrayOffset</b>	End cell of the branch.	

**Begin parallel branch**

node	Property	Description	Possible value
<b>ChartObject</b>			TYPE: 7
	<b>ChartId</b>	ID of the element in the PFC recipe.	
	<b>ChartRow</b>	Row coordinates of the element.	
	<b>ChartCol</b>	Column coordinates of the element.	
<b>Cell</b>		Number of branches from left to right.	
	<b>Connector</b>	Linking points of the branches.	<ul style="list-style-type: none"> <li>▶ TRUE: Branch downwards.</li> <li>▶ FALSE: No sequence selection.</li> </ul>
	<b>ArrayOffset</b>	Initial cell of the branch.	

**End parallel branch**

node	Property	Description	Possible value
<b>ChartObject</b>			TYPE: 8
	<b>ChartId</b>	ID of the element in the PFC recipe.	
	<b>ChartRow</b>	Row coordinates of the element.	
	<b>ChartCol</b>	Column coordinates of the element.	
<b>Cell</b>		Number of branches from left to right.	
	<b>Connector</b>	Linking points of the branches.	<ul style="list-style-type: none"> <li>▶ TRUE: Branch downwards.</li> </ul>

			► FALSE: No sequence selection.
	<b>ArrayOffset</b>	End cell of the branch.	

**Unit allocation**

node	Property	Description	Possible value
<b>ChartObject</b>			TYPE: 9
	<b>ChartId</b>	ID of the element in the PFC recipe.	
	<b>ChartRow</b>	Row coordinates of the element.	
	<b>ChartCol</b>	Column coordinates of the element.	
<b>Allocations</b>		Name of the unit allocation.	► Allocations ► Deallocations
	<b>Unit</b>	Name of the allocated unit.	
	<b>Global</b>	Options for the clearing of the unit.	► FALSE=clearing the last allocation ► TRUE=recipe-wide clearing

**Operation**

node	Property	Description	Possible value
<b>ChartObject</b>			TYPE: 10
	<b>ChartId</b>	ID of the element in the PFC recipe.	
	<b>ChartRow</b>	Row coordinates of the element.	
	<b>ChartCol</b>	Column coordinates of the element.	
<b>Operation</b>		Operation	
	<b>OpId</b>	ID of the operation	
	<b>OpName</b>	Name of the operation.	
	<b>OpType</b>	Type of the operation.	
	<b>OpDescription</b>	Description of the operation.	
	<b>OpStatus</b>	Status of the operation	
	<b>ApprovalTime</b>	Time stamp for approved	

		recipes.	
	<b>ApprovalName</b>	Name of the approved recipe.	
	<b>ApprovalUserId</b>	ID of the user who approved the recipe.	
<b>Structure</b>		Structure of the recipe.	<ul style="list-style-type: none"> <li>▶ PFC_Structure</li> <li>▶ Matrix_Structure</li> </ul>

**Line**

node	Property	Description	Possible value
<b>ChartObject</b>			TYPE: 11
	<b>ChartId</b>	ID of the line.	
	<b>ChartRow</b>	Row coordinates for the starting point of the line.	
	<b>ChartCol</b>	Column coordinates for the starting point of the line.	
	<b>FirstCol</b>	Column coordinates of the cell where the line starts.	
	<b>FirstRow</b>	Row coordinates of the cell where the line starts.	
	<b>SecondCol</b>	Column coordinates of the cell where the line ends.	
	<b>SecondRow</b>	Row coordinates of the cell where the line ends.	
	<b>LineSegments</b>	Column, row, type coordinates of the cells, separated by # where the complete line runs.	<ul style="list-style-type: none"> <li>▶ 0: Straight line from top to bottom</li> <li>▶ 1: 90° from the top to the right</li> <li>▶ 2: 90° from the top to the left</li> <li>▶ 3: Straight line from left to right</li> <li>▶ 4: 90° from the right to the bottom</li> <li>▶ 5: 90° from the left to the bottom</li> </ul>

**Jump target**

node	Property	Description	Possible value
------	----------	-------------	----------------

<b>ChartObject</b>			TYPE: 12
	<b>ChartId</b>	ID of the jump target.	
	<b>ChartCol</b>	Column coordinates for the starting point of the jump target.	
	<b>ChartRow</b>	Row coordinates for the starting point of the jump target.	

## 13.4 Parameter properties

### Phases

node	Property	Description	Possible value
Phase		General properties of the phase.	
	<b>PhaseName</b>	Name of the phase.	
	<b>PhaseDescr</b>	Freely definable string for detailed description of the phase.	
	<b>TOAllocation</b>	Time period, in days, hours, minutes and seconds, that is waited until the unit is allocated.	
	<b>TOInterlocking</b>	Time period in days, hours, minutes and seconds in which the condition defined in the input lock property must return the value <code>TRUE</code> .	
	<b>MinExecTime</b>	Minimum execution duration of the phase.	
	<b>ExplanationMinExecTimeNeeded</b>	Reason for minimum execution duration change necessary	<p><code>TRUE</code>: When entering value changes for this phase in Runtime, a reason for the change must be entered by the user.</p> <p><code>FALSE</code>: No reason necessary.</p>
	<b>MaxExecTime</b>	Time period in days, hours, minutes and seconds in which the condition defined in the phase done condition property must return the value <code>TRUE</code> .	



	<b>TOFollowingCond</b>	Time period in days, hours, minutes and seconds in which the phase must be deactivated.	
	<b>MinExecTimeCrModifiable</b>	Minimum execution duration of the phase in the control recipe can be modified.	<ul style="list-style-type: none"> <li>▶ TRUE: Minimum execution duration of the phase in the control recipe can be modified.</li> <li>▶ FALSE: Minimum execution duration of the phase in the control recipe cannot be modified.</li> </ul>
	<b>RtLocalFlags</b>	Flags whose properties are no longer linked to the Editor configuration. They are not overwritten when reloading.	<p>These properties are a bit-coded DWORD in that each bit stands for a property. In a 0-based index, the coding is as follows:</p> <ul style="list-style-type: none"> <li>▶ Timeout phase done condition: Bit 0 (decimal 1)</li> <li>▶ Timeout input lock: Bit 1 (decimal 2)</li> <li>▶ Timeout subsequent condition: Bit 2 (decimal 4)</li> <li>▶ Timeout unit allocation: Bit 3 (decimal 8)</li> <li>▶ Minimum execution duration: Bit 7 (decimal 128)</li> </ul>
	<b>CSActive</b>	Active control strategies	<ul style="list-style-type: none"> <li>▶ TRUE: Active control strategies</li> <li>▶ FALSE: Control</li> </ul>

			strategies inactive
	<b>CSTag</b>	control strategy tag	
	<b>CondInterlocking</b>	Node for the condition of the input lock.	
	<b>CondDone</b>	Node for phase done condition.	
	<b>CondEscaping</b>	Node for the condition of leaving the phase.	
	<b>CondPausing</b>	Node for the change from pausing to paused.	
	<b>CondHolding</b>	Node for the change from holding to held.	
	<b>CondStopping</b>	Node for the change from stopping to stopped.	
	<b>CondAborting</b>	Node for the change from aborting to aborted.	
	<b>CondRestarting</b>	Node for the condition for the change from starting to running.	
	<b>CondFailure</b>	Node for the condition to recognize a communication fault.	
	<b>CondConnReconnect</b>	Node for the condition to acknowledge a communication fault.	
	<b>CondPlcError</b>	Node for the condition for PLC error.	

**Parameter**

node	Property	Description	Possible value
<b>Day</b>			
	<b>TagName</b>	Name of the parameter.	
	<b>TagDescr</b>	Description of the tag.	

	<b>TagType</b>	Tag type	<ul style="list-style-type: none"> <li>▶ 0: Value parameter</li> <li>▶ 1: Return tag</li> <li>▶ 2: Initial parameters</li> </ul>
	<b>DataType</b>	Data Type	<ul style="list-style-type: none"> <li>▶ 0: Bool</li> <li>▶ 1: String</li> <li>▶ 2: Numeric</li> <li>▶ 3: Time period</li> </ul>
	<b>Variable</b>	Variable which is linked to the tag.	
	<b>VariableDataType</b>	Data type of the variable.	
	<b>VariableDriver</b>	Name of the EXE file of the driver of the variable.	
	<b>VariableDriverDescr</b>	Name of the driver of the variable.	
	<b>TagValue VariantType</b>	Value of the tag.	
	<b>ValueMin VariantType</b>	Minimum value of the tag.	
	<b>ValueMax VariantType</b>	Maximum value of the tag.	
	<b>VariableMin</b>	Minimum value of the variable.	
	<b>VariableMax</b>	Maximum value of the variable.	
	<b>MeasUnit</b>	Unit of the variable.	
	<b>MaxStringLength</b>	String length as engineered at the variable.	
	<b>EditableInRecipe</b>	Can be changed in the master recipe	<ul style="list-style-type: none"> <li>▶ TRUE: Tag values can be changed in the master recipe.</li> <li>▶ FALSE: Tag values cannot be changed in the master recipe.</li> </ul>
	<b>ExplanationNeeded</b>	Reason for value change	<ul style="list-style-type: none"> <li>▶ TRUE: When</li> </ul>

		necessary	<p>entering value changes for these parameters in Runtime, a reason for the change must be entered by the user.</p> <ul style="list-style-type: none"> <li>▶ FALSE: Reason not required.</li> </ul>
	<b>TagModified</b>	Tag value changed.	<ul style="list-style-type: none"> <li>▶ TRUE: Tag value changed</li> <li>▶ FALSE: Tag value not changed.</li> </ul>
	<b>EditableInCr</b>	Tag value can be changed in control recipe	<ul style="list-style-type: none"> <li>▶ TRUE: Tag value can be changed in the control recipe.</li> <li>▶ FALSE: Tag value cannot be changed in the control recipe.</li> </ul>
	<b>UseKeyboard</b>	Use screen Keyboard	<ul style="list-style-type: none"> <li>▶ TRUE: Keyboard screen is used for this parameter.</li> <li>▶ FALSE: Keyboard screen is not used for this parameter.</li> </ul>
	<b>KeyboardPictureName</b>	Name of the selected keyboard screen. Only available if the keyboard screen property has been activated.	

## Reactions

node	Property	Description	Possible value
<b>Reaction</b>		Contains general properties for reactions.	
	<b>EventType</b>	Selection of the event type when the reaction is to be executed. For each event several reactions can be defined.	<ul style="list-style-type: none"> <li>▶ 0: Exit Runtime initiated</li> <li>▶ 1: Input interlocking blocked</li> <li>▶ 2: Waiting period</li> </ul>

			input interlocking exceeded ▶ 3: Phase activated ▶ 4: Maximum execution period exceeded ▶ 5: Phase deactivated ▶ 6: Waiting period following condition exceeded ▶ 7: Linked variable interrupted ▶ 8: Status change: Continue ▶ 9: Status change: Running ▶ 10: Status change: Pausing ▶ 11: Status change: Paused ▶ 12: Status change: Executed ▶ 13: Status change: Restarting ▶ 14: Status change: Holding ▶ 15: Status change: Held ▶ 16: Status change: Stopping ▶ 17: Status change: Stopped ▶ 18: Status change: Aborting ▶ 19: Status change: Aborted ▶ 20: Mode change: Automatic ▶ 21: Mode change: Semi-automatic ▶ 22: Mode change:
--	--	--	--

			<p>Manual</p> <ul style="list-style-type: none"><li>▶ 28: Waiting period unit allocation exceeded</li><li>▶ 29: Phase started multiple times</li><li>▶ 30: Unit allocation not possible</li><li>▶ 31: Finished writing value tags</li><li>▶ 32: Phase done condition completed</li><li>▶ 33: Command tag without value</li><li>▶ 34: Runtime restart</li><li>▶ 35: Escape condition started</li><li>▶ 36: Escape condition fulfilled</li><li>▶ 37: Loss of communication</li><li>▶ 38: Loss of communication fixed</li><li>▶ 39: Loss of communication acknowledged</li><li>▶ 40: Phase started</li><li>▶ 41: PLC error</li><li>▶ 42: PLC error rectified</li><li>▶ 43: PLC error rectified by deactivating the phase</li><li>▶ 44: Input lock checked successfully</li></ul>
--	--	--	---

	<b>ReactionDescr</b>	Description per reaction.	
	<b>ReactionPrio</b>	Displays the execution order when several reactions were defined for the same event. The order is defined by the position in the list and only displayed here.	
	<b>ModeCommand</b>	This reaction can be controlled in the Runtime of the execution mode as a reaction to the event.	<ul style="list-style-type: none"> <li>▶ 0: Ignore</li> <li>▶ 1: Automatic</li> <li>▶ 2: Semi-automatic</li> <li>▶ 3: Manual</li> </ul>
	<b>StateCommand</b>	Recipe or phase command which is executed in the Runtime at the occurrence of the event.	<ul style="list-style-type: none"> <li>▶ 0: Ignore</li> <li>▶ 2: Pause recipe</li> <li>▶ 3: Recipe resuming</li> <li>▶ 4: Hold recipe</li> <li>▶ 5: Restart recipe</li> <li>▶ 6: Recipe stopping</li> <li>▶ 7: Recipe aborting</li> <li>▶ 8: Phase pausing</li> <li>▶ 9: Phase holding</li> <li>▶ 10: Restart phase</li> <li>▶ 11: Phase resuming</li> </ul>
	<b>Function</b>	Function that is to be carried out.	
	<b>CelEnabled</b>	Create CEL entry	<p>TRUE: The text defined in the CEL message text property is entered in the Chronological Event List, CEL.</p> <p>FALSE: No entry in the CEL.</p>
	<b>CelMsg</b>	CEL message text. Only available if CEL entry property is active.	
	<b>CelGroup</b>	Allocation of a pre-existing alarm/event group to CEL messages for the selected	

		event.	
	<b>CelClass</b>	Allocation of a pre-existing alarm/event class to CEL messages for the selected event.	
	<b>SourceTag</b>	Tag whose value is written to the tag selected in the destination tag property.	
	<b>DestinationTag</b>	Tag to which the value of the tag defined in source tag property is transferred.	
	<b>SetTag</b>	Tag on which the defined set value should be written as reaction of the event.	
	<b>SetValueNum</b>	Set value for numeric parameter.	
	<b>SetValueStr</b>	Set value for <code>string</code> tag.	
	<b>SetValueBool</b>	Set value for a binary tag.	<ul style="list-style-type: none"> <li>▶ 0: Off</li> <li>▶ 1: On</li> <li>▶ 4: Toggle</li> </ul>
	<b>SetValueDuration</b>	Set value in days, hours, minutes and seconds for duration tags. The time span is written as value in seconds on the variable linked at the tag.	
	<b>ExecuteBeforeStartEvent</b>	Allow execution before start event.	<p>TRUE: The event can be executed before the phase started event.</p> <p>FALSE: The event can only be started if the phase started event has been executed.</p>

**Control strategies**

node	Property	Description	Possible value
<b>ControlStrateg</b>		Contains properties for	



y		the configuration of the control strategies.	
	<b>CSName</b>	Name of the control strategy.	
	<b>CSDescription</b>	Description of the control strategy.	
	<b>ActiveCSNumber</b>	Unique number of the control strategy for identification within this phase.	
	<b>CSTag</b>	Node for the definition of a parameter linked for this control strategy.	

#### Control strategies

node	Property	Description	Possible value
<b>ConfiguredControlStrategy</b>		Contains properties for the configured control strategy in the recipe.	
	<b>CSName</b>	Name of the control strategy.	
	<b>CSDescription</b>	Description of the control strategy.	
	<b>ActiveCSNumber</b>	Unique number of the control strategy for identification within this phase.	

## 14. CEL

Messages, information and errors for recipes, units, commands, reactions, events etc. are saved and displayed in the **Chronological Event List (CEL)**.

## GROUPS AND CLASSES

CEL information can be allocated to groups and classes. These correspond to the **alarm/event groups** and **alarm/event classes** created in the project. The respective group or class is also entered in the CEL and can be used for filtering and grouping.

Groups and classes are allocated for actions for:

- ▶ Recipes
- ▶ Commands
- ▶ Value changes
- ▶ Jumping, forcing and step-by-step execution

## RECIPES

The set group or class is entered into the CEL for the following recipe actions:

- ▶ Master recipes list
  - New
  - Create new version
  - Delete
  - Duplicate
  - Rename
  - Release
  - Highlight as outdated
  - New control recipe
- ▶ Control recipe list
  - Rename
  - Duplicate
  - Delete
- ▶ Operation list
  - New
  - Rename
  - Duplicate
  - Delete
  - Release

## COMMANDS

The set group or class is entered for all recipe commands, phase commands, mode switches and restart messages.

## VALUE CHANGES

The set group or class is entered in the event of value changes to parameters in master recipes in test mode and for control recipes.

## JUMPING, FORCING AND STEP-BY-STEP EXECUTION

The set group or class is entered for all manual steps, when jumping steps and when executing steps.

## ALLOCATION

Groups and classes are allocated

- ▶ In general: in the properties of the **Batch Control** node in the **CEL groups/classes** group
- ▶ Reactions: in the properties of the event of a reaction in the **Reactions/CEL entry** group

# 15. Failure handling

If communication failures or PLC errors occur, these can be detected in Runtime using a formula configured in the editor. In the event of a communication error, the phase concerned is paused or held (depending on configuration).

## 15.1 communication errors

Detection of problems can be specially configured for each phase, because phases can also run on different PLCs. The reaction to a communication error is defined globally for all phases. In addition, a reaction to the `communication error` (on page 32) event can be configured.

To configure the detection of communication errors:

1. highlight the desired phase
2. Click, in the **Loss of communication** property group, on the **Loss of communication** property
3. In the formula editor (on page 255) that opens, define the condition for detecting communication errors
4. click on property **Loss of communication acknowledged**
5. Define the condition to detect the reestablishment of communication
6. Navigate to the properties for the Batch Control module
7. In the **General/Loss of communication** group, open the **Action for loss of communication** property
8. Select the desired reaction to a communication error from the drop-down list
  - Phase holding
  - Phase pausing

## CHECKING IN RUNTIME

When starting a phase or when restarting Runtime, 60 seconds is waited for values for the formula to check communication. If no values are received within this waiting time, a communication error is assumed.

Another check for communication errors is made if the phase is in a new step.

When restarting Runtime, the waiting for values can be in many steps, because the communication is started again in the middle of the process. A check for communication failures is also made at areas where a check for faults is made.

## REESTABLISHMENT

### CHECKING THE FORMULA FOR REESTABLISHING COMMUNICATION

A loss of communication to the PLC is determined if the value of the formula for the **Loss of communication** property is **TRUE**. Waiting is carried out until communication has been reestablished, i.e. the value of the formula is **FALSE**. After this, waiting is continued until the value of the formula for the **Loss of communication acknowledged** property is **TRUE** or empty. The execution of the object can be continued from this time only.

In the time between the loss of communication and reestablishment of communication:

- ▶ No phase commands can be sent to the phase concerned
- ▶ If the status is changed directly, transient conditions are not checked
- ▶ No reactions are triggered for a status change

- ▶ the procedure path of the phase concerned remains the same when a `continue` or `restart` global command is executed
- ▶ The subsequent execution positions are offset until the communication to the phase has been reestablished

## SKIPPING THE REESTABLISHMENT OF COMMUNICATION

In the event of a communication error, this can be skipped both whilst the error is active and whilst communication is being reestablished. In both cases, the communication is considered reestablished. All reactions to the skipped steps are triggered. The skipping is displayed in the tooltip.

If a communication error has been skipped, then a new one cannot occur until the value of the formula for the **Loss of communication** property has not changed to `FALSE`.

## COUNTER FOR THE RECIPE AND OPERATION FOR REESTABLISHED COMMUNICATION

In the status line of the recipe editor, there is information about errors in the recipe available above the symbol of the error display and its tooltip. Only the information with the highest priority is shown. Active errors and historical errors are not shown at the same time.

Prioritization and coding of the error display in the tooltip:

Priority	Color	Description
1	red	Currently $\times$ errors active.
2	red	Currently $\times$ communication errors not yet acknowledged.
3	yellow	Errors were present.
4	green	faultless

Note: "**Currently  $\times$  errors active**" counts both procedure errors and communication errors.

## 15.2 PLC error

The detection of PLC errors can be configured for each phase.

To configure the detection of PLC errors in Runtime:

1. Configure a variable that reports the PLC error in zenon
2. Highlight the desired phase in the Editor in the Batch Control module.
3. Click, in the **Loss of communication** property group, on the **PLC error** property
4. In the formula editor (on page 255) that opens, define the condition for detecting PLC errors using the configured variables.

Note: The formula can be displayed in the report (on page 250).

To display the number of active and rectified PLC errors in a screen:

1. Select the unit for whose phases PLC error detection has been configured.
2. Go to the property group **Runtime information/Error**.
3. Configure the variables for the properties.
  - **Number of active PLC errors**
  - **Number of rectified PLC errors**
4. Configure the evaluation and display of variables in Runtime.

## CHECKING IN RUNTIME

If a formula is linked for PLC errors, this is checked whilst the phase is active. Checking starts once communication with the PLC has been established and lasts until the phase has been deactivated. The connection is considered established if all values for the formula for detecting communication problems have been received.

If a communication error (on page 291) occurs, the PLC error is not checked as long as the communication error has not been acknowledged. Once it has been acknowledged, checking continues with the currently-pending value.

PLC errors and the rectification of these are displayed with their own symbol in the phase or in the operation and each displayed with an entry in the tooltip.

If there is a PLC error when a phase is ended, this is amended to *rectified*. In contrast to a normal deactivation of a PLC error, a separate event is sent: `PLC error rectified by deactivation of the phase`. This only applies for the actual closing, but not for a restart. When restarting, no new event is sent for PLC errors that continue to be active.

# 16. Error Handling

Warnings and error messages are written in log files and can be analyzed with the Diagnosis Viewer. For this you must activate module Batch Control in the filter settings.

## SYSTEM DRIVER VARIABLE

Batch Control provides the system driver **SYSDRV** with information via system driver variables. For information about their messages see manual **SYSDRV** ([sysdrv.chm::/22853.htm](#)) in chapter **Topic - Batch Control** ([sysdrv.chm::/34270.htm](#)).

## **RECONSTRUCTION OF INDEX FILES**

If the index file is not read when Runtime is started or it does not exist, it is reconstructed upon starting. During this time, requests from clients that need an ID or are processing an ID cannot be processed. Modifications are rejected, queries are answered after the index has been created.