www.asix.com.pl

Asix.Evo - System of Alarms

Doc. No ENP7E008
Version: 2012-08-24

ASKOM

ASKOM Sp. z o. o., ul. Józefa Sowińskiego 13, 44-121 Gliwice,
tel. +48 32 3018100, fax +48 32 3018101,

http://www.askom.com.pl, e-mail: office@askom.com.pl

Table of Contents

1 Basic Concepts 3
2 Alarm System Parameterization 5
3 Workstation Roles 6
4 Alarm Definition Database 8
4.1. Attributes 8
4.2. Alarm Description Texts 10
4.3. Alarm Groups 11
4.4. Import from Excel Spreadsheets 12
5 Alarm Detection Strategies 15
5.1. Bitwise Strategy 16
5.2. Conditional Strategy 17
5.3. Buffered Strategy 18
5.4. Asix Strategy 19
5.5. External Strategy 21
5.6. External Global Strategy 21
6 Active Alarm Log 22
7 Historical Alarm Log 23
7.1. SQL Alarm Archive 23
8 Alarm State Visualization 25
8.1. Active Alarm Tables 25
8.2. Historical Alarm Tables 25
8.3. Other Objects 26
9 Alarm Management 27
9.1. Alarm Acknowledgement 27
9.2. Alarm Exclusions 27
9.3. One-time Exclusions 28
9.4. Alarm Filtering 29
9.5. Sound Signalling 30
9.6. Printing 30
10 Integration with AsAlert 32

1 Basic Concepts

The Asix.Evo system features built-in flexible, customisable alarm management system which provides full functionality for handling the emergency situations and events occurring during the technological process.

The Asix.Evo alarm system is organised in the so-called domains. The **domain** is identified by the name and virtually is fully functional and independent of other domains standalone system. The domains allow for logical and functional allocation of alarms to some independent areas. **In a typical application, the application uses a single domain to handle all the alarms**. In more complex cases where the application supports multiple independent installation, it may be useful to divide the alarms on several domains.

Note:

In the Asix applications, the so-called network name of the alarm container was the direct equivalent of the domain. The main difference is the ability to use multiple, simultaneously active Asix.Evo domains.

The alarm states are identified by means of the **alarm detection strategy**. There are different types of strategies detecting alarms the by various methods.

Detected alarms are stored in two **logs**:

- **Active alarm log**

It stores information about all currently active alarms. When properly parameterized it may also contain information on the alarms recently ended.

- **Historical alarm log**

It provides long term storing of the information about all detected alarms.

The registered alarms can be viewed using **tables of active and historical alarms**. Active alarm statuses can also be shown directly on the synoptic diagrams.

For applications using multiple Asix.Evo workstations, the state of the alarm system is synchronized between these workstations. The workstations being activated reconcile

the active alarm state and update the historical alarm archive. The alarm handling carried out by the operator is also synchronized, such as acknowledgement of the alarm at the one of workstation is immediately visible on the other workstations.

The Asix.Evo application alarms are identified by **name**. All alarm parameters (description, category, detection method, etc.) are stored in the **alarm definition database**. The definitions can be imported from the Excel worksheets.

2 Alarm System Parameterization

The alarm system parameterization is carried out in the *Alarm system configuration* panel opened by selecting the *Alarms System* node of the *Application Explorer* panel.

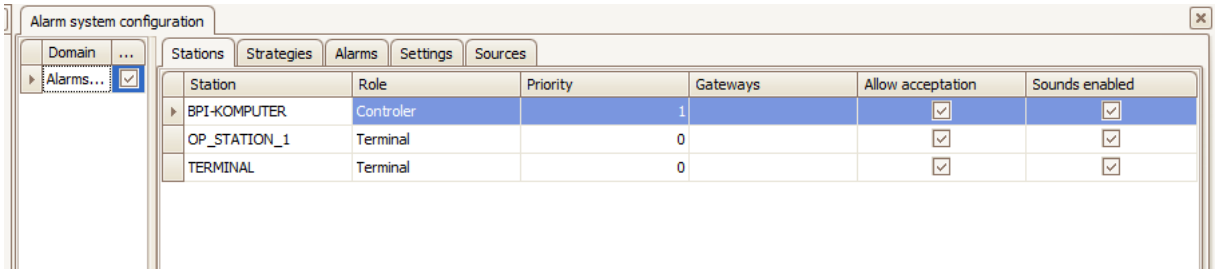


Fig. Alarm System Configuration Panel.

The configuration panel consists of two sections. The left one is used to create and select domains. In the right one, the domain parameterization can be done using the switchable tabs. The particular tabs allow for:

- *Stations*

Determination of the role performed in the alarm system by the individual workstations defined within the application.

- *Strategies*

Defining the alarm detection strategy.

- *Alarms*

Defining the domain alarms.

- *Settings*

The parameterization of various elements of the domain, such alarm log operation method.

- *Sources*

Determining the sources of alarm definitions (Excel worksheets), and importing a definitions from these sources.

3 Workstation Roles

Each of the workstation participating in the domain operation performs a specific role in that domain. The workstations may perform one of the following roles:

- *Controller*

The workstation participates actively in the alarm handling process. The controllers are responsible for the detection of alarms and sending information about the alarm statuses between workstations. Each domain must have at least one controller type workstation. In order to provide redundancy, a larger number of controllers may be used. In this case, the active controller is selected automatically during the operation. Other controller workstations temporarily switch to the controller operation mode similar to the terminal workstations. However, they are still ready to take over the active controller function. The alarm detection strategies may be also running on them.

Each controller has an priority assigned. In situations where there is an active controller reconciled, the controller of the highest priority is chosen, provided that it is possible.

- *Terminal*

The terminal workstations are used only to show the alarm status and provide the operator's functions. The terminal can not operate independently, must be connected to the controller. The connection may be direct or via a gateway. In the latter case, the gateway workstation names should be specified in the workstation parameterization, in the *Gateways* column. If the gateways are not specified for the terminal, this terminal will connect only with the domain active controller.

- *Gateway*

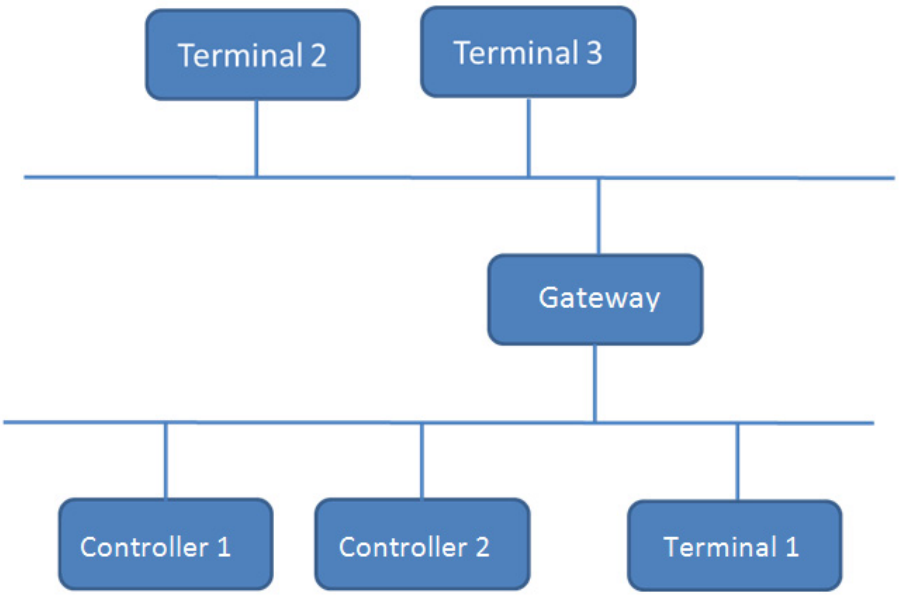
The alarm handling functionality is the same as in the case of terminals. The gateway may additionally serve as an intermediary in the exchange of data between the terminal workstations and the controller. The gateways are used when the terminals are in a different network than the controller workstations. Another application is to use the gateway to reduce the controllers usage (in particular when the number of terminals is large).

- *None*

If no role has been selected, this workstation does not participate in the domain operation and do not have access to the domain alarms.

In the simplest case, the application may consist of a single workstation acting as a domain controller. The following diagram shows the configuration with the two

redundant controllers. One terminal retrieves the alarm details directly from controllers. Two other terminals connected to the network, which has no direct connection with the controllers, use the workstation acting as a gateway.



Some restrictions on the roles available in the alarm system are introduced by the role which the workstation performs in the communication system. Only the workstation which is the communication server may act as a controller or gateway.

4 Alarm Definition Database

All alarms used in the Asix.Evo application must be defined in advance. It can be done via the alarm table in the *Alarms* tab.

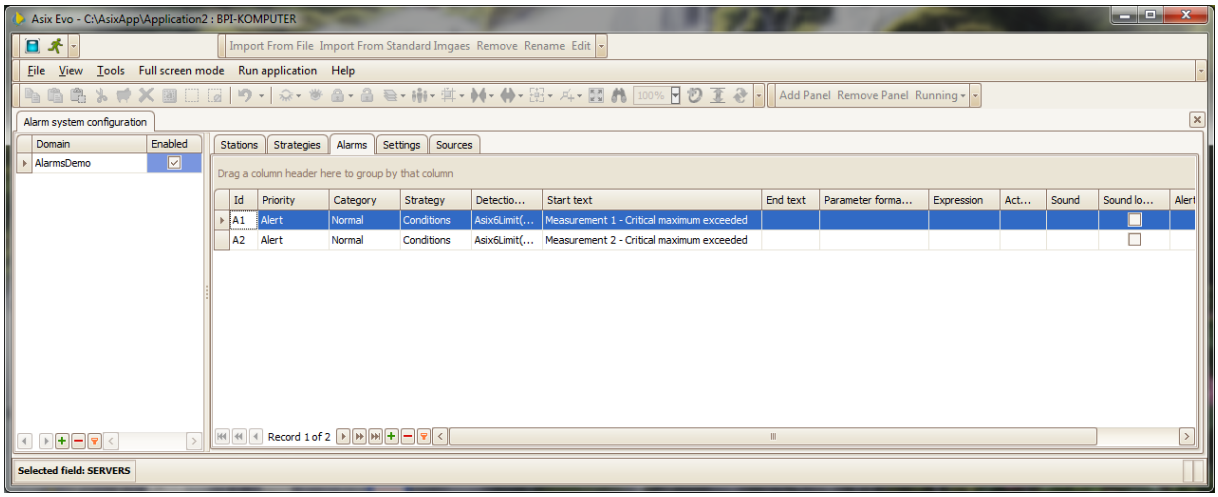


Fig. Definition of Alarms.

4.1. Attributes

The alarm definition consists of attribute set. The meaning of each attribute is described in the table below.

Name	**Meaning**
ID	Alarm Text ID
Priority	The alarm priority indicating the alarm importance. The following priorities are available (in descending order of importance): • Critical • Urgent • Alert • Warning • Message Alarms of different priorities are displayed in the alarm tables, highlighted with a different colours. A critical and urgent alarms need a selective acknowledgement.
Category	Specifies the alarm category. Two categories are available: *Normal* and *Start only*. The difference between the *Start only* and *Normal* alarm is that only the beginning event is detected for the *Start only*. The *Start*

	only alarms have no ends.
Strategy	Specifies the name of the strategy used for the alarm detection. The detection strategy parameterization method is described in the separate section.
Detection parameters	Specifies the parameters used for alarm detection. The attribute form depends on the type of strategy used. The detection strategy parameterization method is described in the separate section.
Start text	Specifies the text describing the alarm beginning event. The text creation and use are described in the separate section.
End text	Specifies the optional text describing the alarm end event. If the text is not defined, the alarm end is described in the same way as its beginning. The text creation and use are described in the separate section.
Parameter formatters	Upon detection of an alarm, the alarm may forward some additional detection parameters altogether with the alarm notification. The set and importance of parameters is specific to a specific type of strategy. At the time of notification, the parameter values are converted to text format. The *Parameter formatters* attribute allow controlling the conversion. This attribute is a collection of formatting texts; the successive elements are used to format the successive detection parameters. The formatting text is as follows: [-]*Width:Format* *Width* specifies the minimum width of the text generated. If necessary, the text is supplemented by spaces. An optional minus sign indicates alignment of the field to the left. *Format* specifies the conversion method being in accordance with the one used in the .NET platform. Example: 0:f3 Conversion into floating point format rounded to three decimal places. The length of the text corresponds to the converted parameter value. 6:d Conversion into an integer. The field will have 6 characters as minimum, and a number will be aligned to the right.
Expression	Specifies an expression whose value at the time of the beginning and end of the alarm is recorded in the log. The value of the expression can be displayed in the contents of the text of the alarm beginning or end. This value is also shown in the alarm details window accessible via the buttons of alarm tables.
Action	Specifies the action associated with the alarm. The operator may execute the action pushing the button available in the active alarm table.
Sound	Specifies the name of the sound that should be played at the alarm occurrence.

Sound looping	Determines if an alarm sound should be repeated until it will be muted by the operator. The sound can be muted with the button located in the active alarm table or by the StopSound operator action.
Alert mode	Specifies if an alarm is to be signalled by means of SMS or e-mail messages through the AsAlert program. The use of AsAlert is described in the separate section.
Alert recipients	Specifies the recipients to which, through the AsAlert program, a notification should be sent. The use of AsAlert is described in the separate section.
Grouping attributes	The names and number of grouping attributes are determined by the application designer. The grouping rules are described in the separate section.

4.2. Alarm Description Texts

An alarm definition contains two texts describing the alarm - the text for the beginning event and the text for the end event. The end event text is optional. The alarm texts are used in the alarm table objects. The application designer chooses which texts are to be displayed in the alarm line. It is also possible to choose the option that displays only one column of the text, but its contents is determined by the context. If the event relates to the alarm beginning, the beginning text is displayed. If the event relates to the alarm end, the end text is displayed (if defined).

The alarm text can include additional information. This may be the expression value specified in the *Expression* attribute or value of the alarm detection parameter (notified by the alarm strategy, scripts or operator actions). To insert this additional information to the alarm text, the tags in form of {*number*} should be inserted in the text. *Number* 0 means insertion of the expression value specified in the *Expression* attribute. The numbers 1 and higher relate to the successive alarm detection parameters.

Example:

Exceeded limit of {2}, value {1}, machine state {0}

The machine state is taken from the *Expression* attribute, and the limit and the value are derived from the alarm notification (e.g. in the user script).

The elements inserted in the alarm texts can not be additionally formatted. This is because both the expression and detection parameters are converted into a text at the time of the alarm notification, and the alarm text is used only at the time of its display. If conversion of other type than the default is needed, the *Parameter formatters* attribute should be properly defined for the detection parameters, and appropriate conversion should be ensured for the expression, inside this expression (e.g. using the *Format* function).

4.3. Alarm Groups

The Asix.Evo alarm system features flexible alarm selection mechanism using the so-called grouping attributes. The first step in parameterization of alarm grouping is to create a set of grouping attributes. The following picture shows the section of the *Settings* tab, where it is possible to add any number of attributes used for grouping. Each attribute is defined by a group ID and descriptive text. The ID is of organizing significance and the descriptive text is used in all user interfaces.

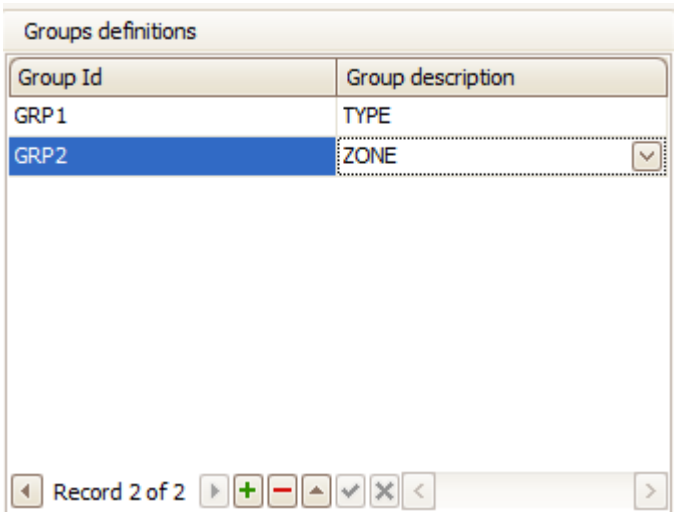


Fig. Definition of Alarm Groups.

When the grouping attributes are defined, the alarm definitions of the *Alarms* tab will be expanded with additional columns.

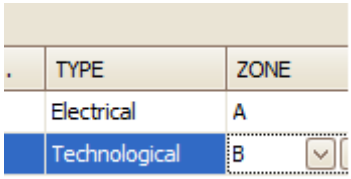


Fig. Column of Groups.

Each alarm should be provided with the grouping attribute values. Alarms of the same attribute value belong to the same group.

In the application run mode, the user may select the alarms shown in the historical / active alarm table using pre-defined groups. The selection criteria button opens the window in which the user may select the groups to be shown.

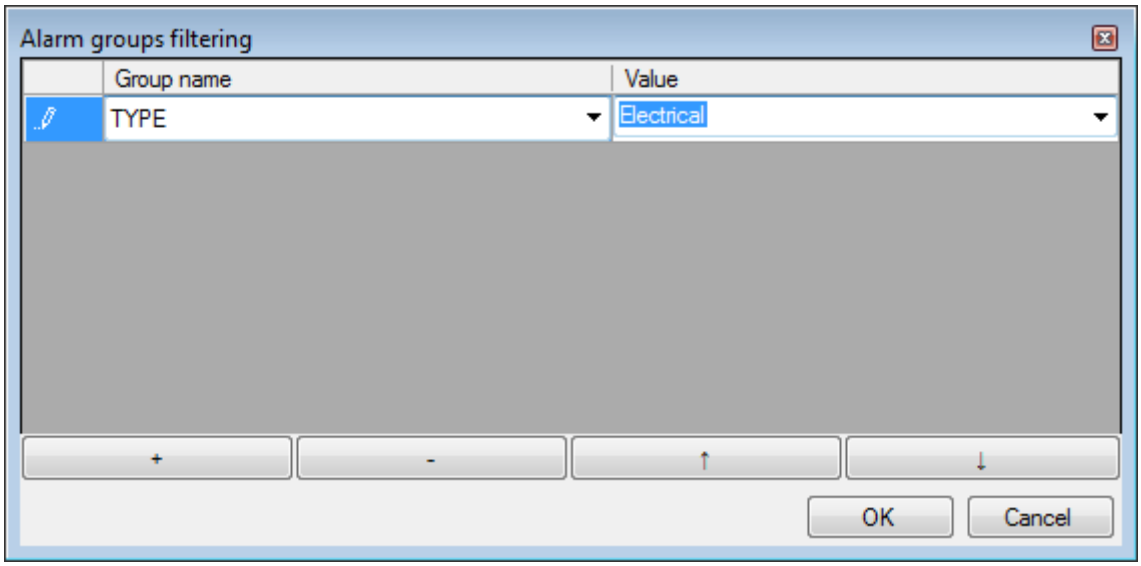


Fig. The 'Alarm groups filtering' Window.

The above example should be interpreted as follows: these alarms will be shown which *TYPE* attribute is equal to *Electrical.*

4.4. Import from Excel Spreadsheets

For applications with higher number of alarms it can be useful to define alarms in MS Excel spreadsheets and to generate alarm definition databases based on them.

Support for the import function of Ms Excel spreadsheet is provided by the *Sources* tab.

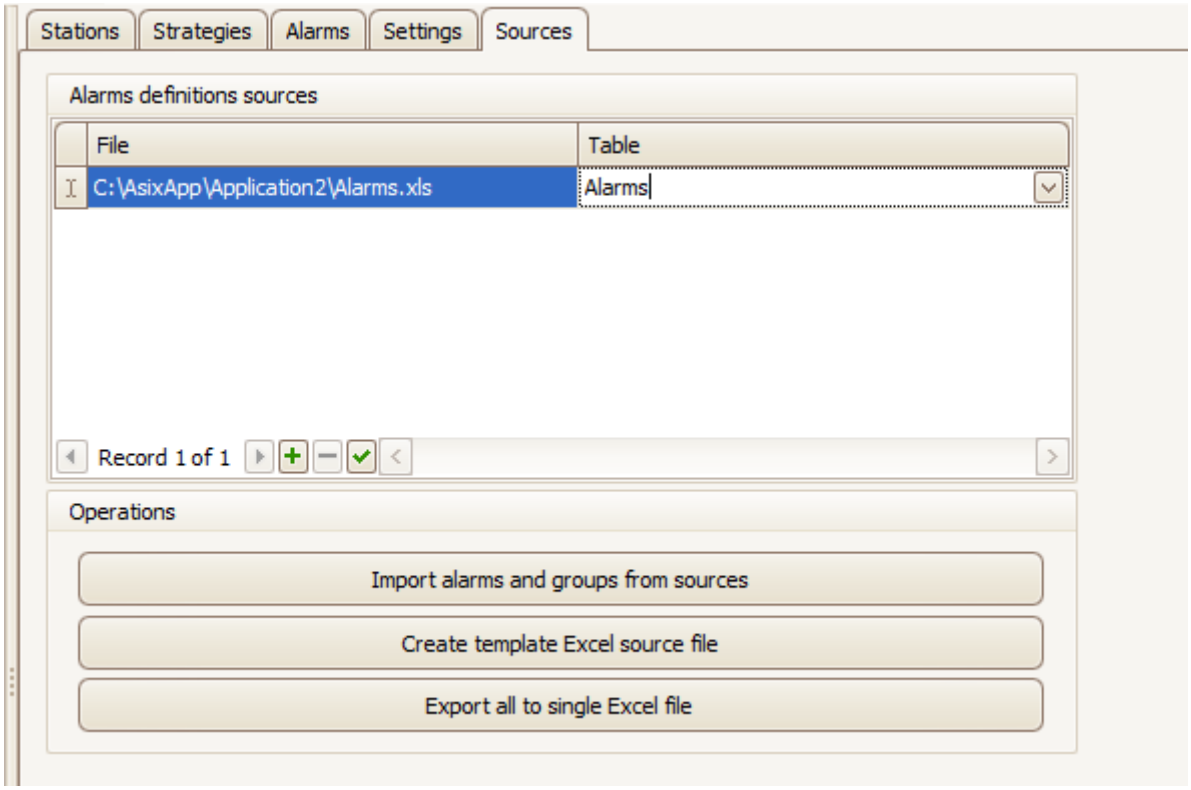


Fig. Accessing to the MS Excel import function.

It is possible to define any number of source spreadsheets. After pressing the *Import alarms and groups from source* button, the alarm definition database based on the current contents of the spreadsheets will be created.

The picture below shows a sample spreadsheet of alarm definitions.

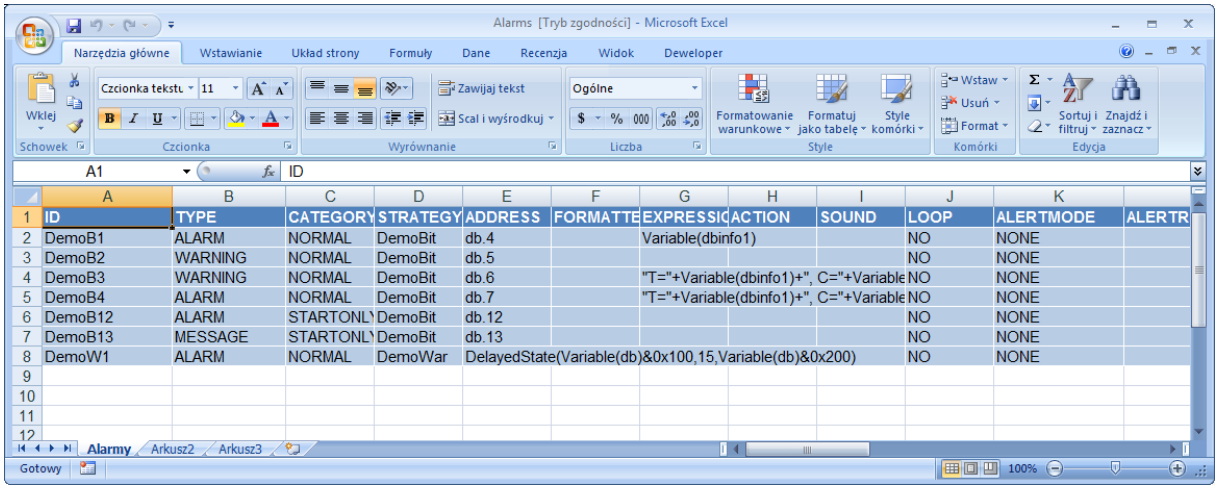


Fig. Alarm Definition Spreadsheet.

The easiest way to create a proper model of alarm definition spreadsheet is to use the *Create template Excel source file* button. **The model created will take into account the languages defined in the application and a grouping attributes added**.

It is also possible to create Excel file including all the currently defined alarms, using the *Export all to single Excel file* button.

5 Alarm Detection Strategies

Each alarm domain may use free number of alarm detection strategies. The strategies may be parameterised in the *Strategies* tab of the alarm system parameterization panel.

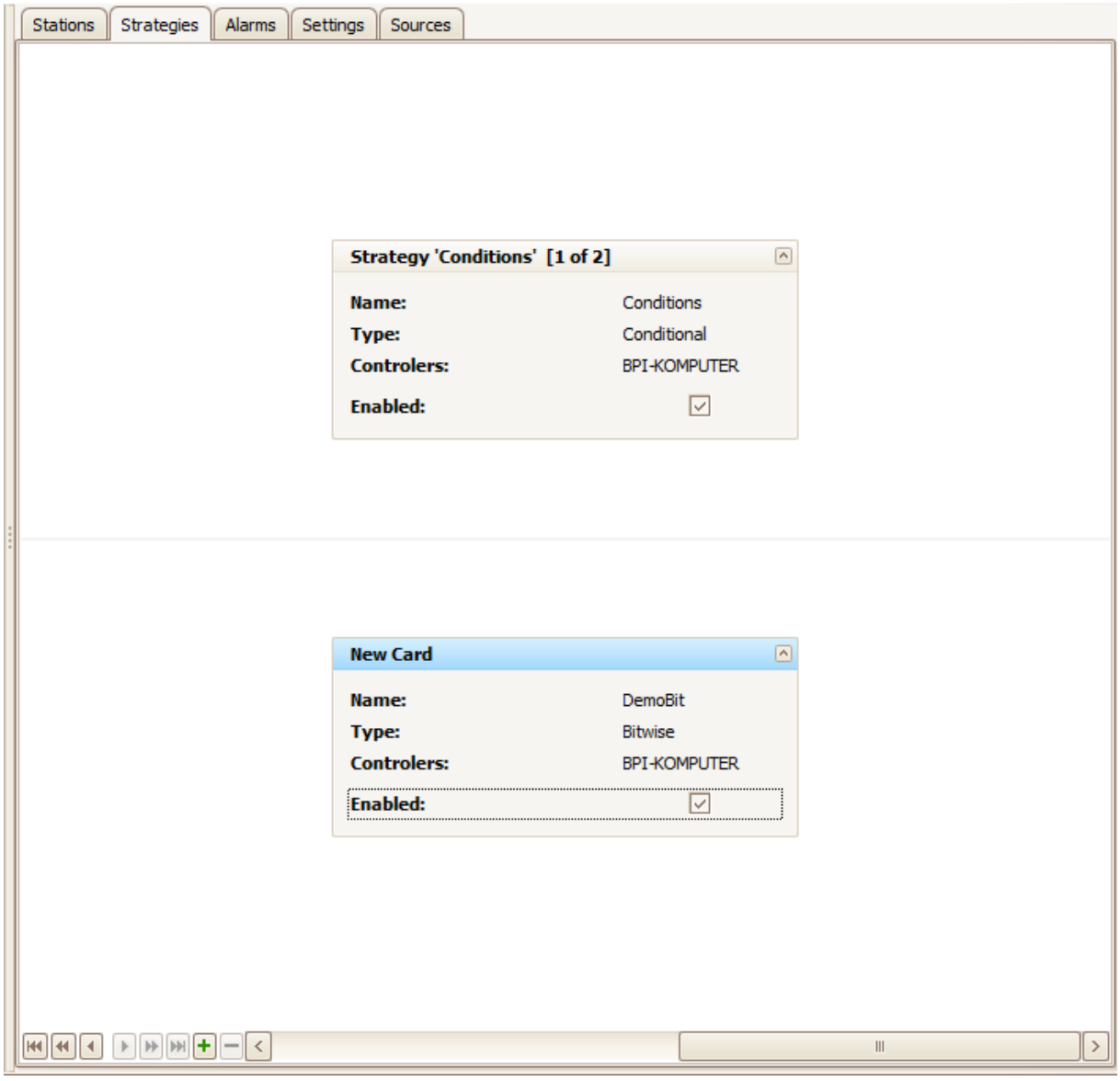


Fig. The 'Strategies' Tab of the Alarm System Parameterization Panel.

The alarm detection strategy my be, by analogy, considered as a communication channel which is used to report an alarms. Each strategy can be run only on a single controller (active or passive) within the domain. The active controller always determines on which controller the strategy is to be run. Each strategy is defined by:

- *Name*

The name identifying the strategy. The alarms which in their definition, in the *Strategy* attribute have a name specified, are detected by this strategy.

- *Type*

Specifies the type of strategy, and consequently the alarm detection method. Available types of strategies are described later in this documentation.

- *Controllers*

Specifies a list of controller workstation names on which the strategy may be run. An active controller with appropriate permissions will run the strategy on its own workstation. If not, it chooses one of the controllers listed.

- *Enabled*

Specifies whether the strategy is to be used.

- *Other parameters*

Depending on the type of the strategy chosen the additional parameters may be required to set.

5.1. Bitwise Strategy

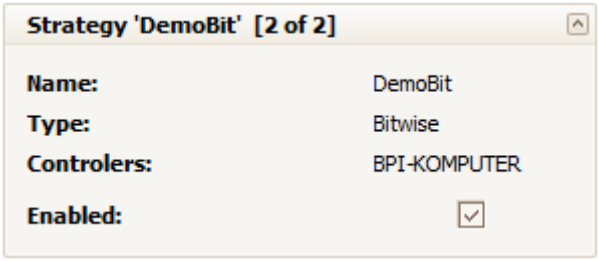


Fig. Bitwise Strategy.

The bitwise strategy is designed to detect alarms on the basis of a single bit of a particular process variable.

For the alarms controlled by the bit strategy, the *Detection parameters* attribute in the alarm definition must have the following form:

variable_name.bit_no [r]

variable_name specifies the name of the process variable controlled, *bit_no* determines the number of the bit controlled (least significant bit is marked with 0 number), and the *r* optional letter indicates the use of reversed detection logic (change the bit value from 1 to 0 indicates on the alarm beginning).

The state of many process variables may be controlled within a single strategy. There is no limit on the size of the variables controlled.

Example:

al100.1r

The alarm beginning will be detected at the change of the bit numbered 1 (bitmask 0x0002) of the *al100* variable from 1 to 0.

5.2. Conditional Strategy

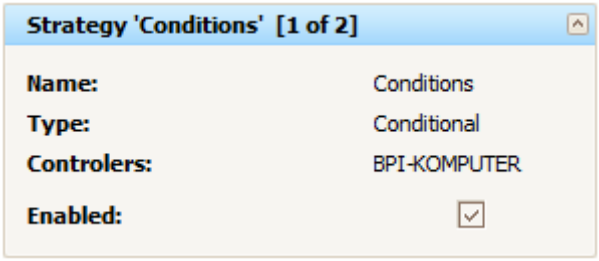


Fig. Conditional Strategy.

The conditional strategy is designed to detect alarms based on monitoring the value of any expression specified as the alarm detection parameter.

For the alarms controlled by the conditional strategy, the *Detection parameters* attribute in the alarm definition must have an arithmetic expression form. The expression monitored must return a boolean type result. The expression value change from *false* to true, indicates the alarm beginning while the reversed change indicates the alarm end.

The function set provided by Asix.Evo includes the functions dedicated specifically for use within a conditional strategy:

- DeadBand - limit exceeding control with hysteresis
- DelayedState - detection of lack of anticipated process state changes
- RateOfChange - monitoring the expression value change rate

Example:

Variable (Speed1)> 1000 && (Variable (SMode) & 1)

The alarm is activate when the *Speed1* variable value will be greater than 0 and the lower bit of the *SMode* variable will be set.

DelayedState (Variable (MState) & 0x100, 15, Variable (MState) & 0x200)

The alarm is active when the bit numbered 9 of the *MState* variable will not be set within 15 seconds after setting the bit numbered 8.

5.3. Buffered Strategy

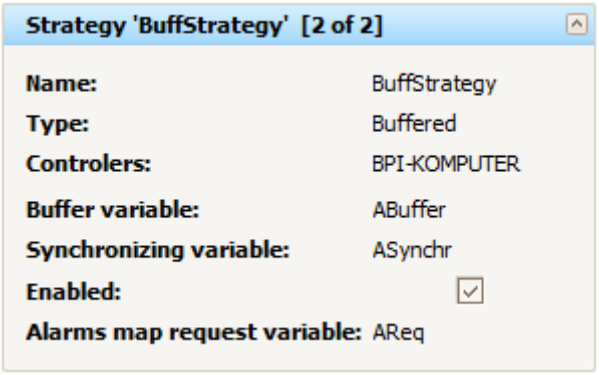


Fig. Buffered Strategy.

The *buffered* strategy is a special strategy that allows for precise detection of alarms with millisecond resolution while ensuring the event sequence maintaining. However, the strategy requires integration with the driver software. The driver is responsible for the alarm detection and alarm time stamping. The exact description of the data exchange protocol is included in the help file of the Asix 6 system (DrBUFOR.PDF).

The buffer strategy requires additional parameters related to the data exchange protocol used between the controller and Asix.Evo application:

- Buffer variable - the variable through which alarm status information is passed. The value of the variable must be an array of 8 - or 16-bit unsigned integers.

- Synchronizing variable - the variable used to synchronise access to the alarm buffer. The value of the variable must be 16-bit unsigned integer.
- Alarms map request variable - the variable through which Asix.Evo may inquire sending the current statuses of all alarms. The value of the variable must be 16-bit unsigned integer.

For the alarms controlled by a buffer strategy, the *Detection parameters* attribute in the alarm definition must be next alarm number available in the alarm pool supported by the strategy. The first alarm of the strategy is 0.

5.4. Asix Strategy

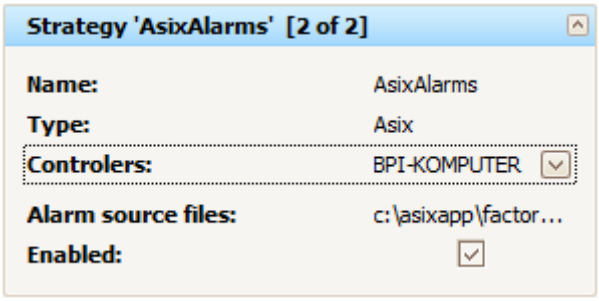


Fig. Asix Strategy.

The Asix strategy is a special strategy, which allows taking over the alarms from the Asix application. It is used in the configurations in which the Asix.Evo application serves as the browser version of Asix application.

The first step in strategy parameterization process is to configure the Asix application to create an alarm export file. It may be done by means of the Architect program of the Asix package, in the SQL type alarm archive configuration tab. This is shown on the following picture:

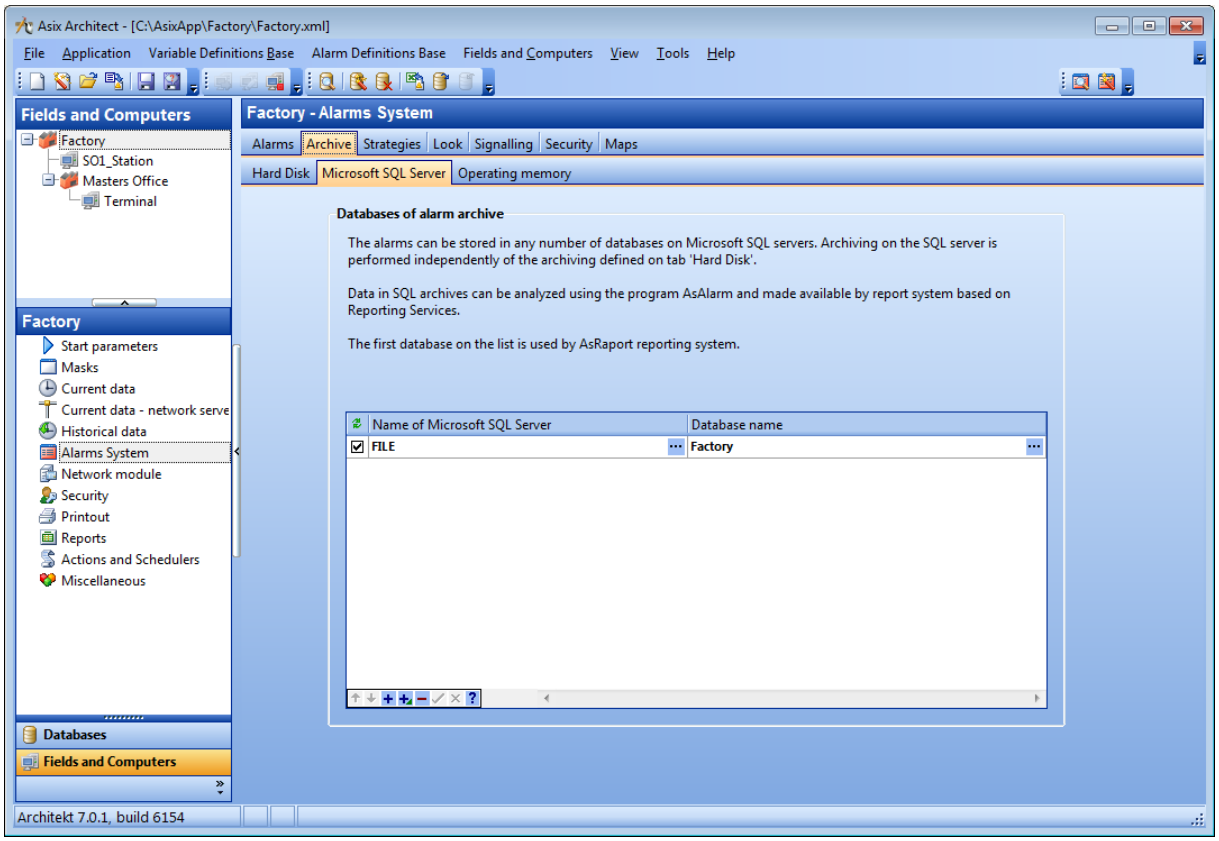


Fig. Configuration of the Asix Application to Create an Alarm Export File.

It is important to specify name *FILE* in the *Microsoft SQL server name* column.

The Asix strategy has one additional parameter named *Alarm source files*. It should specify the path to the file in which the Asix application stores the alarm events. When connecting to the application with redundant alarm servers it is necessary to specify paths to export files of all the servers. The export file is created in the alarm log directory of the Asix application. In the case shown in the picture the correct name is:

C:\AsixApp\Fabryka\Alarms\FILE_Factory.xml

An additional requirement is a correspondence between the alarm number in the Asix system and the alarm ID in Asix.Evo, e.g. the notification of alarm no. 100 in Asix causes the notification of alarm ID *100* in Asix.Evo.

5.5. External Strategy

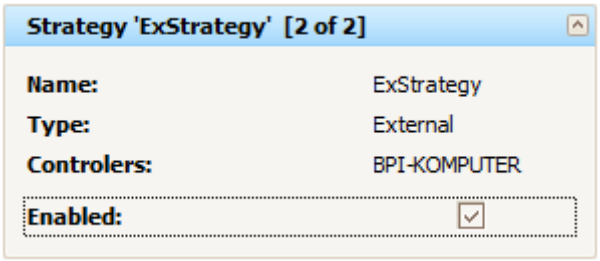


Fig. External Strategy.

The external strategy is designed to handle alarms, the changes of which are reported using the *StartAlarm* and *EndAlarm* operator actions or using scripts, and *Cancel* and *Raise* functions of the *IAlarm* interface. The strategy itself features no alarm detection methods. It serves as a container for alarms reported with external methods. Despite the fact that the strategy itself does not report alarms, subject to normal management by the active controller. The strategy is active at only single workstation at the given moment - this means that if a script running on different workstations repeatedly raise the same alarm, this alarm will be accepted only on the single workstation (on which the strategy is running).

5.6. External Global Strategy

The external global strategy works in a similar way to the normal external strategy. The global strategy is not defined anywhere. It is always created. The alarms, which in its definition has no strategy name specified are automatically allocated to a default external strategy.

The only difference in the functioning when compared to the normal external strategy is the lack of activity control. The global strategy is always active - the notification of alarm belonging to it is always accepted. The application designer must ensure that the alarm notifications are not duplicated on different workstations.

6 Active Alarm Log

The active alarm log keeps information of all currently activated alarms. Depending on configuration, it can also store information about alarms recently terminated. The alarms stored in the log are shown in the active alarm tables.

The operation mode of active alarm log is parameterized in the section of *Settings* tab shown below.

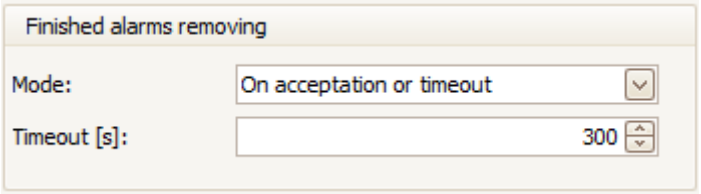


Fig. Parameterization of the Operation Mode of Active Alarm Log.

The available operation modes are described in the table below.

Tab. The available operation modes of Active Alarm Log.

Mode	**Method of operation**
Immediately	The alarm is removed from the log immediately after its termination, regardless of any other conditions. It should be noted that in this mode, alarms of *Start only* category will never be stored in the active alarm log, as they are considered as immediately terminated.
On acceptation	The terminated alarm is deleted only when confirmed by the operator.
On acceptation or timeout	The terminated alarm is deleted only when confirmed by the operator or when the time specified in the *Timeout* parameter has elapsed.

Note:

Regardless of the mode used to remove the alarms terminated, the active alarm table objects may introduce their own buffering (*Remove finished alarms* property), which results in displaying the alarm already deleted from the active alarm log. However, it is necessary for the table to be open when the alarm was included in the active alarm log.

7 Historical Alarm Log

The historical alarm log stores information on all alarm events, both the alarm beginning and end. The log may keep an event history for many days.

The operation mode of historical alarm log is parameterized in the section of *Settings* tab shown below.

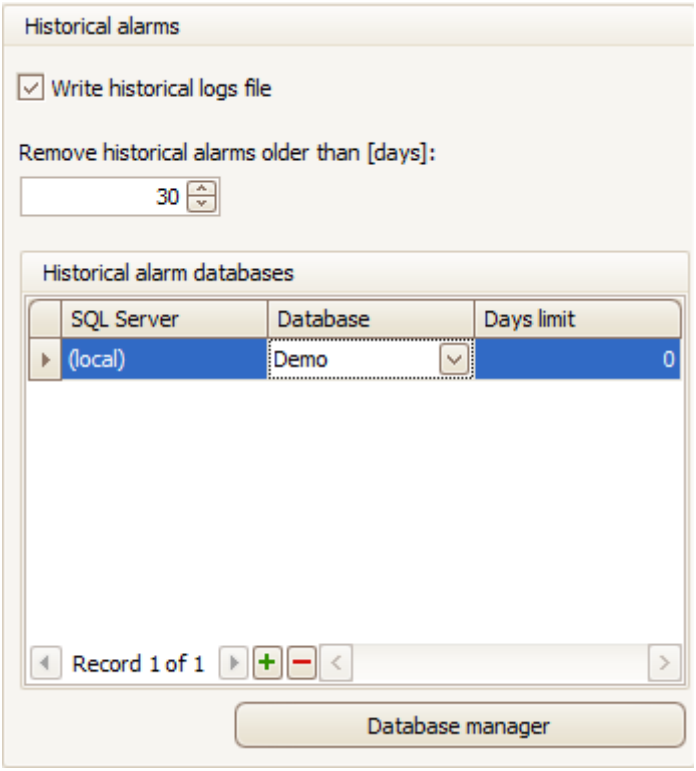


Fig. Parameterization of the Operation Mode of Historical Alarm Log.

Basic alarm log, used for the current alarm handling is created at each domain workstation. The contents of the historical logs of the respective workstations are mutually synchronised. By default, the log is stored in a local file; in the *Remove historical alarms older than [days]* parameter it may be specified how long the alarm event information is to be kept. If the option *Write historical logs file* is not checked, the historical log is stored only in the memory when the application is running. After restarting the application, the log will be created from scratch.

7.1. SQL Alarm Archive

The standard historical alarm log is completed with SQL type alarm archive. The archive is used in the alarm analysis performed with the AsAlarm program and in the reporting systems. In order to improve the reliability, it is possible to create archives on multiple servers simultaneously. But even if

a single server is used, the Asix.Evo application provides buffering mechanisms that allow for operation when the connection with SQL Server is lost for a while. The SQL archive should be parameterized identically at each controller workstation. This follows from the fact that a record to the SQL database is executed only by the active controller workstation.

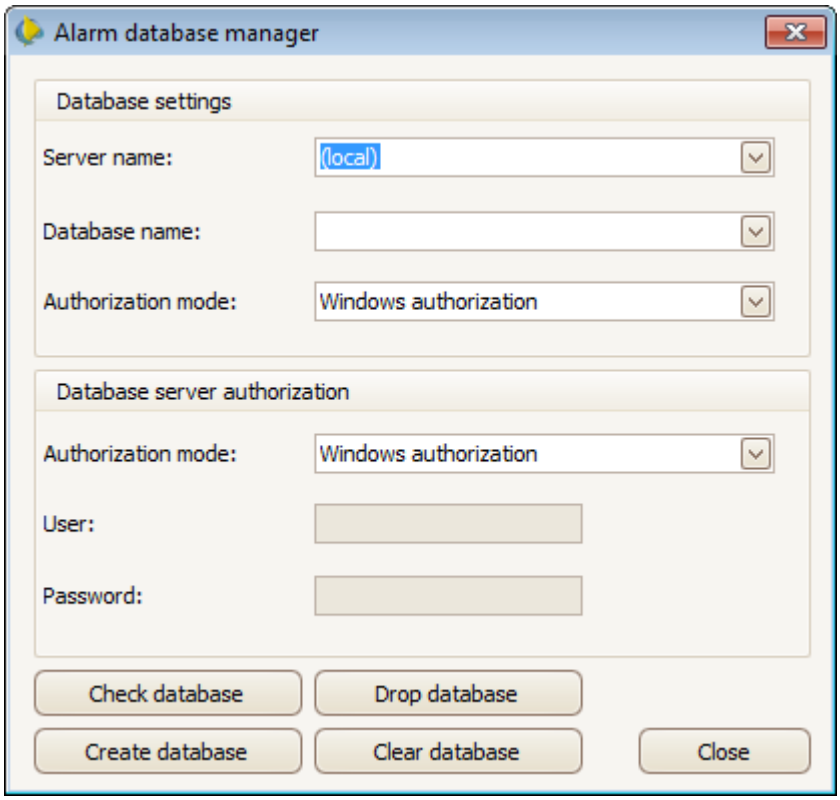


Fig. Alarm Database Manager.

The alarm SQL database must be created prior to start archiving. This is done with the window shown above, opened with the *Database manager* button. The window also allows performing other administrative tasks. The credentials of the *Database server authorization* group must provide such permissions in SQL server which allow performing database creation and deletion operations.

8 Alarm State Visualization

The basic elements used to visualize the alarm state are an alarm table objects. There are two types of tables, designed to handle active and historical alarms. Other universal objects may be used as a supplementary for alarm state presentation.

8.1. Active Alarm Tables

The *Active Alarms Viewer* object displays information on alarms stored in the active alarm log.

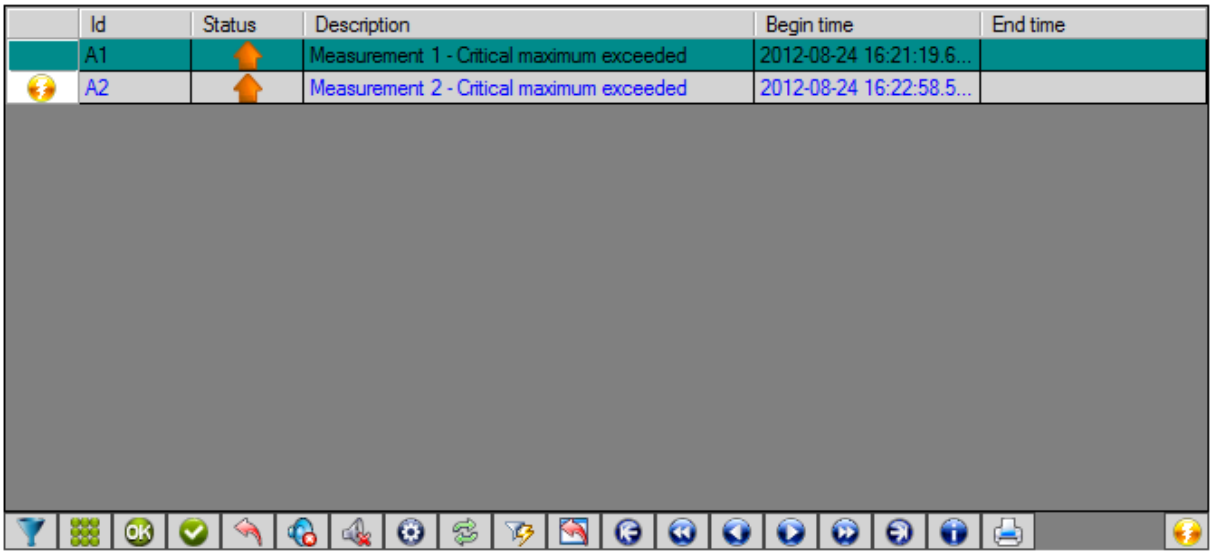


Fig. The Active Alarms Viewer Object.

The object can be also operated by the operator. The buttons on the integrated toolbar are used for this purpose.

8.2. Historical Alarm Tables

The *Historical Alarms Viewer* object displays information on alarms stored in the historical alarm log.

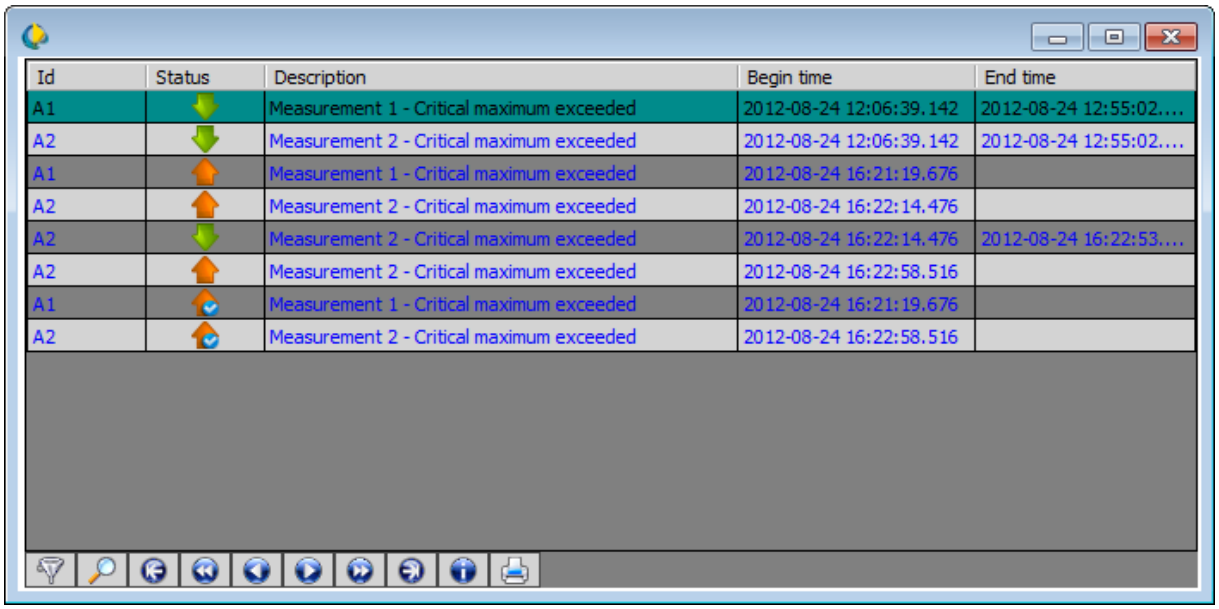


Fig. The Historical Alarms Viewer Object.

The object features mechanisms for easy determination of alarm set shown in the table.

8.3. Other Objects

Alarm status may be shown on standard objects. To do this, use the *IsAlarm*, *IsAlarmUnaccepted* or *IsAlarmExcluded* function. All these functions return information about the status of alarm selected.

The following example shows the parameterization of *Text* object displaying the value of *VA001* variable. If the alarm of *A_VA001* ID is active, the variable value is displayed in red.

Basic Properties		
	Main variable	VA001
State properties, Basic Properties		
	Text	#
	Color	Black
State properties, State {1}		
	State Condition	**=IsAlarm(A_VA001)**
	Color	Red

9 Alarm Management

Most of the alarm management function available for the operator may be accessed through the interface of the *Active Alarms Viewer* object. Some functions can be executed using the operator actions. This allows the application dsigner to integrate maintenance functions directly with the synoptic diagram objects.

9.1. Alarm Acknowledgement

The basic alarm acknowledgement mechanism is provided by the button localised in the toolbar of *Active Alarms Viewer* object. Pressing this button confirms all visible alarms. Only alarms of urgent and critical priorities require prior selection of the alarm row. An alternative acknowledgement method is to execute the *AcceptAlarm* operation used in the diagram object event handling.

Alarm acknowledgment function is controlled by a authorization system. The first condition is authorization to acknowledge alarms on the workstation. It is declared in the workstation configuration in the alarm system parameterization panel. The second condition relates to individual permissions of the currently logged user. The user must play a role that has *Alarms acceptation right* permission.

9.2. Alarm Exclusions

Alarm exclusion mechanism provides possibility to exclude any alarm from handling. All events relating to the alarm excluded are ignored. Exclusions are useful in order to eliminate alarms, which, because of some failure, are not detected correctly.

The exclusion management is done via the window below which is opened by the button from the *Active Alarms Viewer* object toolbar.

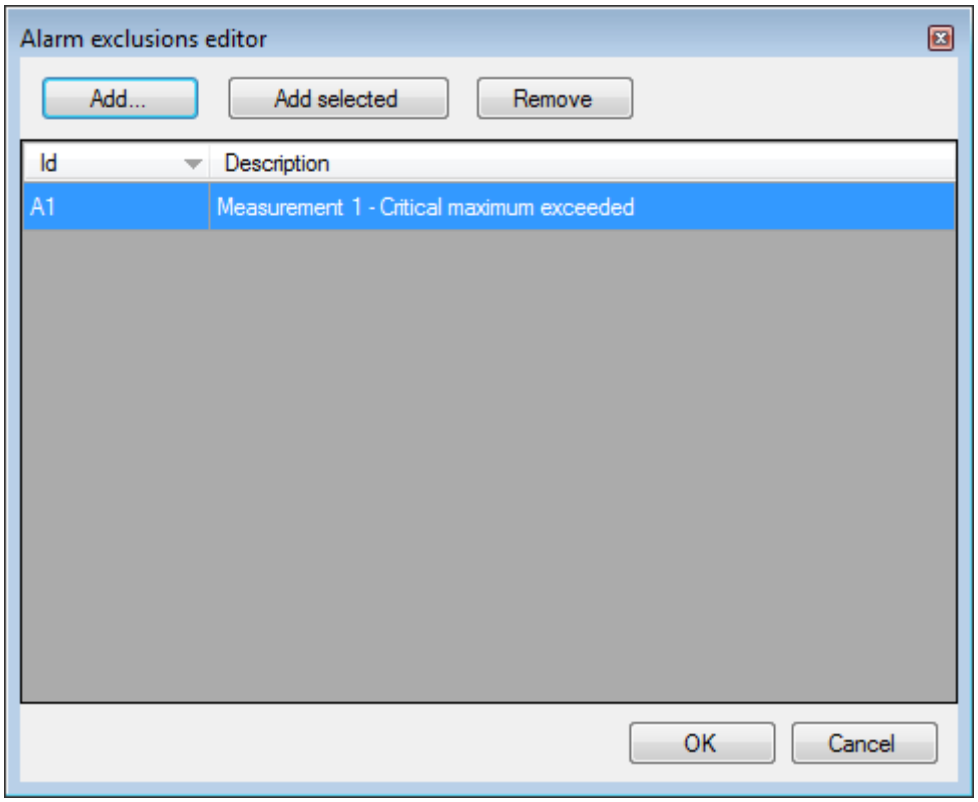


Fig. Alarm Exclusions Editor.

The list shows all alarms that are currently excluded.

Alarm exclusion function is controlled by the authorization system. The currently logged user must perform a role that has *Alarm exclusions edit right* permission.

9.3. One-time Exclusions

One-time exclusion mechanism allows the operator to remove from the table any active alarm. In contrast to standard exclusions, next detection of the alarm will be reported as normal.

One-time exclusion is performed by selection of the alarm in the active alarm table and pressing the appropriate toolbar button.

One-time alarm exclusion function is controlled by the authorization system. The currently logged user must perform a role that has *Alarm exclusion once right* permission.

9.4. Alarm Filtering

Alarm filtering mechanism allows restricting the number of alarms notified. The mechanism eliminates short-term alarm events that can result from malfunction of the sensors.

There are two modes of filtering:

- *Short alarms elimination*

The short-term changes in alarm statuses are ignored. The status change will be accepted if the new state maintain for a predetermined time.

- *Fast sentences elimination*

The sequences of short-term status changes ("vibrating contacts") are ignored. First, even short-term change is immediately accepted, but the next will be accepted only after a waiting period.

The filter management is done via the window below which is opened by the button from the *Active Alarms Viewer* object toolbar.

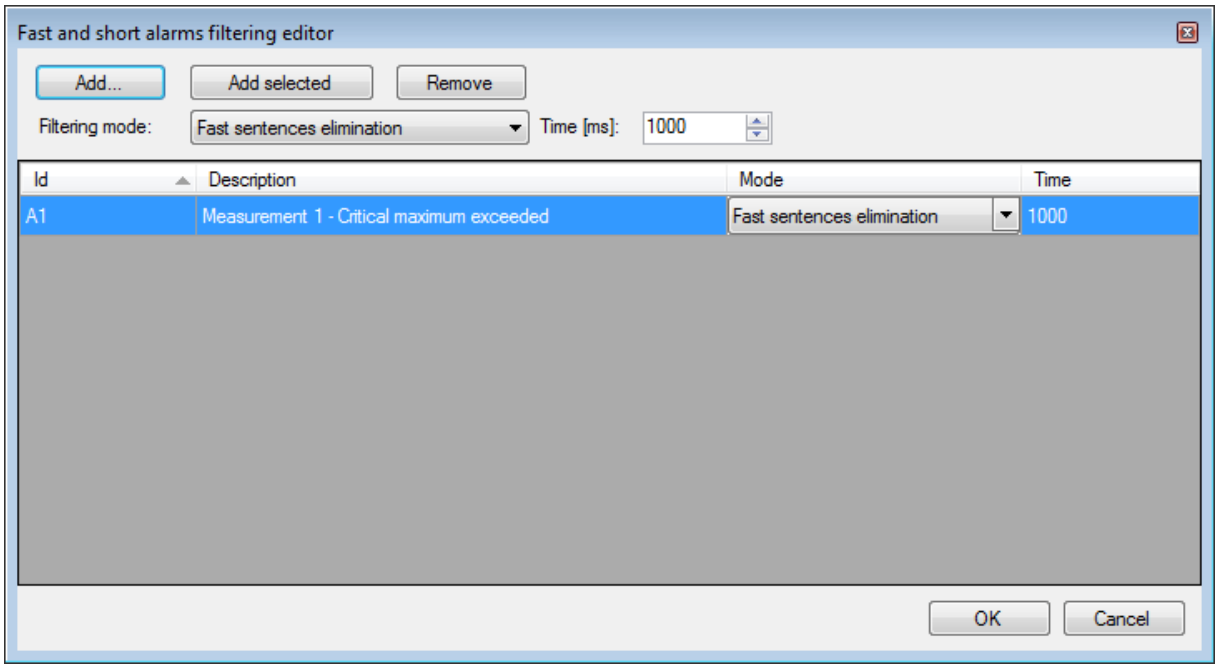


Fig. Fast and Short Alarms Filtering Editor.

To enable the filter: choose the alarm by the *Add...* button, select the filter mode and set the filter time.

Alarm filtering function is controlled by the authorization system. The currently logged user must perform a role that has *Alarm filters edit right* permission.

9.5. Sound Signalling

An alarm detection may be signalled by a sound signal. This happens when in the alarm definition, in the *Sound* attribute the audio file name has been specified.

The operator has the ability to mute the sound. The first method is to use the mute button on the toolbar of *Active Alarms Viewer* object. The second method is to use *StopSound* operator action.

In a case of applications using several workstations, a sound signal is played on all these workstations. The alarm table toolbar features two different mute buttons. The first one mutes sound only on the local workstation, the second does it on all domain workstations.

9.6. Printing

The alarm printing function is available in the *Active Alarms Viewer* and *Historical alarms Viewer* objects. Pressing the *Print* button will open the following window in which the operator can set print options.

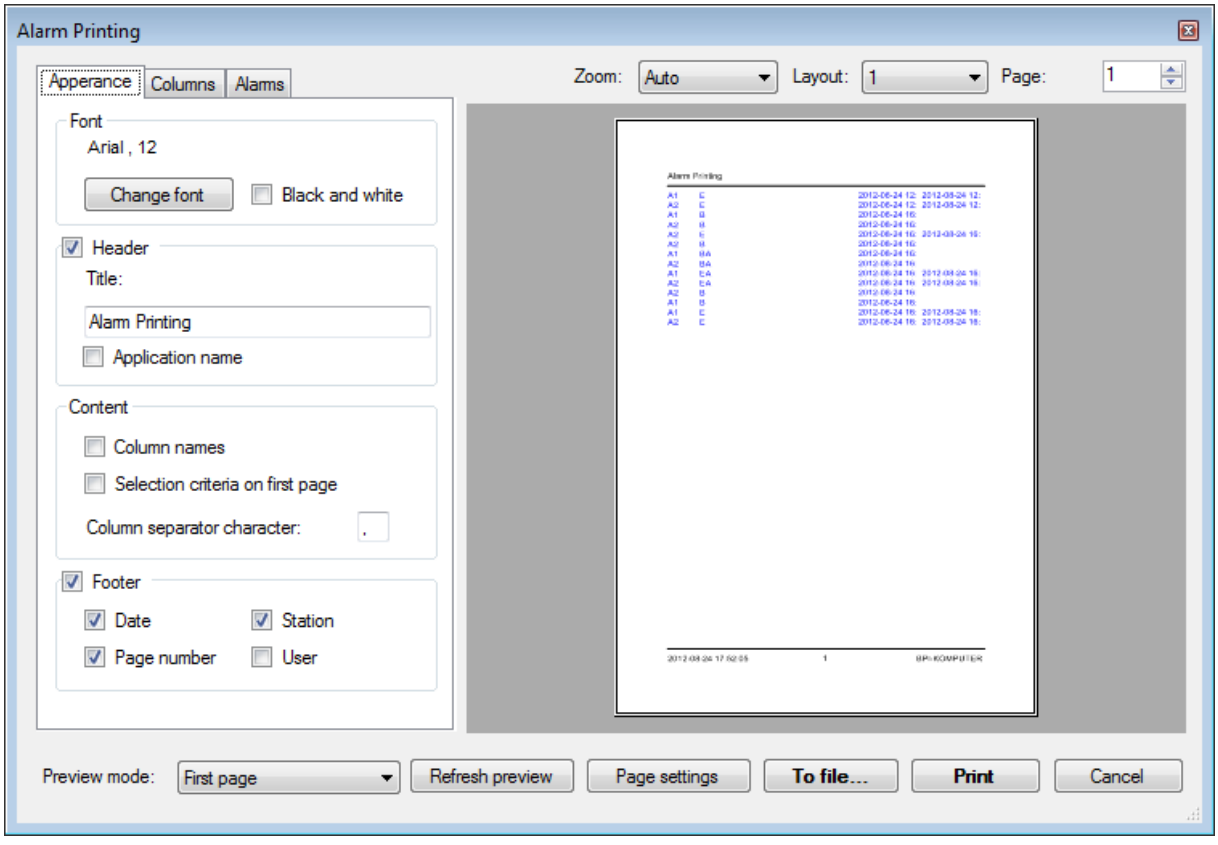


Fig. Print Option Window.

The appearance of the printout, the dataset to be printed and alarm selection criteria can be specified in the print dialogue box. In addition to printing from a printer, the alarm data may be saved to a text file.

10 Integration with AsAlert

The AsAlert program is used in the Asix system applications to send alarm notifications. Depending on the application configuration, the notification may be sent by SMS or e-mail.

The Asix.Evo alarm system may be integrated with the AsAlert program. The interconnection parameterization is performed in two steps. The first step involves basic communication parameters declaration. This can be done in the *Settings* tab section shown below.

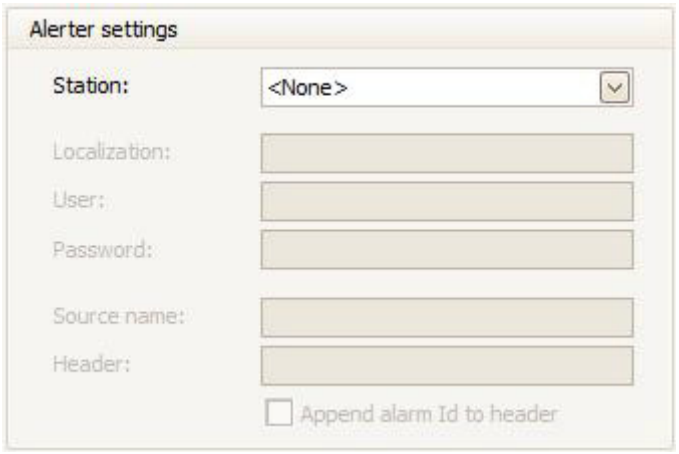


Fig. AsAlert Settings.

The parameters functions are as follows:

- *Station*

Defines the Asix.Evo application workstations, which will initiate a notification process. This may be the name of any workstation involved in the domain operation. It is possible to choose option in which the notifications are sent always by the active controller. The right choice depends on system configuration. In each case, the workstation should be active at all times. So the alerts are often sent from the controller workstations. If AsAlert is installed on the computer of one of the controllers, the notification should be send usually from this workstation. Otherwise, generally a better option is the active controller.

- *Localization*

The system name of computer on which the AsAlert program is installed. When the local installation is used, the parameter should be left blank.

- *User*

The AsAlert program username on account of which notification will be sent.

- *Password*

The AsAlert program user password on account of which notification will be sent.

- *Source name*

Any text which in the alert will be used as the message sender.

- *Header*

Any text which in the alert will be used as the message title.

The second step in the interconnection parameterization is appropriate setting of the *Alert mode* and *Alert recipients* attributes in the definition of each alarm, for which the alerts are to be sent. Alert mode enables selecting alert sending condition: on the alarm beginning or end. The names of addressees specified in the AsAlert program, to which the notification is to be sent must be specified in the *Alert recipients* attribute.