User Manual

YDOC-Messenger

Alarm Management & Dispatching

Version 5
Aug 2015

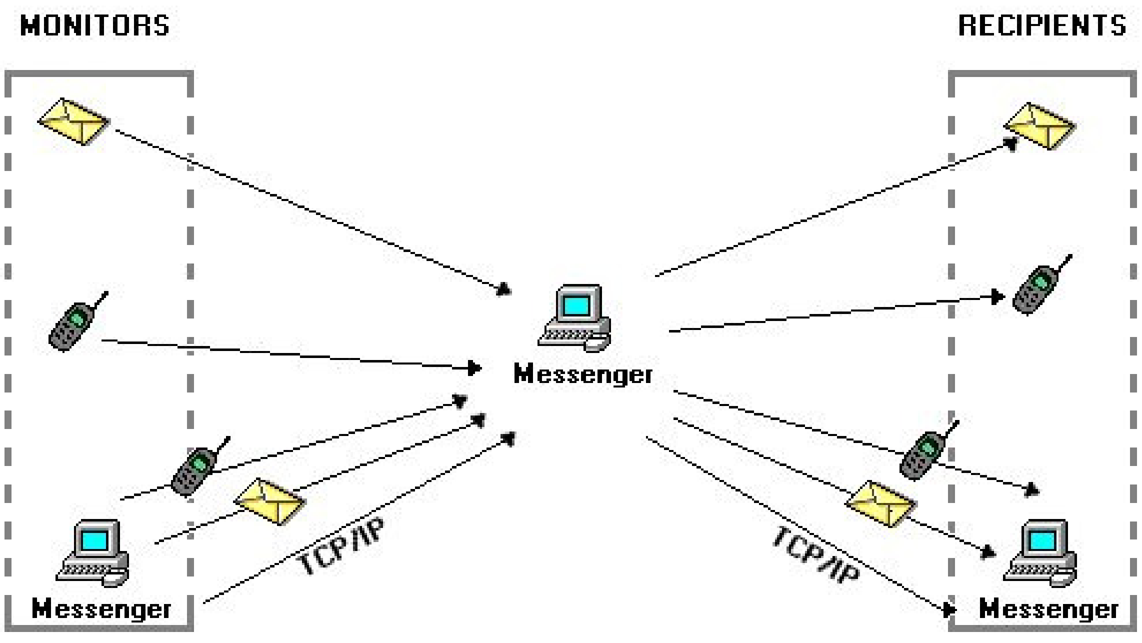
CONTENTS

1. Introduction

1.1 What is Messenger

‘YDOC-Messenger’ is an MS-Windows based alarm management & dispatching system. Redirection of alarms and messages from unattended stations towards fault clearing service or operators. Origin & content sensitive message distribution with a hierarchical fallback mechanism (incase an operator does not respond in time)

‘Messenger’ can accept and dispatch alarms through various channels. Text-based alarm messages can be received & send through a/o SMS, E-mail, TCP/IP and Serial-ports. Values can be sensed and actuators are signaled through a/o MODBUS and Serial-ports. Scripting can be used to pre-process incoming messages and re-format outgoing messages, which makes ‘Messenger’ extremely flexible for interfacing with custom systems.



1.2 Features

1. Origin & content sensitive message dispatching with hierarchical fall-back mechanism.
2. Accepting text messages through various channels (a/o SMS, E-Mail & TCP/IP)
3. Dispatching text messages through various channels (a/o SMS, E-Mail & TCP/IP)
4. Sensing values through various channels (a/o MODBUS & Serial ports)
5. Signaling actuators through various channels (a/o MODBUS & Serial ports)
6. Per alarm transmitter configurable 'heart beat' interval
7. Per alarm transmitter configurable monitoring schedule
8. Per operator configurable fault-clearing schedule
9. Archiving of incoming & outgoing alarm messages
10. Pre-process incoming messages with JScript
11. Re-format outgoing messages with JScript

1.3 Minimum System Requirements

1. Microsoft-Windows 2012, W7, W8 or W10
2. XGA monitor (1024x768)
3. Network adapter (TCP/IP)
4. Optional for exchanging SMS messages: GSM Modem (AT protocol V.25ter, GSM 07.07 en GSM 07.05)

2. System Description

2.1 Introduction

The base of any ‘Messenger’ system consists of minimal one MS-Windows compatible personal computer running the ‘‘Messenger’’ software application. The system is intended to gather and distribute messages regarding informative, alarm or erroneous situations of unattended objects/processes. When at some point alarms or messages are produced and send to the ‘Messenger’, the ‘Messenger’ will dispatch the messages to entities capable to deal with that particular erroneous situation. The ‘Messenger’ uses a hierarchical fallback structure, to avoid unhandled erroneous situations, in case entities are not capable to respond or re-act.

A ‘Messenger’ system consists out of three functional parts.

- Message sources (Monitors)
- Message destinations (Recipients)
- Message dispatching

Message sources

A message source is an Error, Warning or Hint message generating object/process. In example an external alarm module sending SMS or E-mail messages to ‘Messenger’ or an application passing alarm messages to the ‘Messenger’ through the API (Application Programmers Interface, which makes use of “named pipes”).

Message destinations

A message destination is an entity/object able to receive messages. In example the mobile phone of an engineer who has the skills to take a corrective action, it can also be another ‘‘Messenger’’ for further dispatching.

Message dispatching

The ‘‘Messenger’’ incorporates an advanced dispatching algorithm, which enables you to dispatch received messages to an entity with the right capabilities to take the necessary corrective actions. The dispatcher takes care of rosters avoiding that multiple entities are taking care of the same situation simultaneously. If an alarm recipient does not respond in time, the alarm will be dispatched to the next recipient in the hierarchical list.

2.2 Messages

A “Messenger” Message contains (beside the message text it self) additional information a/o for use by the dispatching mechanism.

Origin

This is a text describing the ‘Origin’ of the message. If a monitor sub-divides messages in sub-levels. The ‘origin’ is a text of concatenated level names, separated by each other with slashes (/). The first level name is always the concerning monitor. When configuring the dispatcher, recipients can be bound to any (sub) level. For instance to distinguish between messages meant for electrical engineers and messages meant for mechanical engineers.

Example: *MONITOR1/PUMPSYSTEM/ELECSYSTEM*

Type (Severity)

The following 4 types of messages are possible: Error, Warning, Hint and Heartbeat.

‘Errors’ and ‘Warnings’ are treated equally and differ from each other by color and name only (Error = Red, Warning = Purple). A warning will mostly be used to inform that a system is reaching a critical situation, while an error is used to indicate that the system is in a critical situation. In case of an alarm (an Error or Warning) the ‘Messenger’ will dispatch an alarm to the first recipient in the list of applicable recipients (recipients assigned to the deepest matching level occur in the list first). If the recipient does not respond in time, an alarm will be dispatched to the next recipient in the list.

A ‘Hint’ is an informative message and the ‘Messenger’ will only dispatch a ‘Hint’ to scheduled recipients assigned to the (sub) level matching to the level of the message. A recipient cannot and does not have to respond to ‘Hint’ messages.

A ‘Heartbeat’ message is an invisible message and is used to signal the “Messenger” that a certain object/process is still alive. If a ‘Heartbeat’ is not received in time, the “Messenger” will raise a ‘Missed heartbeat’ message of type ‘Error’ and with category ‘Support’.

Category

The following 4 categories of messages are possible: Operator, Support, R&D and Manager.

When configuring the hierarchical dispatching structure, a recipient can be bound to a list to respond to messages of all categories or to a list to respond to just a specific category. For instance to distinguish between messages meant for operators and support engineers. E.g. an ‘Operator’ message could be a message to instruct an operator to refill a storage tank, while a ‘Support’ message could be an alarm to notify an engineer about a mechanical problem.

2.3 Responding to alarms

If an alarm is send to a person instead of another Messenger or automated system, the alarm is supposed to be a readable text and will normally contain an unique message sequence number. The person can (if it is possible with communication medium) respond to that alarm by replying a message starting with or containing the unique message sequence number, directly followed by a response code.

The following 3 response codes are possible:

1. **ACT** or **Action** indicates that the replying person is taking action to correct the erroneous situation.
2. **SOL** or **Solved** indicates that the replying person has resolved the erroneous situation.
3. **REJ** or **Rejected** indicates that the replying person is not capable to take the necessary corrective actions. After a ‘rejection’, the ‘Messenger’ will dispatch the alarm to the next available recipient immediately, avoiding unnecessary delays.

Example of an SMS alarm:

```
2007/03/20 11:44:11, 21216 OPERATOR WARNING, LEVEL TANK 1 TO LOW
```

Example of an SMS response:

```
21216 ACT
```

Note: Response codes are case insensitive.

3. Message Display

3.1 Introduction

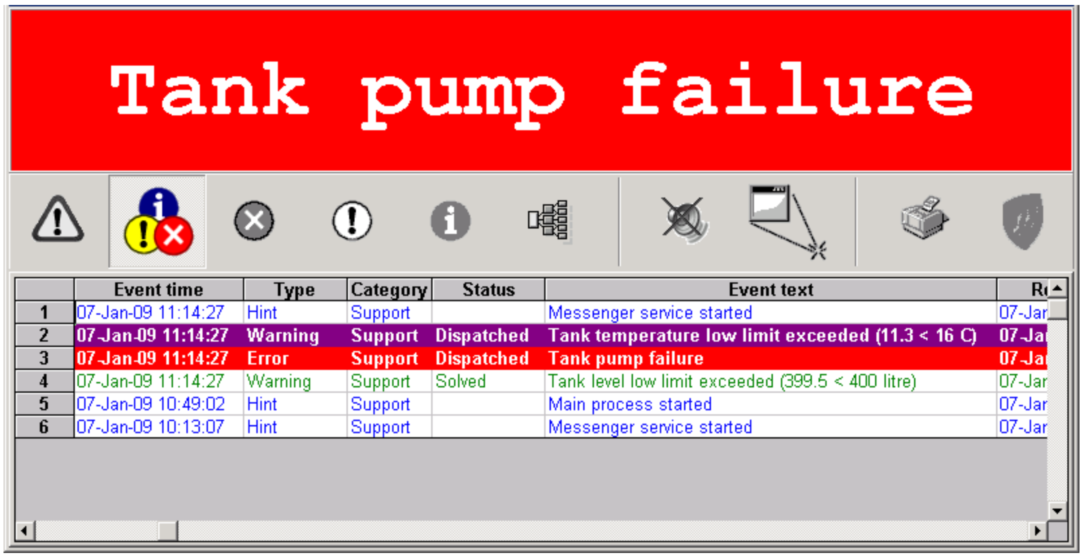
The “Message Display” shows all active alarms and follow-up. Solved alarms and hints are shown as well, but will be removed from the display if the list grows to big or older than seven days. To consult the full alarm history, please start the “History Viewer” application.

Note: The “Message Display” does not have to be active, for “Messenger” to do its job. The “Messenger Service”, which can be installed with the “Properties Editor” application, will take care of the monitoring & dispatching of alarms in the background and will start running as soon as “Windows” starts running (its a System Service).

Quick status panel ->

Tool bar ->

Message pane ->



Tank pump failure

	Event time	Type	Category	Status	Event text	R
1	07-Jan-09 11:14:27	Hint	Support		Messenger service started	07-Jar
2	07-Jan-09 11:14:27	Warning	Support	Dispatched	Tank temperature low limit exceeded (11.3 < 16 C)	07-Ja
3	07-Jan-09 11:14:27	Error	Support	Dispatched	Tank pump failure	07-Ja
4	07-Jan-09 11:14:27	Warning	Support	Solved	Tank level low limit exceeded (399.5 < 400 litre)	07-Jar
5	07-Jan-09 10:49:02	Hint	Support		Main process started	07-Jar
6	07-Jan-09 10:13:07	Hint	Support		Messenger service started	07-Jar

3.2 Quick status panel

This panel has a big colored surface, indicating the most severe alarm situation. So you can see the status in a glance from a distance. The following 5 states, in order of priority, are possible:

1. Red-surface indicates that at least one not yet acknowledged “Error” situation exists.
2. Purple-surface indicates that at least one not yet acknowledged “Warning” situation exists.
3. Red-font indicates that at least one acknowledged “Error” situation exists.
4. Purple-font indicates that at least one acknowledged “Warning situation exists.
5. Green-surface indicates that all alarm situations are resolved.

3.2 Tool bar

- *All active Errors & Warnings*

  Click this button to show a list with only the active alarms, hints and resolved alarms will be hidden.

- *All messages*

  Click this button to show a list with all the received messages, hints, warnings and errors.

- *Errors*

  Click this button to show a list with all the received errors only.

- *Warnings*

  Click this button to show a list with all the received warnings only.

- *Hints*

  Click this button to show a list with all the received hints only.

- *Dispatching*

  Click this button show the dispatching status of all messages. It shows to which recipients messages are send to and which recipients have acknowledged, rejected an alarm or have solved an erroneous situation.

- *Stop Sounds*

  Sounds can be played on new incoming messages. Click this button to Stop playing sound files of all currently active alarms.

-- *Shrink*

  Shrinks the 'Display' to a small (150x300) "Quick Status panel". Double clicking the "Quick Status panel" restores the "Display" back to its original size.

- *Print*

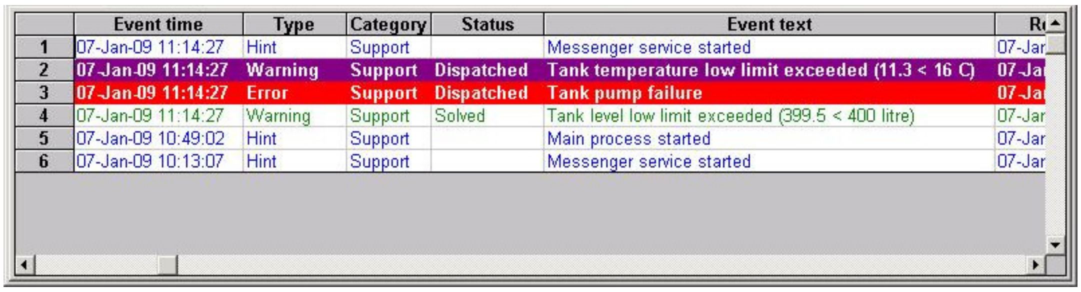
  Prints the selected message list.

- *About*

  Displays the 'About'-box.

3.4 Message pane

This pane displays all received messages. They may originate from external alarm transmitters, other ''Messenger's', the local ''Messenger'', or other applications using the API. Messages with more recent event times appear earlier in the list then messages with later event times.



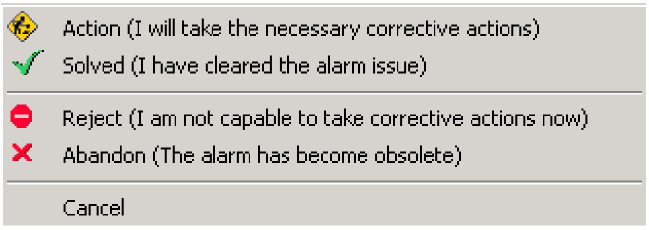
	Event time	Type	Category	Status	Event text	R
1	07-Jan-09 11:14:27	Hint	Support		Messenger service started	07-Jar
2	07-Jan-09 11:14:27	Warning	Support	Dispatched	Tank temperature low limit exceeded (11.3 < 16 C)	07-Ja
3	07-Jan-09 11:14:27	Error	Support	Dispatched	Tank pump failure	07-Ja
4	07-Jan-09 11:14:27	Warning	Support	Solved	Tank level low limit exceeded (399.5 < 400 litre)	07-Jar
5	07-Jan-09 10:49:02	Hint	Support		Main process started	07-Jar
6	07-Jan-09 10:13:07	Hint	Support		Messenger service started	07-Jar

Hints are displayed in blue on a white background. **Errors** are displayed in white on a red background. **Warnings** are displayed in white on a purple background. When an Alarm is acknowledge the text and background color will flip, so **acknowledged errors** are displayed in red on a white background and **acknowledge warnings** are displayed in purple on a white background. A **solved alarm** will be displayed in green.

When you right-click a single message or a selected area, a pop-up menu as shown below will appear.

You can select multiple messages at once by defining an area by left clicking a message and moving down the mouse while holding the left button.

The options shown in the pop-up menu are depending on the type and status of the selected message(s).


Action (I will take the necessary corrective actions)
Solved (I have cleared the alarm issue)
Reject (I am not capable to take corrective actions now)
Abandon (The alarm has become obsolete)
Cancel

Action

Select this option to **acknowledge** that you or someone else is going to undertake the necessary corrective actions to resolve the alarm situation. When a message is acknowledged, further dispatching will be stopped. Recipients already dispatched to will be informed about the status change.

Solved

Select this option to indicate that the alarm situation has been resolved. Recipients dispatched to will be informed about the status change.

Reject

Select this option to indicate that you are not or no longer capable to take the necessary corrective actions. Recipients dispatched to will be informed about the status change and further dispatching will be continued.

Abandon

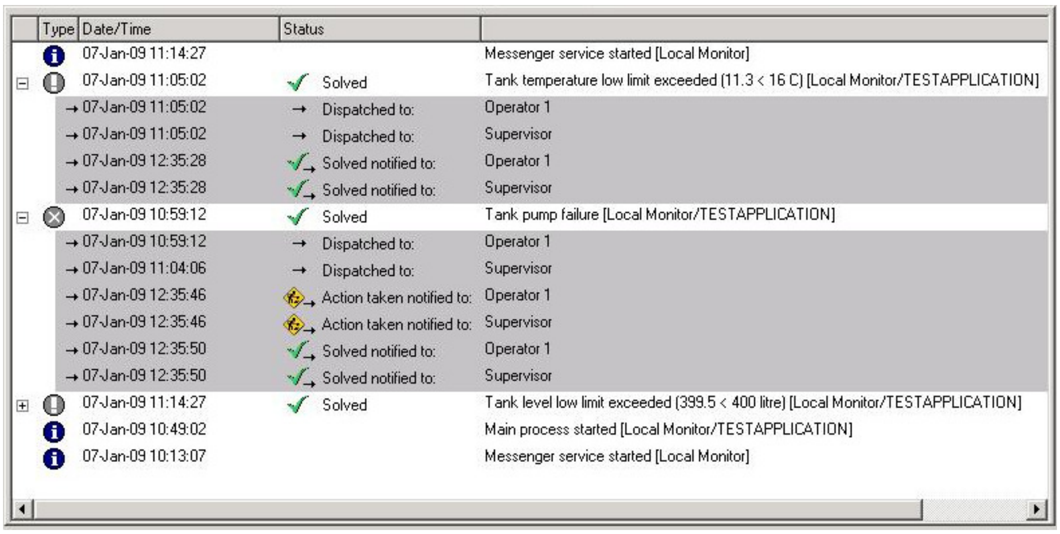
Select this option to abandon the message. Please note that you CANNOT recover from an abandoned message this option must be used only to abandon messages that are of no importance anymore.

Cancel

Select this option to close the popup menu.

3.5 Dispatching pane

This pane shows the dispatching status of all messages. It shows to which recipients messages are send to and which recipients have acknowledged, rejected an alarm or have solved an erroneous situation.



	Type	Date/Time	Status	
	ⓘ	07-Jan-09 11:14:27		Messenger service started [Local Monitor]
⊟	!	07-Jan-09 11:05:02	✓ Solved	Tank temperature low limit exceeded (11.3 < 16 C) [Local Monitor/TESTAPPLICATION]
		→ 07-Jan-09 11:05:02	→ Dispatched to:	Operator 1
		→ 07-Jan-09 11:05:02	→ Dispatched to:	Supervisor
		→ 07-Jan-09 12:35:28	✓ Solved notified to:	Operator 1
		→ 07-Jan-09 12:35:28	✓ Solved notified to:	Supervisor
⊟	⊗	07-Jan-09 10:59:12	✓ Solved	Tank pump failure [Local Monitor/TESTAPPLICATION]
		→ 07-Jan-09 10:59:12	→ Dispatched to:	Operator 1
		→ 07-Jan-09 11:04:06	→ Dispatched to:	Supervisor
		→ 07-Jan-09 12:35:46	Action taken notified to:	Operator 1
		→ 07-Jan-09 12:35:46	Action taken notified to:	Supervisor
		→ 07-Jan-09 12:35:50	✓ Solved notified to:	Operator 1
		→ 07-Jan-09 12:35:50	✓ Solved notified to:	Supervisor
⊞	!	07-Jan-09 11:14:27	✓ Solved	Tank level low limit exceeded (399.5 < 400 litre) [Local Monitor/TESTAPPLICATION]
	ⓘ	07-Jan-09 10:49:02		Main process started [Local Monitor/TESTAPPLICATION]
	ⓘ	07-Jan-09 10:13:07		Messenger service started [Local Monitor]

There is a line in the pane for each individual message. Messages with more recent event times appear earlier in the list then messages with later event times.

There are two icons in front of a message line:

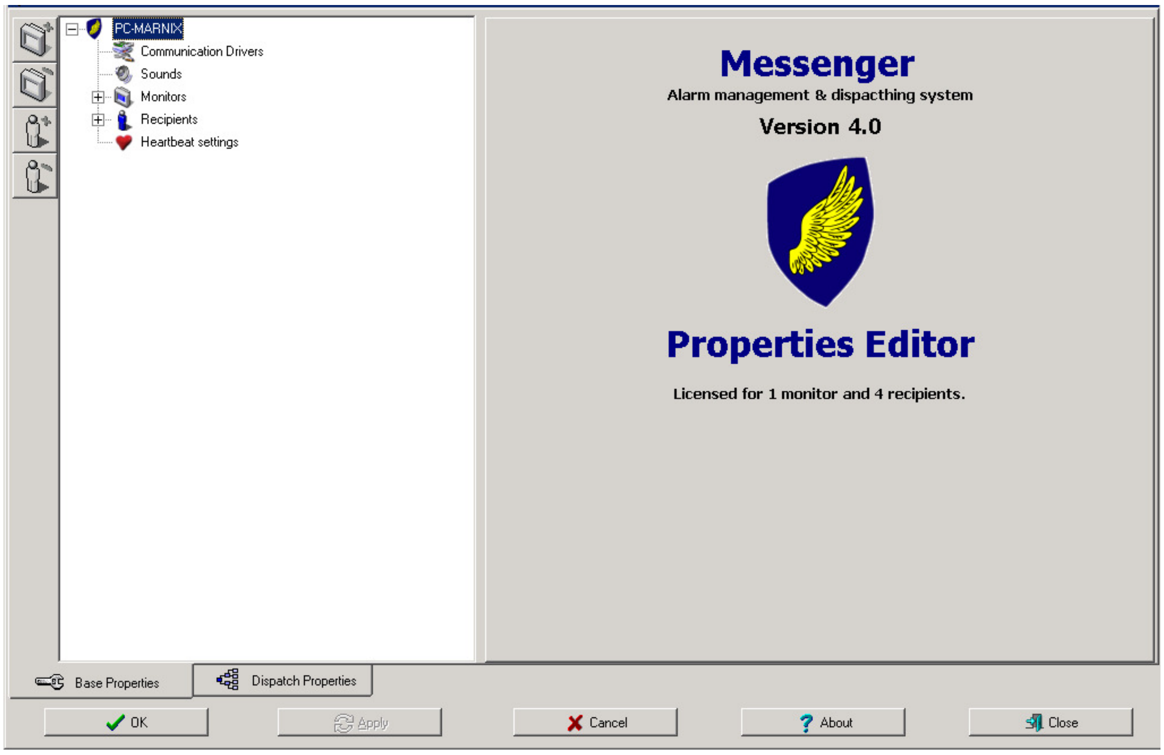
1) '+'-Icon. Click this icon to expand the sub-lines of a message. The sub-lines are showing the follow-up messages. It shows to which recipients messages are send to and which recipients have acknowledged, rejected an alarm or have solved an erroneous situation.

2) 'Type'-Icon. If an alarm is still active, this icon will be colored and if an alarm has been resolved or cancelled, this icon will be grayed.

4. Properties Editor

4.1 Introduction

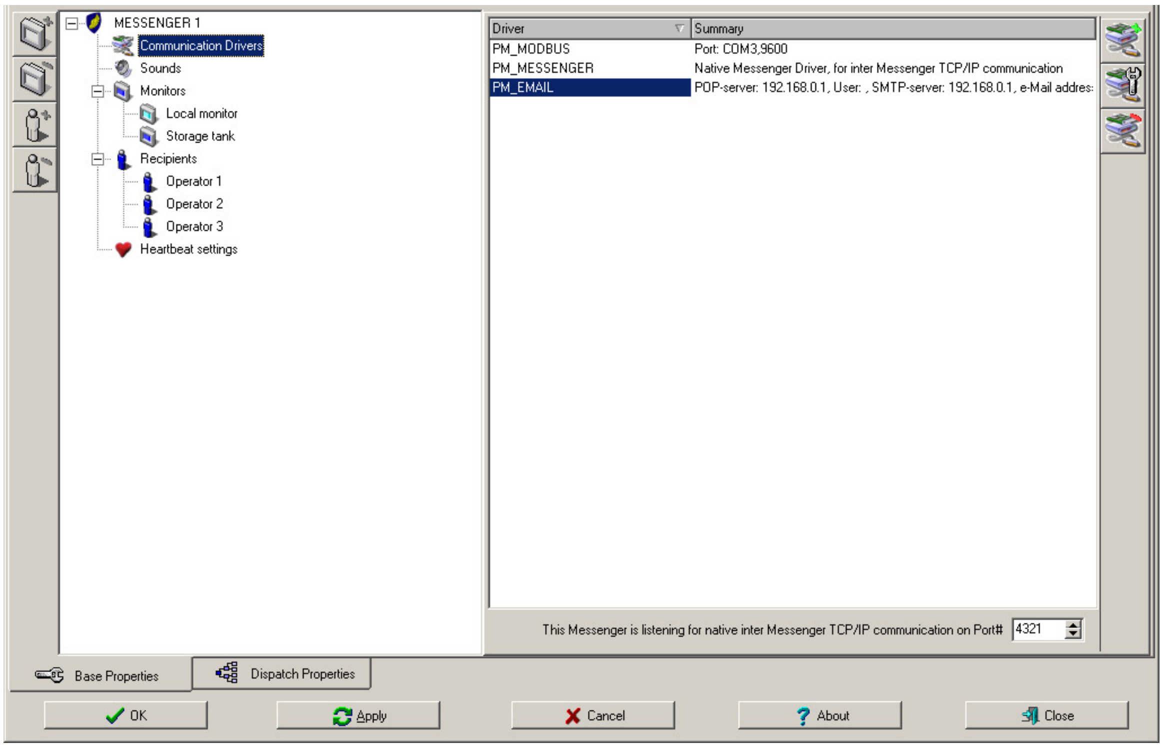
The ‘Properties Editor’ is a separated application that is used to configure the behavior of the ‘Messenger’. You can configure drivers to interface with message generating sources and message receiving destinations. You can define ‘Monitors’ monitoring a message-generating source. You can define message-receiving destinations, referred to as recipients. You can define schedules to avoid dispatching messages to recipients not on duty. You can define a hierarchical dispatching structure to dispatch messages to the best matching recipients first.



4.2 Communication Drivers

You can configure drivers to interface with message generating sources and message receiving destinations. In example an e-Mail driver to exchange messages through e-Mail. Please consult Appendix ‘Communication Drivers’ for a list and description of available drivers.

The pre-installed default driver is ‘PM_MESSENGER’, which can be used for exchange of messages with other ‘Messengers’. If required you can place multiple ‘Messengers’ in cascade.



Driver	Summary
PM_MODBUS	Port: COM3,9600
PM_MESSENGER	Native Messenger Driver, for inter Messenger TCP/IP communication
PM_EMAIL	POP-server: 192.168.0.1, User: , SMTP-server: 192.168.0.1, e-Mail addres

This Messenger is listening for native inter Messenger TCP/IP communication on Port# 4321

Click the ‘Add’-button to add a new driver and pop-up a list with possible drivers. After selecting a driver, a driver specific configuration window will appear. In example to define the SMTP- and POP-server settings for e-Mail communication.

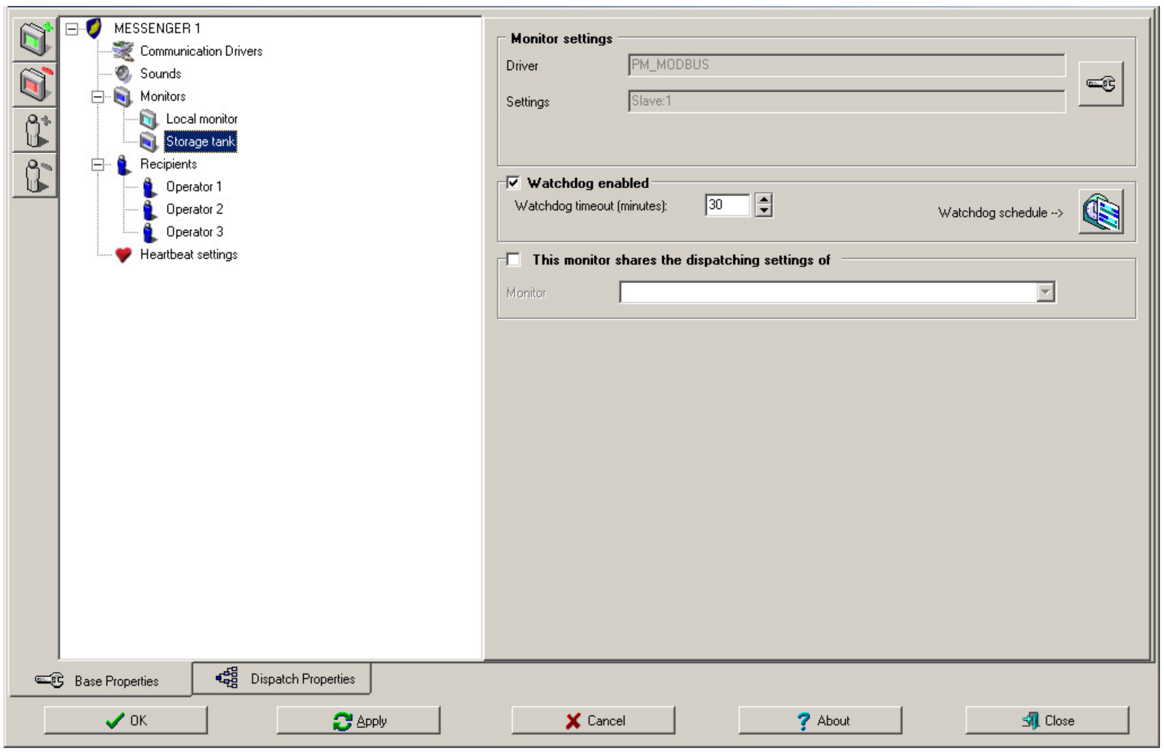
Click the ‘Edit’-button to edit the properties of a driver.

Click the ‘Delete’-button to delete a driver that is no longer of use. You can’t delete a driver where you have defined ‘Monitors’ and/or ‘Recipients’ for.

The ‘Messenger’ has a build in TCP/IP server listening for incoming messages send from other Messengers using the PM_MESSENGER driver. The default TCP-port the ‘Messenger’ is listening at is 4321. If your network firewall blocks this port#, change your firewall settings or choose a non-blocked port#.

4.3 Monitors

Monitors are meant to monitor message-generating sources. A message-generating source can be an external alarm transmitter sending text-based messages through a/o SMS, E-mail or TCP/IP. It can be another ‘Messenger. It can be a Data-acquisition-systems providing sensor values through a/o MODBUS or Serial port.


MESSENGER 1
- Communication Drivers
- Sounds
- Monitors
  - Local monitor
  - Storage tank
- Recipients
  - Operator 1
  - Operator 2
  - Operator 3
- Heartbeat settings

Monitor settings

Driver: PM_MODBUS

Settings: Slave:1

☑ **Watchdog enabled**

Watchdog timeout (minutes): 30

Watchdog schedule -->

☐ **This monitor shares the dispatching settings of**

Monitor:

Click the ‘Add’-button to add a new monitor, select the appropriate communication driver and configure the driver specific properties. In example an e-mail address of the message generating sources where you expect to receive e-mails from.

Click the ‘Edit’-button to edit the driver specific properties of the monitor.

Click the ‘Monitor Delete’-button to delete a monitor that is no longer in use.

4.3.1 Watchdog

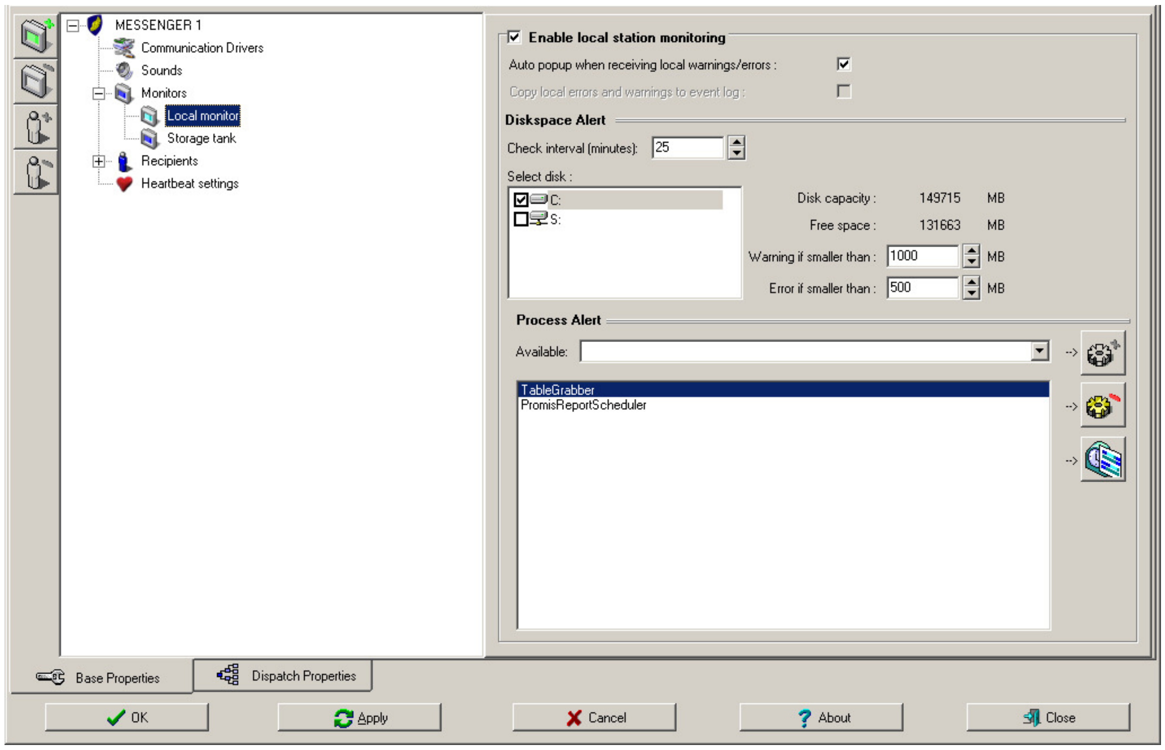
Watchdog enabled:
Enable the ‘Watchdog’-function if you want to raise an alarm when a message-generating source is no longer ‘alive’. In this case, the message-generating source is supposed to send ‘Heartbeat’-messages at regular intervals.

Watchdog time out:
When no ‘Heartbeat’ message is received within the specified timeout, an alarm will be raised!

Watchdog schedule:
Click the ‘Schedule’-button to define when the message-generating source is supposed to be ‘alive’.

4.3.2 The Local Monitor

The ‘Local Monitor’ is used to monitor processes on the ‘Messenger’ computer itself. Applications using the ‘Messenger’-API can pass their messages to and are monitored by the ‘Messenger’. It can also be used to monitor if applications without ‘Messenger’-API are running.



[✓] Enable local station monitoring

Auto popup when receiving local warnings/errors : [✓]

Copy local errors and warnings to event log : [ ]

Diskspace Alert

Check interval (minutes): 25

Select disk :

[✓] C:

[ ] S:

Disk capacity :	149715	MB
Free space :	131663	MB
Warning if smaller than :	1000	MB
Error if smaller than :	500	MB

Process Alert

Available:

TableGrabber
PromisReportScheduler

Enable local station monitoring:
Check if you want to use the local monitoring capabilities. (Switching-off this option does not switch-off monitoring of applications using the ‘Messenger’-API.

Disk-space Alert:
Here you can select the disks you want to watch, per disk you can specify a warning and an error threshold.

Warning if smaller than:
When the disk-space of the selected disk becomes less than this value, the local monitor will generate a warning message.

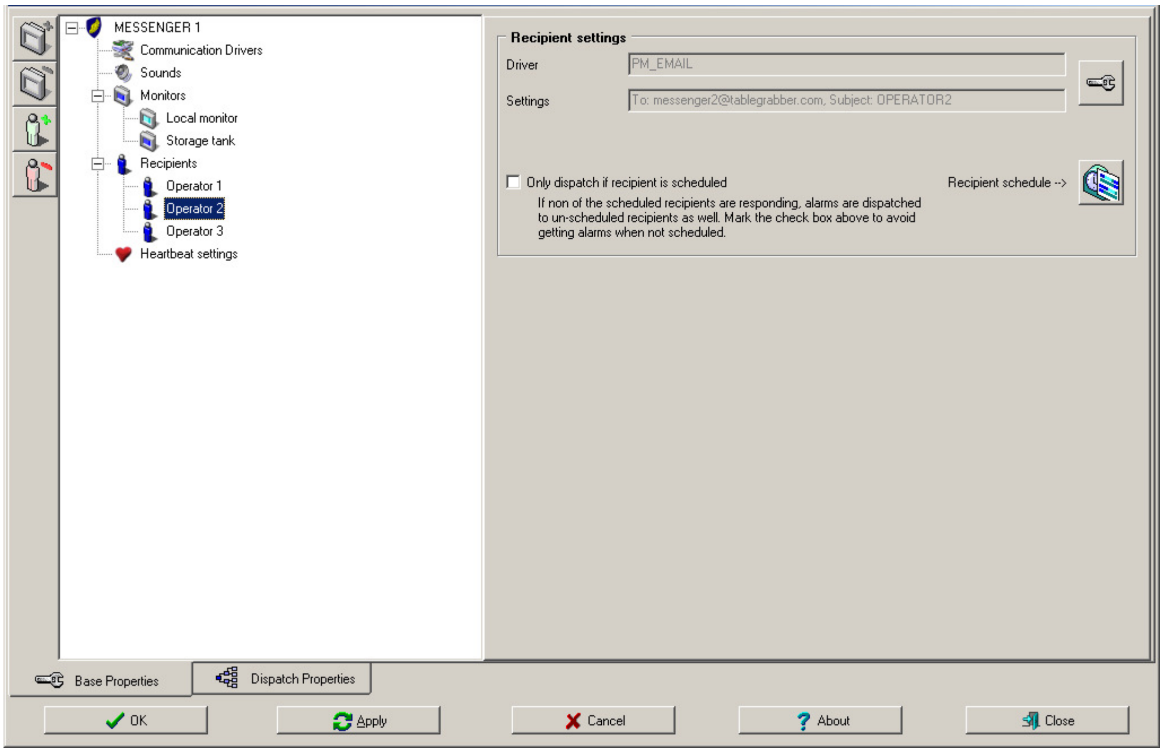
Error if smaller than:
When the disk-space of the selected disk becomes less than this value, the local monitor will generate an error message.

Processes:
Here you can specify which processes should be running when the ‘Messenger’ is running. If one of the specified processes is not running an error will be generated. For each process you can specify a schedule to define when the process is supposed to run.

Add a process by selecting or typing its name followed by pressing the ‘Add’-button.

4.4 Recipients

Recipients are the destinations where messages should be send to. A recipient can be a text-based message receiving system, like a mobile phone, an e-Mail client or another ‘Messenger’. It can also be an actuator, which can be signaled through MODBUS, RS232/485, etc….



Recipient settings

Driver: PM_EMAIL

Settings: To: messenger2@tablegrabber.com, Subject: OPERATOR2

☐ Only dispatch if recipient is scheduled

If non of the scheduled recipients are responding, alarms are dispatched to un-scheduled recipients as well. Mark the check box above to avoid getting alarms when not scheduled.

Recipient schedule -->

Click the ‘Add’-button to add a new recipient, select the appropriate communication driver and configure the driver specific properties. In example an e-mail address to deliver messages at.

Click the ‘Edit’-button to edit the driver specific properties of the recipient.

Click the ‘Recipient Delete’-button to delete a recipient that is no longer in use. A recipient can’t be deleted if it is used in the hierarchical dispatching structure.

Click the ‘Schedule’-button to define when the recipient is supposed to be on duty!

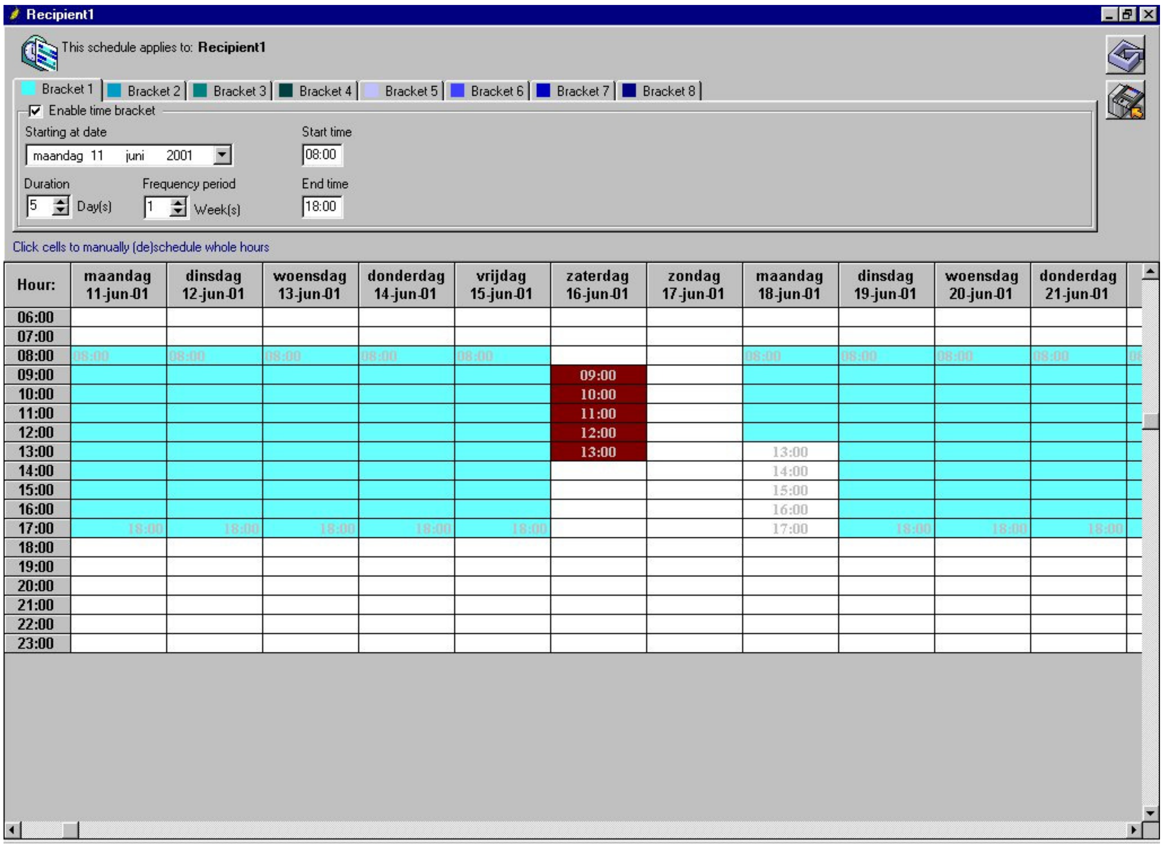
4.5 How to create a schedule

In the ‘Messenger’ it’s possible to define schedules to define when a monitor or process is supposed to be alive or when a recipient is supposed to be on duty. This chapter will explain the possibilities of the scheduler and how it can be configured.

In the scheduler you can specify up to 8 repetitive time brackets and/or you can manually schedule (max. 3 weeks in advance) or de-schedule hours. It’s easy to see if a subject is scheduled, because the corresponding hour cells will be colored.

We will explain the working of the scheduler with some examples.

4.5.1 Schedule example 1


Recipient1

This schedule applies to: **Recipient1**

Bracket 1 | Bracket 2 | Bracket 3 | Bracket 4 | Bracket 5 | Bracket 6 | Bracket 7 | Bracket 8

☑ Enable time bracket

Starting at date: maandag 11 juni 2001		Start time: 08:00
Duration: 5 Day(s)	Frequency period: 1 Week(s)	End time: 18:00

Click cells to manually (de)schedule whole hours

Hour:	maandag 11-jun-01	dinsdag 12-jun-01	woensdag 13-jun-01	donderdag 14-jun-01	vrijdag 15-jun-01	zaterdag 16-jun-01	zondag 17-jun-01	maandag 18-jun-01	dinsdag 19-jun-01	woensdag 20-jun-01	donderdag 21-jun-01
06:00											
07:00											
08:00	🟦 08:00	🟦 08:00	🟦 08:00	🟦 08:00	🟦 08:00			🟦 08:00	🟦 08:00	🟦 08:00	🟦 08:00
09:00	🟦	🟦	🟦	🟦	🟦	🟥 09:00		🟦	🟦	🟦	🟦
10:00	🟦	🟦	🟦	🟦	🟦	🟥 10:00		🟦	🟦	🟦	🟦
11:00	🟦	🟦	🟦	🟦	🟦	🟥 11:00		🟦	🟦	🟦	🟦
12:00	🟦	🟦	🟦	🟦	🟦	🟥 12:00		🟦	🟦	🟦	🟦
13:00	🟦	🟦	🟦	🟦	🟦	🟥 13:00		13:00	🟦	🟦	🟦
14:00	🟦	🟦	🟦	🟦	🟦			14:00	🟦	🟦	🟦
15:00	🟦	🟦	🟦	🟦	🟦			15:00	🟦	🟦	🟦
16:00	🟦	🟦	🟦	🟦	🟦			16:00	🟦	🟦	🟦
17:00	🟦 18:00	🟦 18:00	🟦 18:00	🟦 18:00	🟦 18:00			17:00	🟦 18:00	🟦 18:00	🟦 18:00
18:00											
19:00											
20:00											
21:00											
22:00											
23:00											

Situation

We want to schedule a recipient Monday to Friday from 08.00h till 18.00h. The schedule should be active from 11 June 2001, and it should repeat every week. On Saturday the 16th of June 2001 he takes over duty from a colleague for a few hours and on Monday the 18th of June 2001 he has a half-day of.

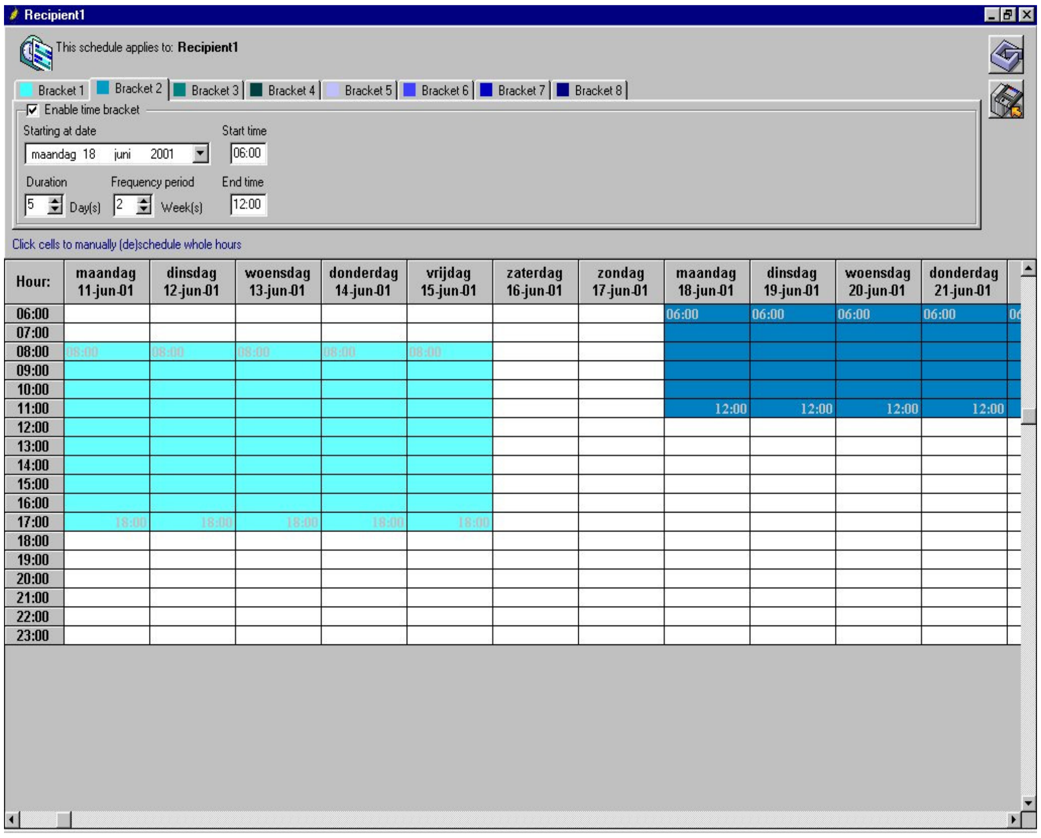
Solution

We begin with the repetitive time-bracket, every week Monday to Friday from 08.00h till 18.00h, starting at 11th of June 2001. For this we take the first bracket (Bracket1) we check the checkbox and fill in the following properties. At starting at date, we choose 11th of June 2001. At start time 08.00h and as end time 18.00h. The time bracket is only for Monday till Friday, those are 5 days so duration is set to 5. The time bracket should repeat every week so frequency is set to 1.

Ok, now we have a time bracket Monday to Friday 08.00h till 18.00h repeating every week. For the extra work on Saturday the 16th of June we simply click on the cells corresponding the hours he will be on duty. For the half-day of on Monday the 18th of June we click on the cells corresponding the hours that he will be of duty.

Now you should see a schedule that looks like the schedule above. The colored cells show the hours that the recipient is scheduled.

4.5.2 Schedule example 2



Recipient1

This schedule applies to: **Recipient1**

Bracket 1 | Bracket 2 | Bracket 3 | Bracket 4 | Bracket 5 | Bracket 6 | Bracket 7 | Bracket 8

☑ Enable time bracket

Starting at date: maandag 18 juni 2001		Start time: 06:00
Duration: 5 Day(s)	Frequency period: 2 Week(s)	End time: 12:00

Click cells to manually (de)schedule whole hours

Hour:	maandag 11-jun-01	dinsdag 12-jun-01	woensdag 13-jun-01	donderdag 14-jun-01	vrijdag 15-jun-01	zaterdag 16-jun-01	zondag 17-jun-01	maandag 18-jun-01	dinsdag 19-jun-01	woensdag 20-jun-01	donderdag 21-jun-01
06:00								🟦 06:00	🟦 06:00	🟦 06:00	🟦 06:00
07:00								🟦	🟦	🟦	🟦
08:00	🟩 08:00	🟩 08:00	🟩 08:00	🟩 08:00	🟩 08:00			🟦	🟦	🟦	🟦
09:00	🟩	🟩	🟩	🟩	🟩			🟦	🟦	🟦	🟦
10:00	🟩	🟩	🟩	🟩	🟩			🟦	🟦	🟦	🟦
11:00	🟩	🟩	🟩	🟩	🟩			🟦 12:00	🟦 12:00	🟦 12:00	🟦 12:00
12:00	🟩	🟩	🟩	🟩	🟩						
13:00	🟩	🟩	🟩	🟩	🟩						
14:00	🟩	🟩	🟩	🟩	🟩						
15:00	🟩	🟩	🟩	🟩	🟩						
16:00	🟩	🟩	🟩	🟩	🟩						
17:00	🟩 18:00	🟩 18:00	🟩 18:00	🟩 18:00	🟩 18:00						
18:00											
19:00											
20:00											
21:00											
22:00											
23:00											

Situation

Lets say we want to create a schedule for a recipient that is differing whether it is an odd or even week number. For even weeks the recipient works on Monday till Friday from 08.00h till 18.00h and for odd weeks he works from Monday till Friday from 06.00h till 12.00h.
The schedule should start on the 11th of June 2001 (even week).

Solution

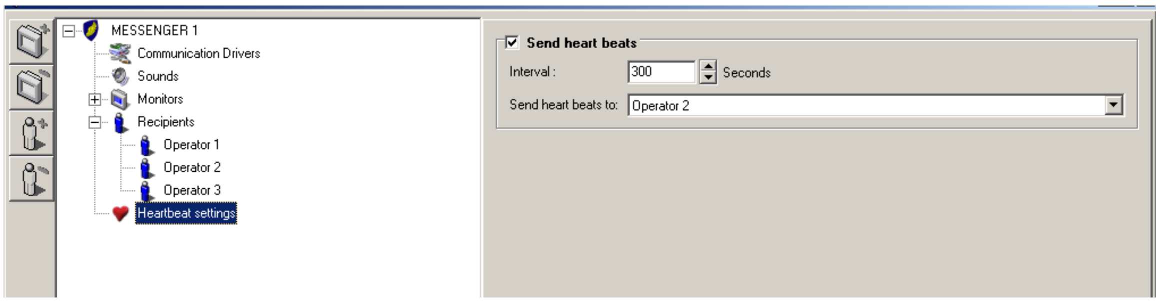
For the even weeks we take the first bracket (Bracket1) we check the checkbox and fill in the following properties. At starting at date, we choose 11th of June 2001. At start time 08.00h and as end time 18.00h. The time bracket is only for Monday till Friday, those are 5 days so duration is set to 5. The time bracket should repeat every two weeks so frequency is set to 2.

For the odd weeks we take the second bracket (Bracket2) we check the checkbox and fill in the following properties. At starting at date, we choose 18th of June 2001 (11th of June + 1 week). At start time 06.00h and as end time 12.00h. The time bracket is only for Monday till Friday, those are 5 days so duration is set to 5. The time bracket should repeat every two weeks so frequency is set to 2.

Now you should see a schedule that looks like the schedule below. The colored cells show the hours that the recipient is scheduled.

4.6 Heartbeat settings

To monitor that a ‘Messenger’ computer is still running, it is possible to send heartbeats to a recipient with a regular interval. If the recipient is another ‘Messenger’, the heartbeat sending ‘Messenger’ has to be configured as a monitor in the receiving ‘Messenger’ and the watchdog settings of that monitor have to be set. When the heartbeat receiving ‘Messenger’ does not receive the heartbeats in time, it will generate a ‘Missed heartbeat’ message of type error with category support. This error will be dispatched to recipients assigned to the ‘Local Monitor’ of the heartbeat receiving ‘Messenger’.



[✓] Send heart beats

Interval : 300 Seconds

Send heart beats to: Operator 2

Send heartbeats:
Check whether or not to send heartbeats.

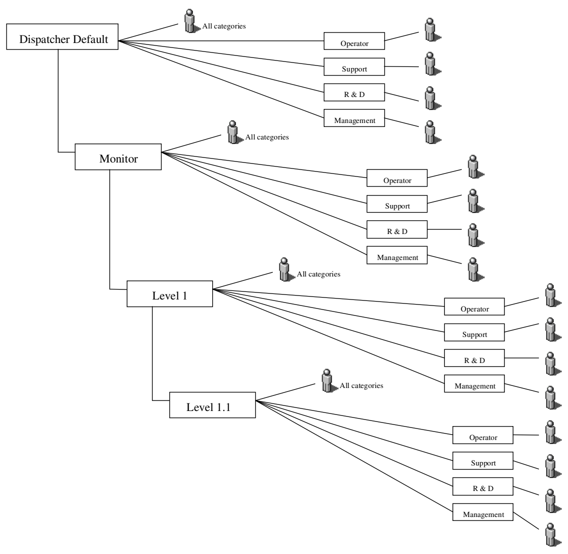
Interval:
The intervals in seconds at which the heartbeats are regularly send.

Send heartbeats to:
Select one of the recipients to send the heartbeats to. Only recipients capable to receive heartbeats will be listed.

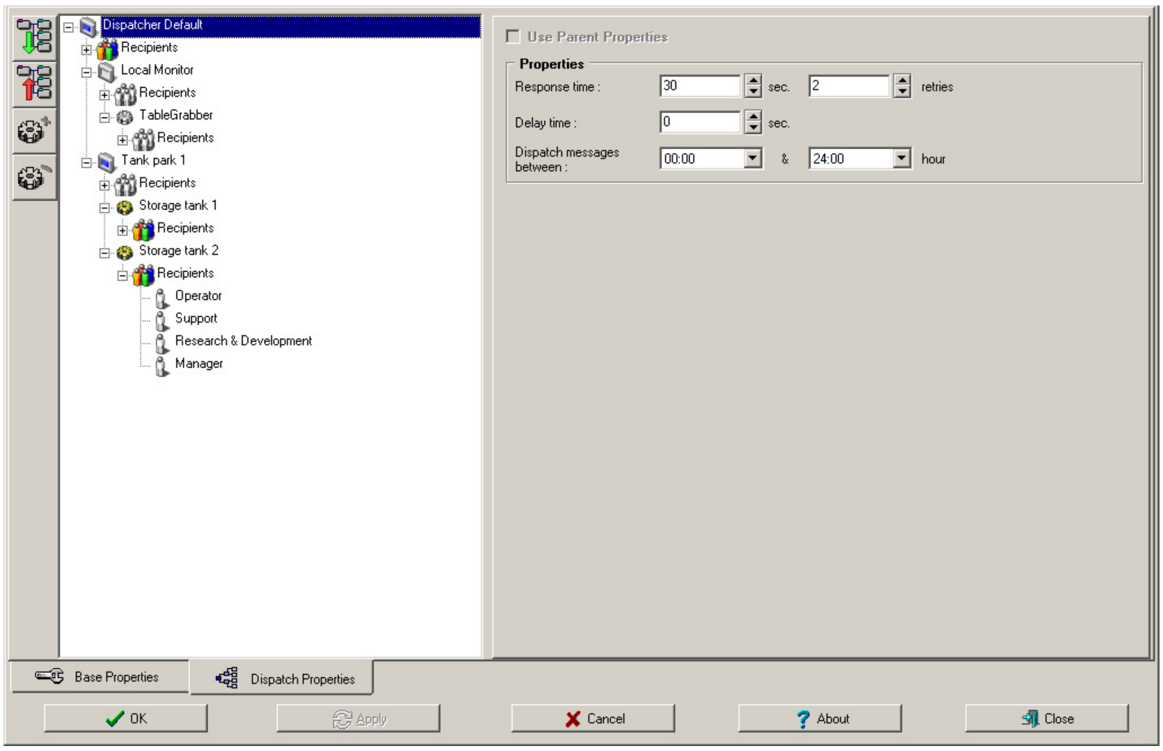
5. Dispatch Properties

5.1 Introduction

The ‘Messenger’ is provided with a hierarchical fallback mechanism to ensure that alarms are being noticed. Messages can be introduced in the system at different levels. When a message is introduced at a certain level, a list will be built of all recipients assigned to that level and who are scheduled at that time. The message will be sent to the first recipient in the list, which must respond within a certain amount of time. If he does not respond in time, the message will be sent to the next recipient, etc. If nobody has responded at all, the list can be tried again and again. If still nobody has responded, then the non-scheduled recipients will be tried once. If still nobody has responded or if there were no recipients assigned to this specific level, the recipients at the parent level will be tried, etc. etc. until the recipients assigned to the ‘Dispatchers Default’ level.



A ‘Messenger’ Message contains beside the message text itself, two additional fields (origin & category), that are used by the dispatching algorithm. The ‘origin’ field describes from which monitor the message is originating from and from which (sub) level within that monitor. Each monitor and (sub) level can have its own recipient list. The ‘category’ field can be used (not obligated) to divide the recipient lists at each level in categories. At each (sub) level you can have a separate recipient list for messages meant for operators, support engineers, R&D and Management. We advise to add at least one always-scheduled recipient to the ‘Recipients’-list at the ‘Dispatcher Default’ level, so you are sure that all alarms will be noticed by at least one recipient.



Use Parent Properties

Properties

Response time : 30 sec. 2 retries

Delay time : 0 sec.

Dispatch messages between : 00:00 & 24:00 hour

The left pane shows the hierarchical dispatching structure, with at the top-level the 'Dispatcher Default' and at the second level all the defined monitors, including the 'Local Monitor'. Within this pane, you can add (sub) levels to the monitors. For compliant monitors (not other Messengers) you can add so called 'tags' at each (sub) level. A 'tag' describes a condition to associate incoming messages with a certain (sub) level. It can also be a command to read out a message or sensed value.

The right pane can be used to edit various dispatching properties depending of the selected item in the left pane. If you select a monitor-node or (sub) level-node, you can set timing properties applicable to that (sub) level and add tags to be associated with that (sub) level. If you select a recipients-node you can add the recipients that are supposed to respond to alarms at that level including its sub-levels.

Click the 'Expand'-button to open all (sub) nodes in the left pane.

Click the 'Collapse'-button to collapse all (sub) nodes in the left pane.

Add a (sub) level to the selected level.

Delete the selected (sub) level.

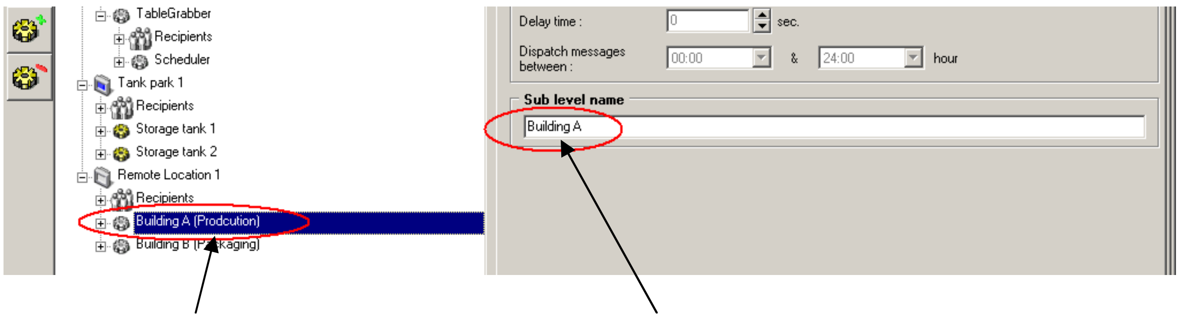
5.2 Hierarchical dispatching structure

There is a difference approach for defining the hierarchical dispatching structure for compatible monitors and compliant monitors. Compatible monitors are the ‘Local Monitor’ and other ‘Messengers’. For compatible monitors, the level of a message is determined by the compatible monitor and given in the message ‘Origin’ text. For compliant monitors you can associate a message to a specific level yourself.

5.2.1 Compatible monitors

When a compatible monitor receives a message it receives an ‘Origin’ text as well. This ‘Origin’ text specifies from which monitor and from which level this message originates from. The (sub) levels are separated from each other with a slash (/). The first level is always the monitor as defined in this ‘Messenger’. The monitor-level is always pre-created in the hierarchical dispatching structure. To treat (sub) level messages differently, you should add a (sub) level to the hierarchical dispatching structure. You can specify you own name for that level in the left-pane and you should specify a ‘sub level name’ matching a part of the ‘origin’ text in the right pane.

Example origin text: *Remote Location 1/Building A/Floor 1*



Delay time : 0 sec.
Dispatch messages between : 00:00 & 24:00 hour
Sub level name: Building A

Your own level name

Part of the origin text

For applications using our API, the first level is always the ‘Local Monitor’, the second level is the process name of the application using our API. The application could of course sub-divides its messages in (sub) levels as well.

Note: The ‘Origin’ text is case insensitive.

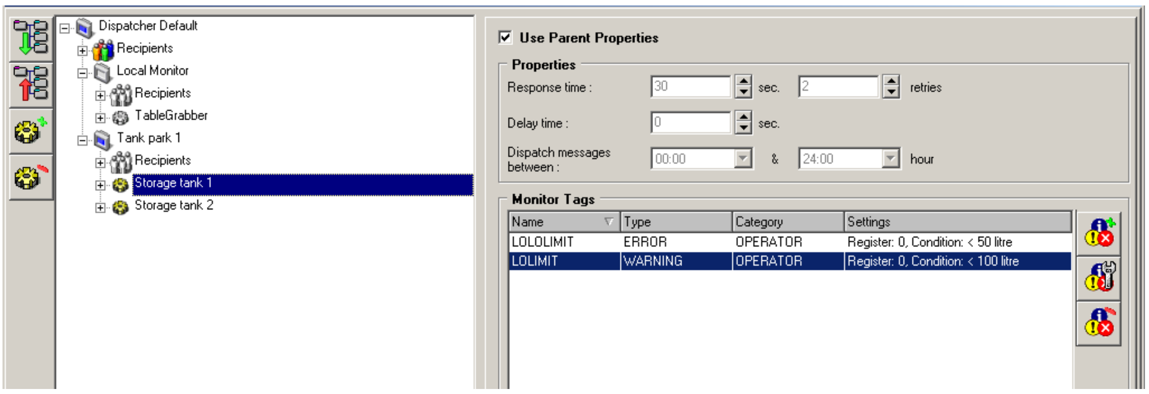
5.2.2 Compliant monitors

Some message generating sources are capable to generate messages autonomously, while others need to get a specific request before they answer.

For sources generating messages autonomously (like an SMS or e-Mail alarm transmitter), all messages are standard dispatched to the recipients assigned to that monitor level. Each incoming message is treated in the same way. Tags can be defined to distinguish messages from each other, so they can be treated in a different way and be sub-divided in different levels. A tag defines a filter used to determine based on textual contents of a message, whether it can or can not be associated with this tag.

For sources requiring requests before they answer (like a MODBUS compatible Data-acquisition-system). A tag defines the request to read-out a sensed value and a condition to determine whether there is an alarm or not.

Tags can be defined at every (sub) level.



- Dispatcher Default
  - Recipients
  - Local Monitor
    - Recipients
    - TableGrabber
  - Tank park 1
    - Recipients
    - Storage tank 1
    - Storage tank 2

☑ Use Parent Properties

Properties

Response time : 30 sec. 2 retries

Delay time : 0 sec.

Dispatch messages between : 00:00 & 24:00 hour

Monitor Tags

Name	Type	Category	Settings
LOLOLIMIT	ERROR	OPERATOR	Register: 0, Condition: < 50 litre
LOLIMIT	WARNING	OPERATOR	Register: 0, Condition: < 100 litre

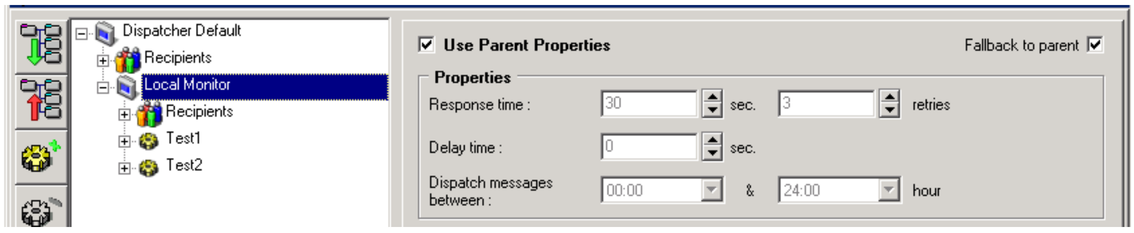
Click the ‘Add’-button to add a new tag and configure the driver specific properties. E.g. a request to read-out a MODBUS value or a filter to associate this tag with a certain message based on its textual content.

Click the ‘Edit’ button to edit the driver specific properties of the selected tag.

Click the ‘Delete’-button to delete the selected tag.

5.3 Dispatch timing

You can specify specific timing properties at each level. After adding a new (sub) level, the timing properties of the parent level is taken and set as default. If you don't want to specify specific timing properties for every (sub) level, you only have to change the timing properties at the 'Dispatcher Default' level.



☑ Use Parent Properties
Fallback to parent ☑

Properties

Response time : 30 sec. 3 retries

Delay time : 0 sec.

Dispatch messages between : 00:00 & 24:00 hour

Use Parent Properties:
Uncheck this option, if you want to specify specific time properties for this level, instead of using those of the parent level.

Fallback to parent:
At default when none of the recipients at a certain level have acknowledged an alarm situation, the Messenger will fall back to the recipients assigned to the parent level. Uncheck this option, if you want to stop the fallback at a certain level.

Response time:
This is the time in seconds in which a recipient is supposed to respond after sending him an alarm.

Retries:
If after an alarm the list with recipients has passed through, but non-of them has responded, the list will be passed through max. 'Retries' again.

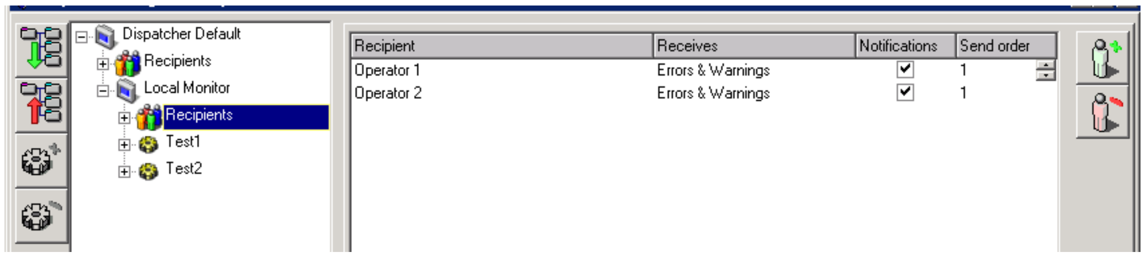
Delay time:
You can introduce a little delay before an incoming alarm will be dispatched, to avoid unnecessarily bothering recipients.

Dispatch messages between:
Some systems are running 24 hours a day and can generate alarms 24 hours a day, it could however be that you don't want to bother none of your recipients during a certain time frame. If you specify a time frame smaller than 24 hours, alarms out-side that time frame will be queued until the time is within the frame. If the issue has been resolved in the mean time, none of the recipients will be bothered.

Examples: If you want to dispatch alarms during office hours only, you could specify a frame like 09:00 – 17:00. If you want to dispatch alarms outside office hours only, you could specify a frame like 17:00 – 09:00.

5.4 Recipients

At each level you can specify a list of recipients who are supposed to receive messages and respond to alarms at that particular level. Recipients can be assigned to a list receiving all categories of messages or they can be assigned to a list receiving only a particular category of messages. If a message is about to be dispatched, there will be no distinction between the recipients assigned to the ‘All’-list and recipients assigned to the ‘Category’-list matching the category of the message. If a recipient is assigned at multiple-levels with a parent/child relationship, only the occurrence closest matching (but not deeper) the level according to the message will be taken into account.



Dispatcher Default
- Recipients
- Local Monitor
  - Recipients
  - Test1
  - Test2

Recipient	Receives	Notifications	Send order
Operator 1	Errors & Warnings	✓	1
Operator 2	Errors & Warnings	✓	1

Click the ‘Add’-button to add a new recipient to the list.

Click the ‘Delete’-button to delete the selected recipient from the list.

You can edit the following properties in the list.

Receives:
Specify which type of messages (errors, warnings and/or hints) the recipient should receive. At default a recipient is set to receive error and warning messages. Regardless of this option, hints are only send to recipients assigned to the exact matching level according to the message. So a recipient assigned to a monitor will never receive hints from a monitor sub level.

Notifications:
Uncheck this option if the recipient should only get the message it self and not the follow-up messages of the message.

Send order:

1) With ‘Send order’ you can specify an order in which recipients will get a message first. Recipients with a lower ‘Send order’ will get alarms earlier then recipients with an higher ‘Send order’. Who gets an alarm earlier is at random for recipients with an equal ‘Send order’.

2) There is no waiting for response delay for a recipient with 'Send order=0', the alarm will be dispatched to a next recipient immediately.

3) Recipients with 'Send order=0' will get the alarms of all sub-levels as well. So a recipient with 'Send order=0' assigned to the 'Dispatcher Default' level, will get all possible alarms immediately. In example, you could connect a matrix printer to a serial port, specify a recipient using the PM_SERIAL driver and assign the recipient to the ‘Dispatcher Default’ level with ‘Send order=0’ to print out all alarms in an old fashioned manner.

6. Database

YDOC-Messenger deploys an open database (ADO/OLEDB) to store its data. The default database as automatically installed with the setup is an Ms-Access database
(C:\ YDOC\ydocmessenger\database\ydocmessenger.mdb).
However It's also possible to use an SQL-Server, Azure SQL or Oracle database.

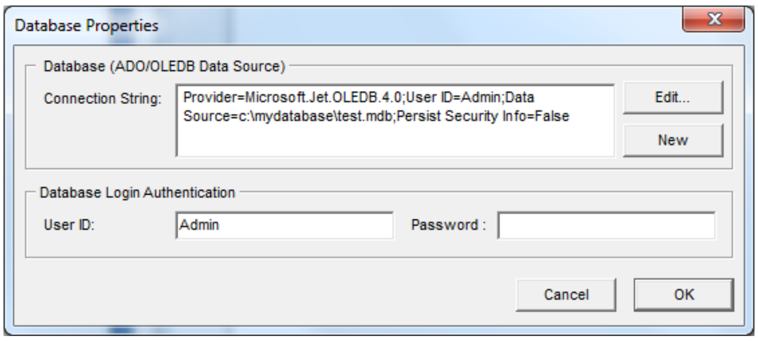
To use a licensed Oracle or SQL-Server database, you have to ask your DBA (Database administrator) to create an empty database. YDOC-Messenger itself will create the necessary tables.

When clicking the 'Change Data Source'-button in the left corner of the "Properties Editor" you can change the database (Data Source) used by YDOC-Messenger.

A Data Source can be specified by its ADO (Microsoft Acces Data Objects) connection string. Database systems like SQL-Server require that you specify authentication info to get access to the database. Please ask your DBA for the right info.

Click the 'Edit…'-button to Edit the existing connection string within a configuration window of the according provider.

Click the 'New''-button if you want to choose another type of database provider.


Database Properties

Database (ADO/OLEDB Data Source)

Connection String: Provider=Microsoft.Jet.OLEDB.4.0;User ID=Admin;Data Source=c:\mydatabase\test.mdb;Persist Security Info=False

Database Login Authentication

User ID: Admin

Password :

ADO Connection string examples:

Ms-Access (Microsoft Jet 4.0 OLE DB): `Provider=Microsoft.Jet.OLEDB.4.0; Data Source=c:\ YDOC\ydocmessenger\database\ydocmessenger.mdb;`

SQL-Server (SQL Native Client): `Provider=SQLNCLI.1; Data Source=<Server>; Initial Catalog=<Database>;`

Azure SQL `(SQL Native Client)`: `Provider=SQLNCLI.1; User ID=<login name>@<server name>; Initial Catalog=<Database>; Data Source=<server name>.database.windows.net,1433;`

Oracle 11g Express(Oracle Provider for OLEDB): `Provider=OraOLEDB.Oracle.1; Persist Security Info=False;User ID=ydocmessenger;Data Source=<Server>:1521/xe`

Oracle 11g (Oracle Provider for OLEDB): `Provider=OraOLEDB.Oracle.1; Persist Security Info=False; User ID=ydocmessenger; Data Source=<Database>`

Where <Server> is the IP address or name of the computer running the database server, <Database> the name of the SQL-server database or Oracle TNS.

Installing SQL-Server Express

If you want to use SQL-server as database system, but have no (corporate) SQL-server database system yet, you can consider to install a free SQL-Server edition.

1. *From mircosoft.com download and install "SQL-Server Express with tools". When asked for during installation choose Express and not LocalDB.*
2. *Start the 'SQL Server Management Studio', connect to your local database engine (`<computer name>\SQLEXPRESS`) using 'Windows Authentication'.*
3. *In the 'Object Explorer' right mouse click 'Databases' and click 'New database' and give the database a name (e.g. YOURDATABASE).*
4. *In the 'Object Explorer' right mouse click 'Databases->Logins->sa' and click 'Properties'.*
5. *Select the 'General'-page, choose 'SQL-server Authentication' and specify a password for the 'sa' login (e.g. yourPassword)*
6. *Select the 'Status'-page, enable the login and grant it connect permission.*
7. *Start YDOC-Messenger and change the dta source connection string to:*
   ```
   Provider=SQLNCLI11.1;Integrated Security=;Persist Security Info=False;User
   ID=sa;Initial Catalog=YOURDATABASE;Data Source=.\SQLEXPRESS;
   ```
8. *Specify as database credentials, User: sa and Password: yourPassword*

Installing Oracle 11g Express

If you want to use Oracle as database system, but have no (corporate) Oracle database system yet, you can consider to install a free Oracle edition.

1. *From Oralce.com download and install "Oracle 11g Express". During installation you are asked to specify a password for the 'SYSTEM' account. Please note and don't forget it. For now we assume the password is yourPassword.*
2. *From Oracle.com download the "32-bit Client Software" as well and install the "Oracle OLEDB driver".*
3. *Start the Oracle SQL command prompt:*
   C:\oraclexe\app\oracle\product\11.2.0\server\bin\sqlplus.exe /nolog
4. *Login to Oracle*
   ```
   SQL>CONNECT SYSTEM as SYSDBA;
   ```
5. *Enter your SYSTEM password: yourPassword*
6. *Create a user for the ydocmessenger database*
   ```
   SQL>CREATE USER ydocmessenger IDENTIFIED BY yourPassword DEFAULT
    TABLESPACE USERS;
   ```
7. ***Note:** The term "DEFAULT TABLESPACE USERS" is very imported, otherwise all your data will be stored in the "TABLESPACE SYSTEM", which has limited space causing your database to bloat after 3 million measurement records.*
8. *Grant rights to the user*
   ```
   SQL>GRANT CONNECT, RESOURCE TO ydocmessenger;
   ```
9. *Optional steps to prepare Oracle Express to allow online backup (backup while the database is in use)*
   a. `SQL>SHUTDOWN IMMEDIATE;`
   b. `SQL>STARTUP MOUNT;`
   c. `SQL>ALTER DATABASE ARCHIVELOG;`
   d. `SQL>ALTER DATABASE OPEN;`
   e. `SQL>ALTER SYSTEM SET DB_RECOVERY_FILE_DEST_SIZE = 16G;`
10. *Start YDOC-Messenger and change the data source connection string to:*
    ```
    Provider=OraOLEDB.Oracle.1;Persist Security Info=False;User
     ID=ydocmessenger;Data Source=localhost:1521/xe;
    ```
11. *Specify as database credentials, User: ydocmessenger and Password: yourPassword*

Appendix 1: Communication Drivers

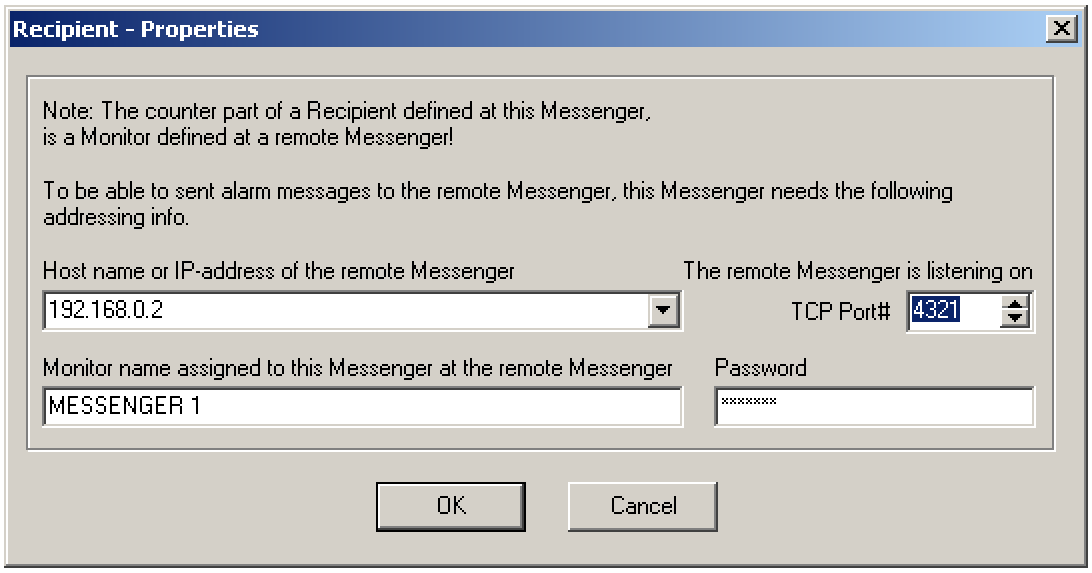
The following ‘Communication Drivers' are available:

1. PM_EMAIL, Exchange text messages through e-Mail (SMTP/POP3)
2. PM_MESSENGER, Exchange messages with other Messengers through TCP/IP.
3. PM_MODBUS, Monitor sensors & signal actuators using MODBUS
4. PM_SERIAL, Exchange ASCII data through serial channels (RS232/485/TCP)
5. PM_SMS, Exchange text messages with mobile phones/modems through SMS (GSM)

PM_MESSENGER (exchange messages with other Messengers)

This driver can be used for exchange of messages between ‘Messengers’. Assume ‘Messenger 1’ is using ‘Messenger 2’ as recipient. Then in ‘Messenger 1’ there should be a recipient defined referring to ‘Messenger 2’ and in ‘Messenger 2’ there should be monitor defined referring to ‘Messenger 1’. You could say that the two ‘Messengers’ are cross-linked:

Recipient properties



Recipient - Properties

Note: The counter part of a Recipient defined at this Messenger,
is a Monitor defined at a remote Messenger!

To be able to sent alarm messages to the remote Messenger, this Messenger needs the following addressing info.

Host name or IP-address of the remote Messenger: 192.168.0.2

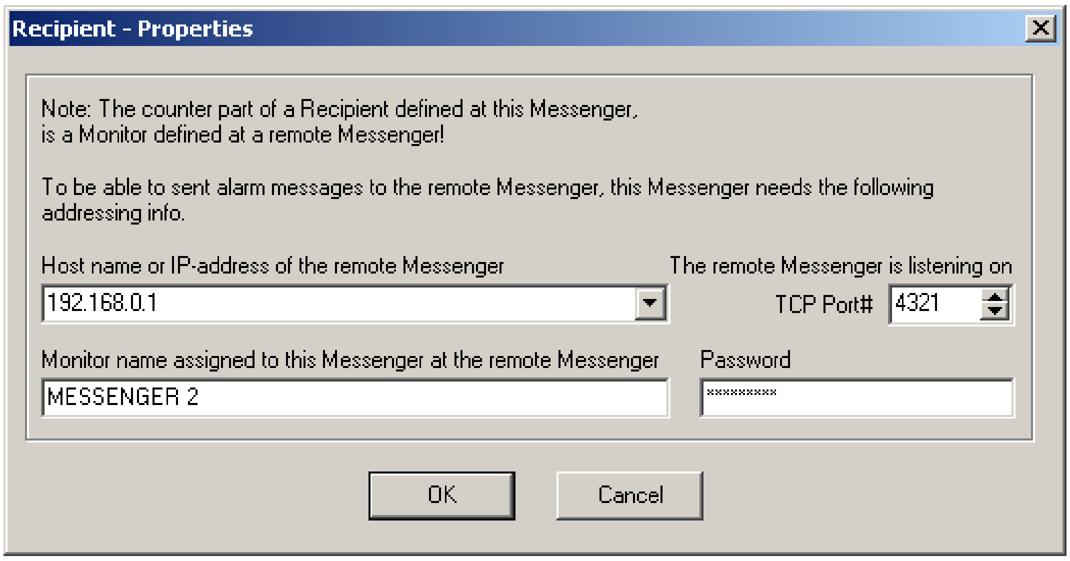
The remote Messenger is listening on TCP Port# 4321

Monitor name assigned to this Messenger at the remote Messenger: MESSENGER 1

Password: *******

Within ‘Messenger 1’ there should be specified at which IP-address and TCP-port ‘Messenger 2’ is listening. Because of security reasons, ‘Messenger 1’ must be authorized by ‘Messenger 2’ and therefore identify itself by providing the monitor name under which ‘Messenger 1’ is known within ‘Messenger 2’, along with an optional password.

Monitor properties



Recipient - Properties

Note: The counter part of a Recipient defined at this Messenger,
is a Monitor defined at a remote Messenger!

To be able to sent alarm messages to the remote Messenger, this Messenger needs the following addressing info.

Host name or IP-address of the remote Messenger: 192.168.0.1

The remote Messenger is listening on TCP Port# 4321

Monitor name assigned to this Messenger at the remote Messenger: MESSENGER 2

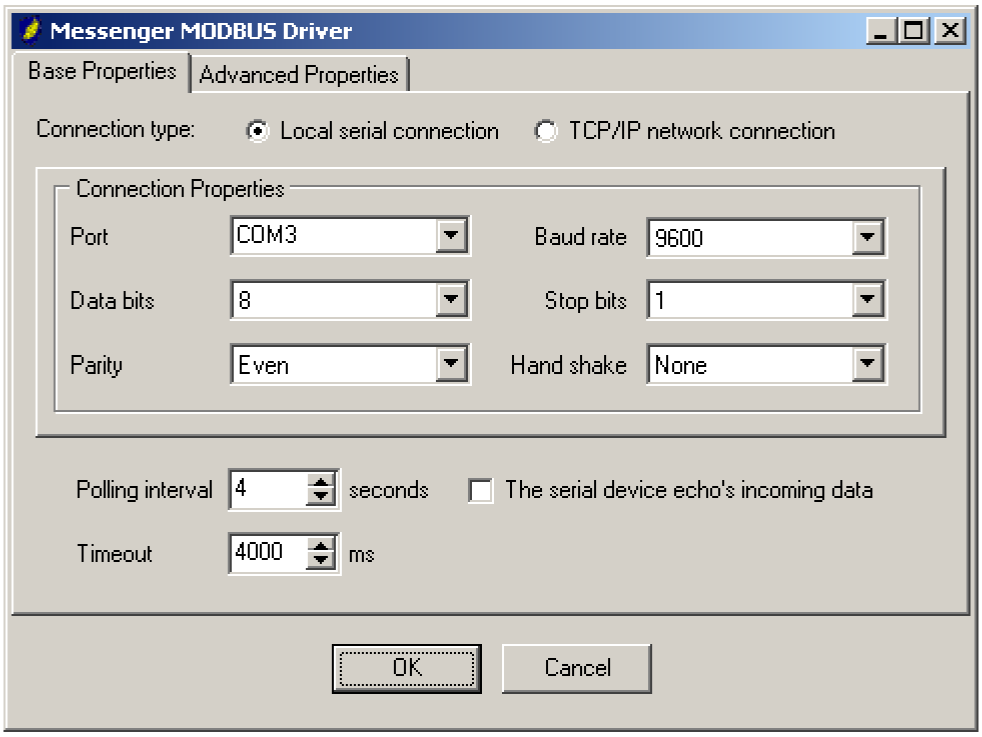
Password: **********

Within ‘Messenger 2’ there should be specified at which IP-address and TCP-port ‘Messenger 1’ is listening. Because of security reasons, ‘Messenger 2’ must be authorized by ‘Messenger 1’ and therefore identify itself by providing the recipient name under which ‘Messenger 2’ is known within ‘Messenger 1’, along with an optional password.

PM_MODBUS (monitor sensors & signal actuators using MODBUS)

This driver can be used to monitor sensors & signal actuators accessible through a MODBUS-RTU network/device.

Driver properties


Messenger MODBUS Driver

Base Properties | Advanced Properties

Connection type: [✓ Local serial connection] TCP/IP network connection

Connection Properties

Port	COM3	Baud rate	9600
Data bits	8	Stop bits	1
Parity	Even	Hand shake	None

Polling interval 4 seconds ☐ The serial device echo's incoming data

Timeout 4000 ms

Connection Properties

In case the MODBUS network is connected to the computer with a serial communication port physically or virtually (USB) you have to specify the used communication port, baud rate etc. If the MODBUS network is connected to the computer through an RS232-TCP/IP converter, the IP-address and TCP-port# of the converter should be specified.

Poll Interval

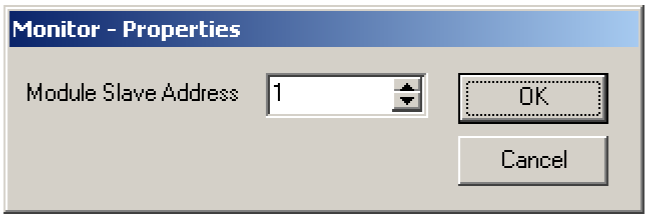
The interval between series of MODBUS readouts. The lower this value the more responsive the system will be, but it could cause an unacceptable network load (in case of using a RS232 TCP/IP converter). The default value is 60 seconds.

Echo & Timeout

If the MODBUS network echo's the commands send from the driver, the driver needs to know that, to filter them out. If a MODBUS slave does not answer within the 'Timeout', the read-out of that slave will be terminated and the driver will continue with the next slave.

Jscript

JScripting can be used, to calculate output values to signal actuators. The 'Advanced properties'-page contains a checkbox to enable JScripting. It contains an edit window as well, that can optionally be used to define global variables or functions (within the scope of the driver instance) that can be used within the recipient specific output value calculation functions.


Monitor - Properties

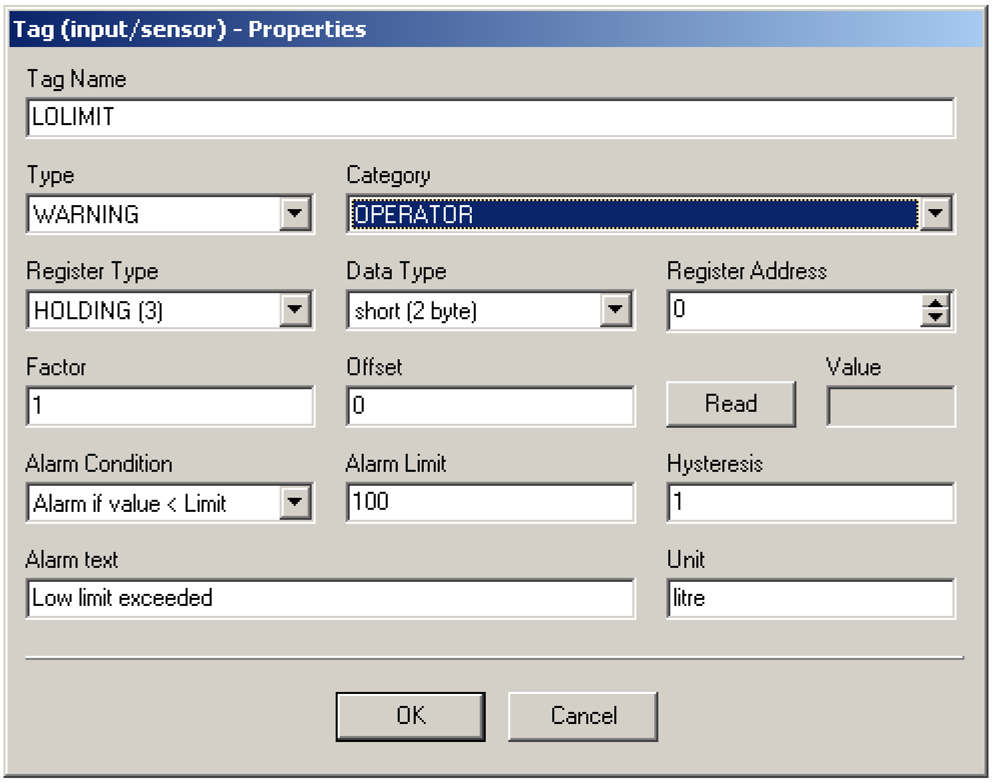
Module Slave Address: 1

Monitor properties

The only monitor setting to be made is the ‘Slave address’ at which a particular MODBUS device is listening. ‘Tags’ are used to instruct which sensors/registers should be monitored.

Tag properties

A ‘tag’ defines which MODBUS sensor/register should be monitored and under which condition an alarm should be raised.


Tag (input/sensor) - Properties

Tag Name: LOLIMIT

Type: WARNING

Category: OPERATOR

Register Type: HOLDING (3)

Data Type: short (2 byte)

Register Address: 0

Factor: 1

Offset: 0

Value:

Alarm Condition: Alarm if value < Limit

Alarm Limit: 100

Hysteresis: 1

Alarm text: Low limit exceeded

Unit: litre

Register Type, The type of register to monitor, which can be; HOLDING(3) or INPUT(4)

Data Type, The type of data to monitor, which can be; unsigned short integer (2 byte), signed long integer (4 byte), floating-point number (4 byte) or double number (8 byte).

Register Address, the address of the register to monitor.

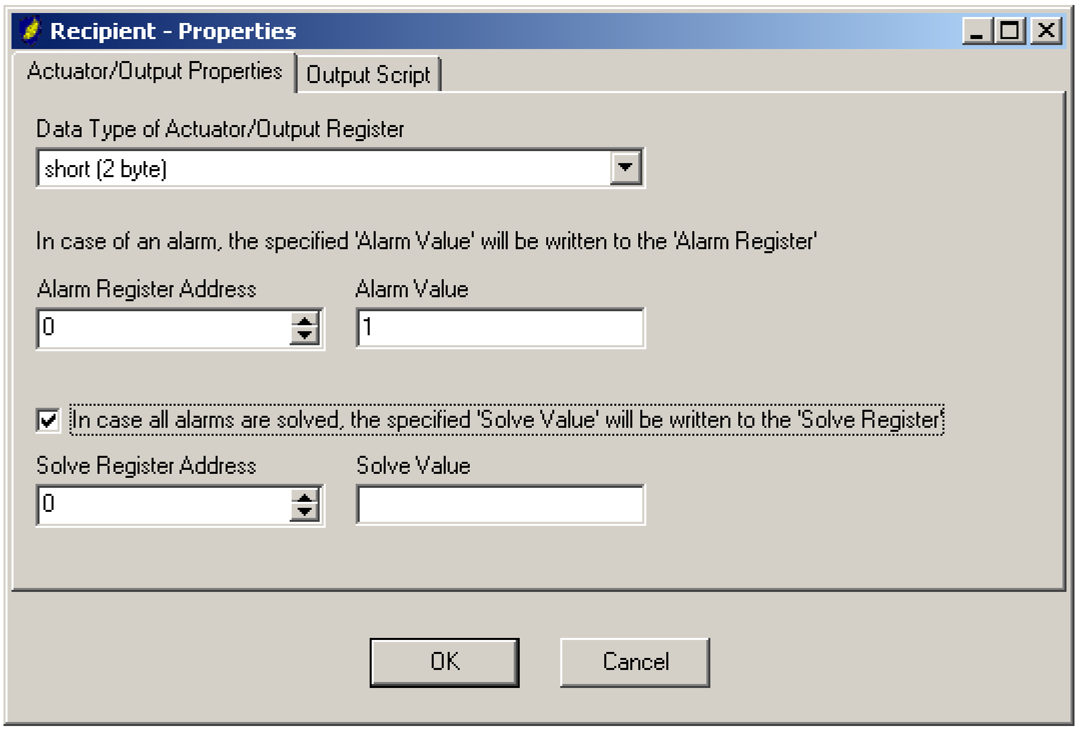
Factor & Offset, can be used to recalculate the value to an engineering unit. The formula is as follows: Eng. value = (register value * factor) + offset.

Alarm Condition, Alarm Limit and Hysteresis, An alarm will be raised if the engineering value exceeds the specified low or high ‘Alarm limit’. A ‘Hysteresis’ can be defined, to avoid nervous alarm behavior. An alarm is released, if the value is within limits, with a margin of at least the ‘Hysteresis’.

Alarm text, the alarm text to be displayed in case of an alarm.

Recipient properties

A MODBUS recipient can be seen as one single actuator, which will be activated in case of an alarm and which will be de-activated if all alarms (concerning this recipient) are solved.


Recipient - Properties

Actuator/Output Properties | Output Script

Data Type of Actuator/Output Register: short (2 byte)

In case of an alarm, the specified 'Alarm Value' will be written to the 'Alarm Register'

Alarm Register Address: 0

Alarm Value: 1

[✓] In case all alarms are solved, the specified 'Solve Value' will be written to the 'Solve Register'

Solve Register Address: 0

Solve Value:

Data Type, The type of data to be written to the MODBUS registers, which can be; unsigned short integer (2 byte), signed long integer (4 byte), floating-point number (4 byte) or double number (8 byte).

Alarm Register & Value

In case of an alarm dispatched to 'this recipient' a fixed value will be written to the specified register. This could be in example a '1' to trigger a relay switching-on a flashing light or buzzer.

Solve Register & Value

In case all alarms concerning 'this recipient' are solved, a fixed value can be written to the specified register. This could be in example a '0' to release a relay switching-off a flashing light or buzzer.

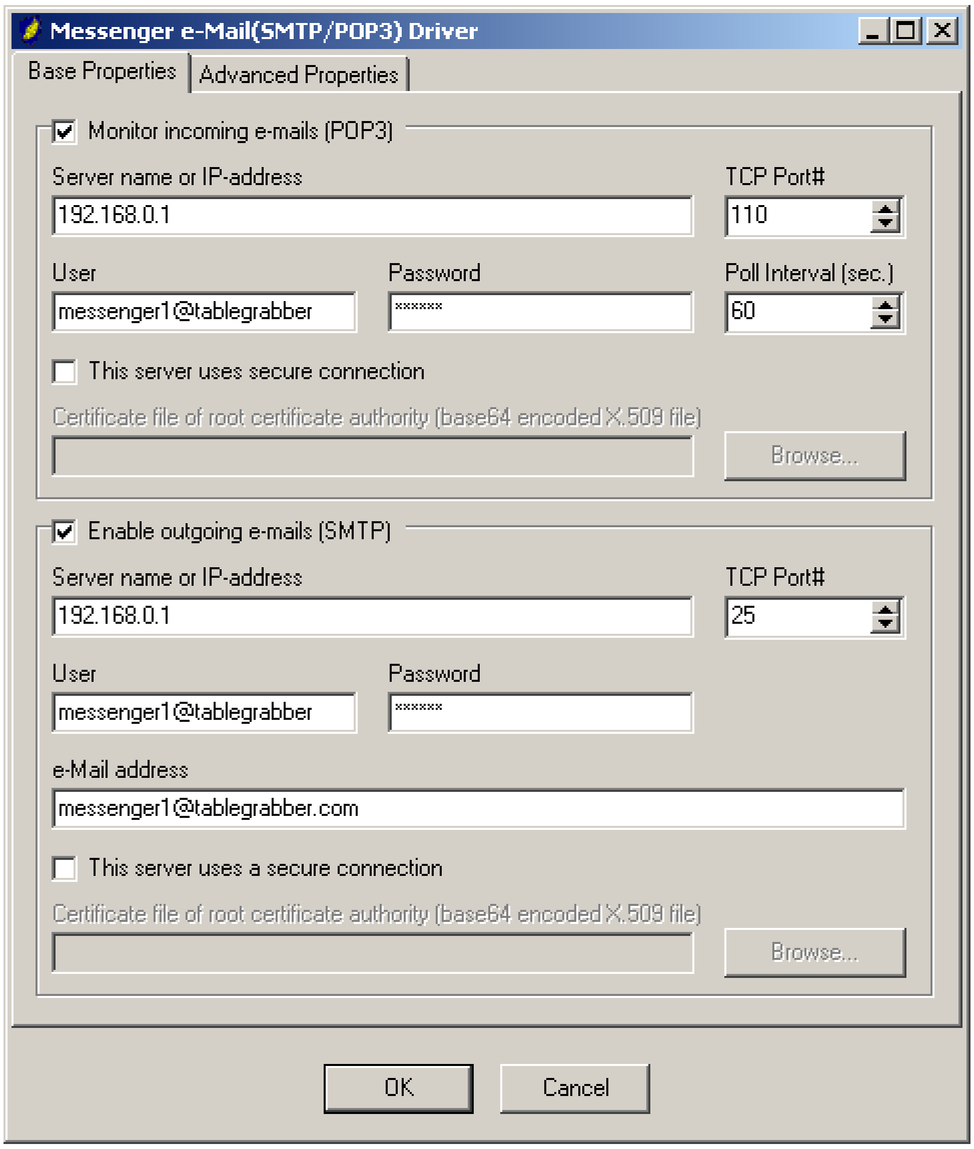
Output Script

JScripting can be used to calculate the output values to be written to the MODBUS device. The 'Output Script'-page contains a checkbox to enable JScripting. It contains an edit window as well, that can be used to define the calculation function.

PM_EMAIL (exchange text messages through e-Mail)

This driver can be used for exchange of text-based message through e-Mail, using the SMTP-protocol for outgoing messages and POP3 for incoming. The driver can be used to accept/send messages from/to e-Mail compatible alarm transmitters/recipients and it can be used for exchanging messages with other 'Messengers' as well. JScripting can be used to pre-process incoming and (re) format outgoing messages, which makes it possible to exchange message with a variety of systems.

Driver properties



[✓] Monitor incoming e-mails (POP3)

Server name or IP-address: 192.168.0.1		TCP Port#: 110
User: messenger1@tablegrabber	Password: ******	Poll Interval (sec.): 60

[ ] This server uses secure connection

Certificate file of root certificate authority (base64 encoded X.509 file):

[✓] Enable outgoing e-mails (SMTP)

Server name or IP-address: 192.168.0.1		TCP Port#: 25
User: messenger1@tablegrabber	Password: ******	

e-Mail address: messenger1@tablegrabber.com

[ ] This server uses a secure connection

Certificate file of root certificate authority (base64 encoded X.509 file):

POP3

POP3 is a protocol for reading incoming messages from an e-Mail server. If this driver is going to be used to process message from alarm transmitters or replies from recipients, it's required specifying the POP3 settings below.

> **Server name or IP-address:** The name or IP-address of the POP-server under which it is known on the Internet. In example: *pop.gmail.com* or *66.249.91.109*
>
> **TCP Port#:** The TCP port number the POP-server is listening at. The default is 110 for a standard POP-server and 465 for a secure one.
>
> **User/Password:** A user & password authorized to read-out e-mails send to the e-mail address used by this 'Messenger'.
>
> **Poll Interval:** The interval between POP-server readouts. The lower this value the more responsive the system will be, but it could cause an unacceptable network load. The default value is 60 seconds.
>
> **Secure connection:** Check this option if the POP-server requires a secure connection. It might be necessary to load a specific X.509 certificate for the 'root certificate authority'

SMTP

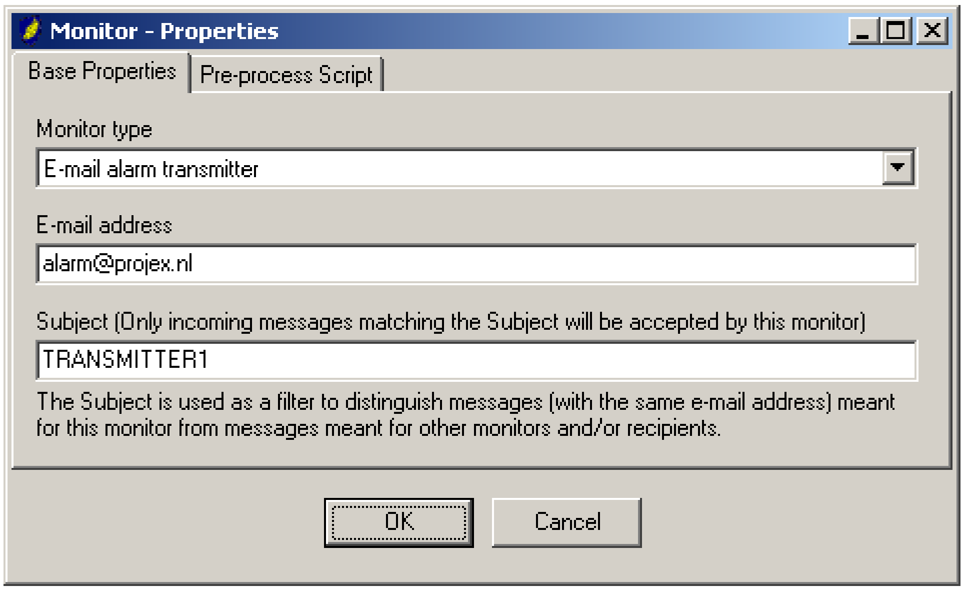
SMTP is a protocol for relaying outgoing messages to an e-Mail server. If this driver is going to be used to dispatch messages, it's required specifying the SMTP settings below.

> **Server name or IP-address:** The name or IP-address of the server under which it is known on the Internet. In example: *smtp.gmail.com* or *66.249.91.109*
>
> **TCP Port#:** The TCP port number the SMTP-server is listening at. The default is 25 for a standard SMTP-server and 995 for a secure one.
>
> **User/Password:** A user & password authorized to relay e-mails through the SMTP-server.
>
> **E-mail address:** The E-mail address used by this 'Messenger' so recipients can recognize the sender of an e-Mail and know where to reply to.
>
> **Secure connection:** Check this option if the SMTP-server requires a secure connection. It might be necessary to load a specific X.509 certificate for the 'root certificate authority'

Jscript

JScripting can be used, if a recipient requires an e-mail in a specific format or if an incoming e-mail needs pre-processing. The 'Advanced properties'-page contains a checkbox to enable JScripting. It contains an edit window as well, that can optionally be used to define global variables or functions (within the scope of the driver instance) that can be used within the specific monitor/recipient pre-process and format functions.

Monitor properties



Monitor type
E-mail alarm transmitter

E-mail address
alarm@projex.nl

Subject (Only incoming messages matching the Subject will be accepted by this monitor)
TRANSMITTER1

The Subject is used as a filter to distinguish messages (with the same e-mail address) meant for this monitor from messages meant for other monitors and/or recipients.

Monitor type

If the e-mail transmitter is another ‘Messenger’, please specify whether it is an old version (<= V2.5) or a new version (>= V3.0), because there is a difference between the e-mail format.

E-mail address

The e-mail address as used by the transmitter as the sender address.

Subject

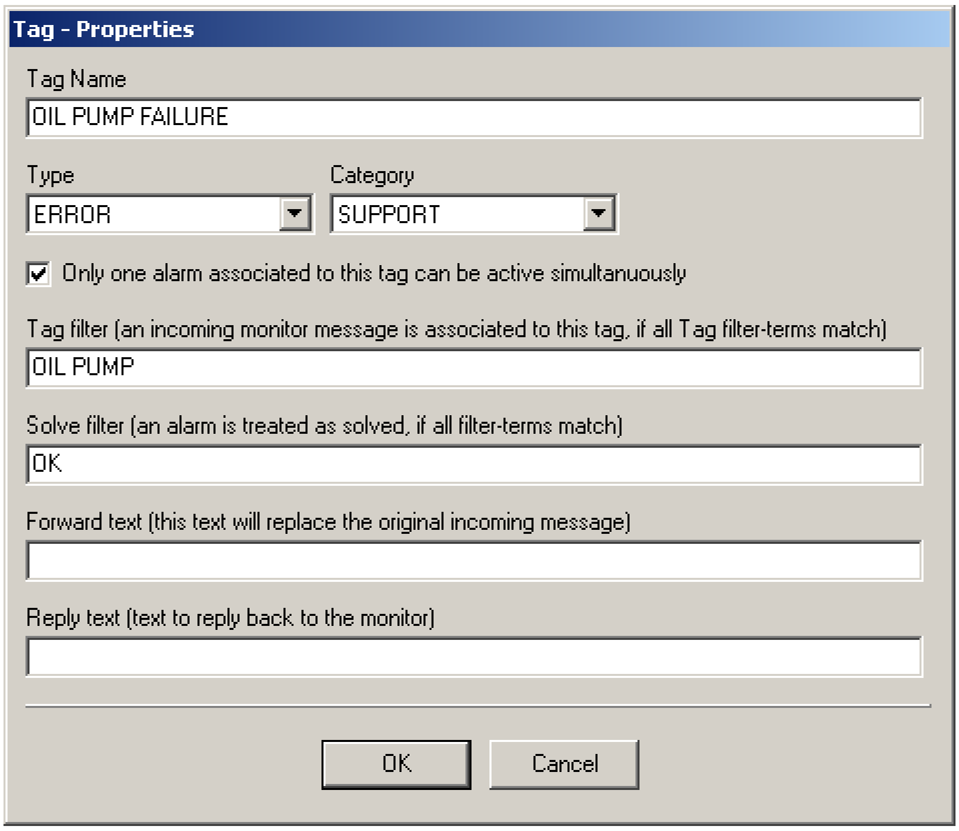
If multiple transmitters are using the same sender e-mail address, the subject can be used as a filter to distinguish between alarm transmitters. Only e-mails containing the fragments as specified in ‘Subject’ will be recognized as messages from ‘this monitor’.

Pre-process Script

JScripting can be used to pre-process incoming messages. The ‘Pre-process Script’-page contains a checkbox to enable JScripting. It contains an edit window as well, that can be used to define a pre-process function, to transform the text of an incoming message into another.

Tag properties

A ‘tag’ describes a condition to distinguish between incoming messages that should be treated differently! Some messages could be warnings about a rising critical situation, while others are just hints. Tags can not be defined, if the monitor is another ‘Messenger’.


Tag - Properties

Tag Name

OIL PUMP FAILURE

Type: ERROR

Category: SUPPORT

[✓] Only one alarm associated to this tag can be active simultanuously

Tag filter (an incoming monitor message is associated to this tag, if all Tag filter-terms match): OIL PUMP

Solve filter (an alarm is treated as solved, if all filter-terms match): OK

Forward text (this text will replace the original incoming message):

Reply text (text to reply back to the monitor):

A ‘Tag filter’ can be used to distinguish between different messages. A ‘Tag Filter’ may contain multiple fragments separated by spaces. Only messages containing the filter fragments (regardless of their order in the text) will be associated with this tag. If a fragment itself contains spaces, it can be enclosed within double quotes to prevent the fragment from being seen as multiple fragments.

Tags will be processed on their ‘Tag filters and fragments’ in descending alphabetical order, for instance to prevent a message containing ‘OIL PUMP’ to be associated with a tag filtering on ‘PUMP’ only, while there is a tag specifically filtering on ‘OIL PUMP’.

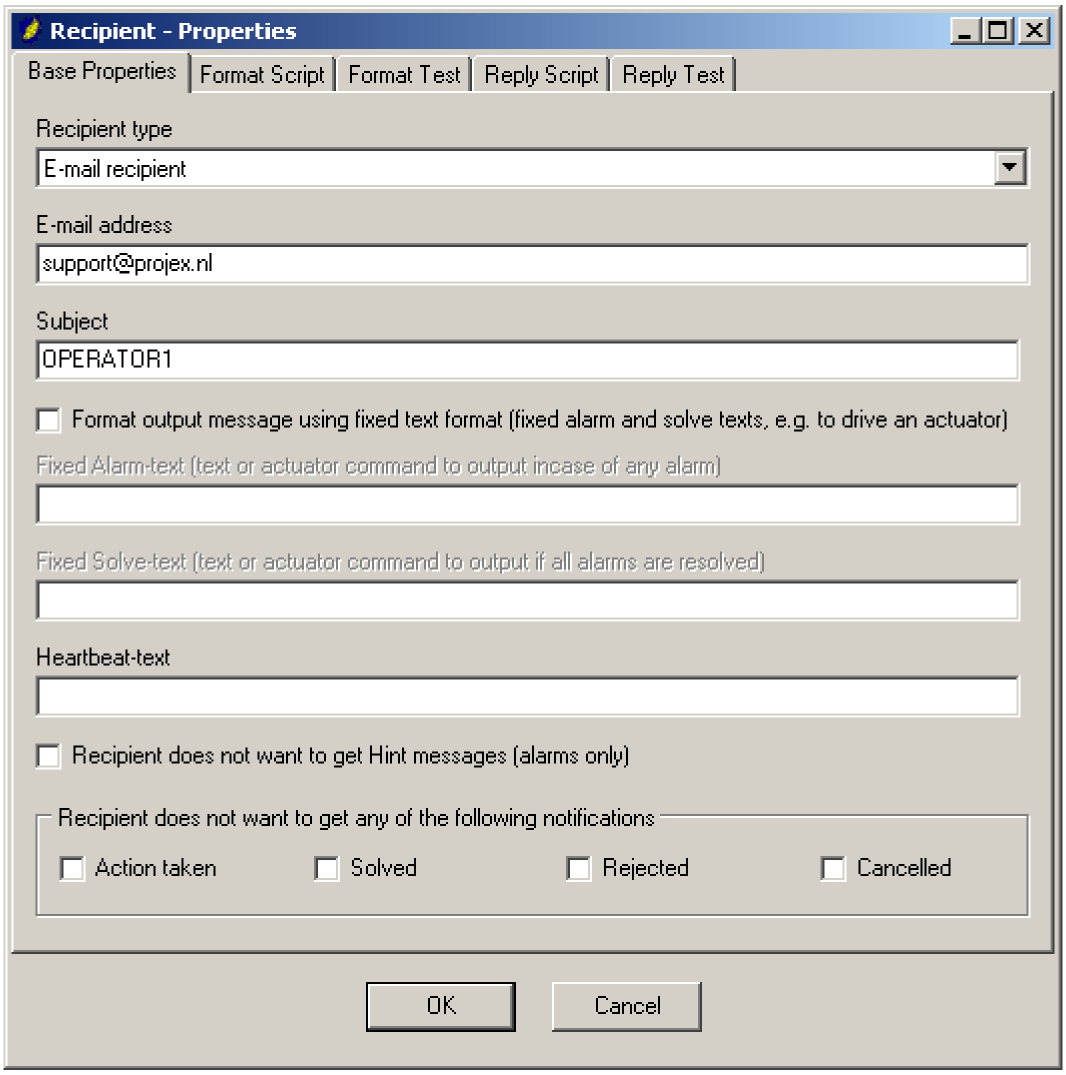
Messages that cannot be associated with a tag are treated as alarms of type ‘Error’ with category ‘Support’. To change this default behavior, you could at a tag with a blank ‘Tag filter’.

If the alarm transmitter is able to send solve messages, the ‘Messenger’ can recognize them, if you specify a ‘Solve filter’. The ‘Solve filter’ may contain, like the ‘Tag filter’, multiple fragments separated by spaces. Only messages matching both the ‘Tag filter’ and ‘Solve filter’ will be treated as a ‘Solve’ message.

If the message text is rather cryptic, it is possible to replace the original text with a ‘Forward text’.

A ‘Reply text’ can be specified if the alarm transmitter requires one.

Recipient properties


Recipient - Properties

Base Properties | Format Script | Format Test | Reply Script | Reply Test

Recipient type: E-mail recipient

E-mail address: support@projex.nl

Subject: OPERATOR1

☐ Format output message using fixed text format (fixed alarm and solve texts, e.g. to drive an actuator)

Fixed Alarm-text (text or actuator command to output incase of any alarm):

Fixed Solve-text (text or actuator command to output if all alarms are resolved):

Heartbeat-text:

☐ Recipient does not want to get Hint messages (alarms only)

Recipient does not want to get any of the following notifications: ☐ Action taken ☐ Solved ☐ Rejected ☐ Cancelled

OK | Cancel

Recipient type

If the e-mail recipient is another ‘Messenger’, please specify whether it is an old version (<= V2.5) or a new version (>= V3.0), because there is a difference between the e-mail format.

E-mail address

The e-mail address to send to.

Subject

The subject of the e-mail. If multiple recipients are using the same e-mail address, the subject is used as a filter to distinguish between replies from ‘this recipient’ and others. Only e-mails containing the fragments as specified in ‘Subject’ will be recognized as messages from ‘this recipient’.

Fixed output message

Instead of transmitting the alarm message itself, it is possible to send fixed text messages, e.g. to signal a e-mail driven actuator. When an alarm is dispatched to ‘this recipient’, the ‘Fixed alarm text’ will be send instead and as soon as all alarms dispatched to ‘this recipient’ are solved, then the ‘Fixed solve text’ will be send to ‘this recipient’

Heartbeat-text

If ‘this recipient’ is supposed to get heartbeat messages, the heartbeat text to send to ‘this recipient’ should be defined.

Format Script

JScripting can be used to (re) format outgoing messages. The ‘Format-Script’-page contains a checkbox to enable JScripting. It contains an edit window as well, that can be used to define a (re) format function, to format the text of the outgoing message.

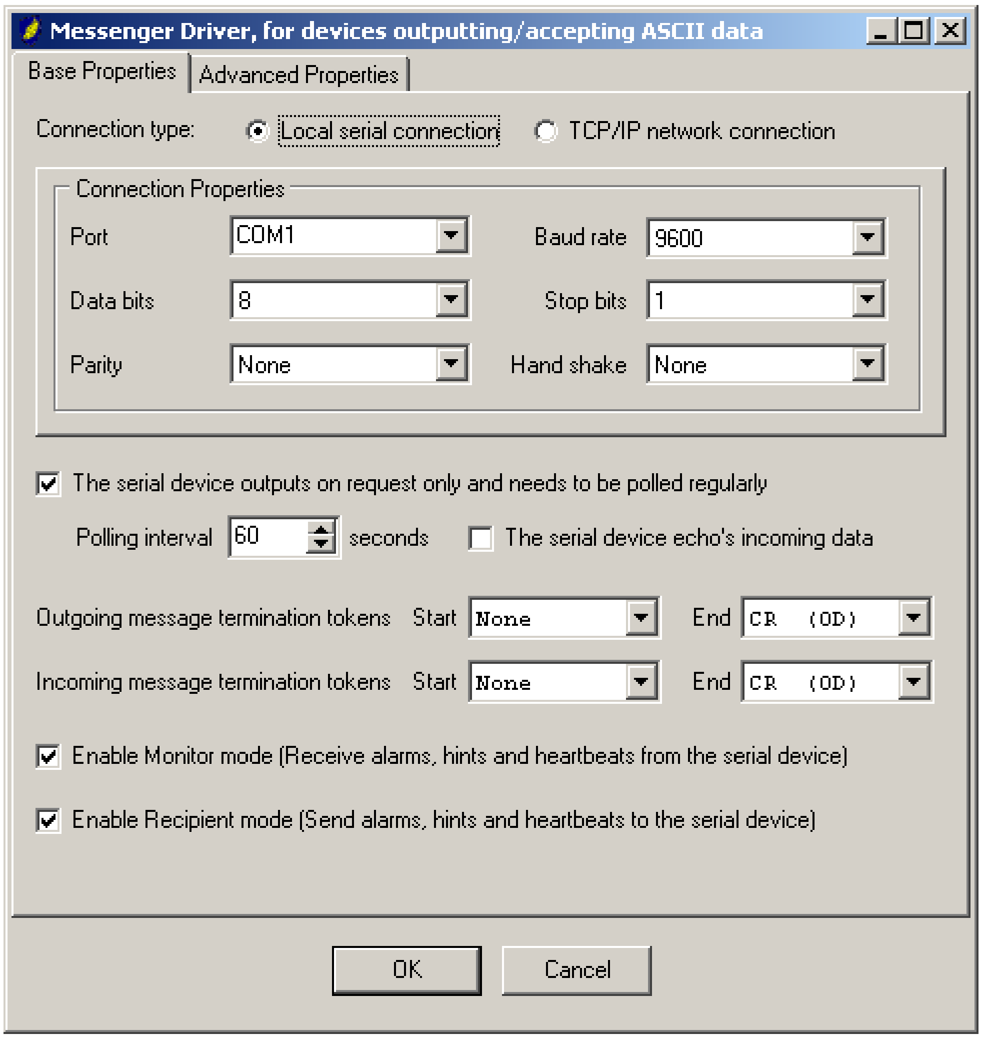
Reply Script

JScripting can be used to pre-process incoming replies. The ‘Reply Script’-page contains a checkbox to enable JScripting. It contains an edit window as well, that can be used to define a pre-process function, to transform the text of an incoming reply into another.

PM_SERIAL (exchange ASCII data through serial channels)

This driver can be used for exchange of text-based message with serial devices (RS232, RS485, RS422 or TCP/IP). JScripting can be used to pre-process incoming and (re) format outgoing messages, which makes it possible to exchange message with a variety of devices.

Driver properties



Messenger Driver, for devices outputting/accepting ASCII data

Base Properties | Advanced Properties

Connection type: [✓ Local serial connection] TCP/IP network connection

Connection Properties

Port	COM1	Baud rate	9600
Data bits	8	Stop bits	1
Parity	None	Hand shake	None

[✓] The serial device outputs on request only and needs to be polled regularly

Polling interval 60 seconds [ ] The serial device echo's incoming data

Outgoing message termination tokens	Start	None	End	CR (0D)
Incoming message termination tokens	Start	None	End	CR (0D)

[✓] Enable Monitor mode (Receive alarms, hints and heartbeats from the serial device)

[✓] Enable Recipient mode (Send alarms, hints and heartbeats to the serial device)

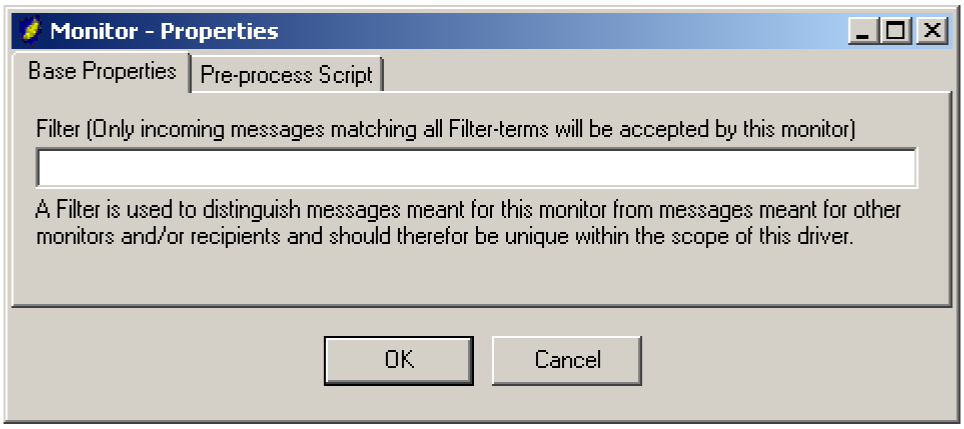
In case the serial device is connected to the computer with a serial communication port physically or virtually (USB) you have to specify the used communication port, baud rate etc. If the serial device is connected to the computer through TCP/IP, the IP-address and TCP-port# where the device is listening at should be specified.

Some serial devices are outputting message autonomously, while others need to be asked to output a message. In such case the driver needs to know how often the device needs to be asked to do so ('Poll interval') The lower the 'Poll interval' the more responsive the system will be, but it could cause an unacceptable network load (in case of using a TCP/IP device). The default value is 60 seconds. Please check the box 'The serial device echo's incoming data' if it does echo the request commands.

Jscript

JScripting can be used, if a recipient requires an e-mail in a specific format or if an incoming e-mail needs pre-processing. The ‘Advanced properties’-page contains a checkbox to enable JScripting. It contains an edit window as well, that can optionally be used to define global variables or functions (within the scope of the driver instance) that can be used within the specific monitor/recipient pre-process and format functions.

Monitor properties



Monitor - Properties

Filter (Only incoming messages matching all Filter-terms will be accepted by this monitor)

A Filter is used to distinguish messages meant for this monitor from messages meant for other monitors and/or recipients and should therefor be unique within the scope of this driver.

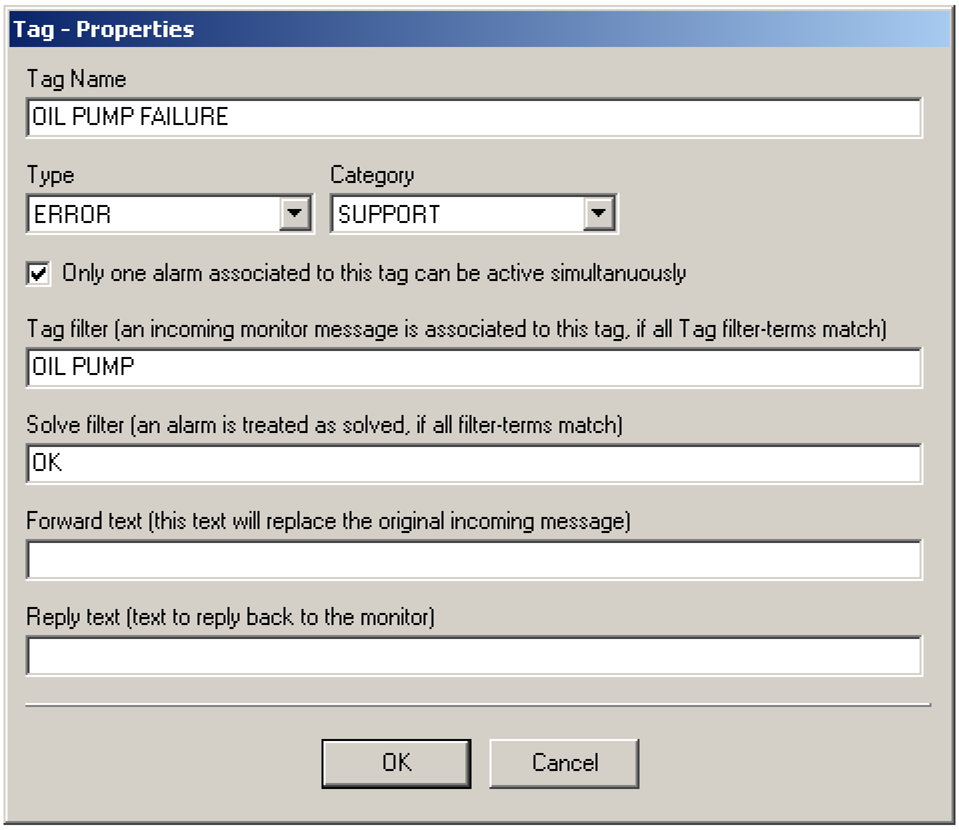
Only messages containing the filter fragments (regardless of their order in the text) will be accepted and processed. If a fragment itself contains spaces, it can be enclosed within double quotes to prevent the fragment from being seen as multiple fragments. Don’t specify a filter, if all messages should be processed.

Pre-process Script

JScripting can be used to pre-process incoming messages. The ‘Pre-process Script’-page contains a checkbox to enable JScripting. It contains an edit window as well, that can be used to define a pre-process function, to transform the text of an incoming message into another.

Tag properties

A ‘tag’ describes a condition to distinguish between incoming messages that should be treated differently! Some messages could be warnings about a rising critical situation, while others are just hints. Tags cannot be defined, if the monitor is another ‘Messenger’.



Tag - Properties

Tag Name: OIL PUMP FAILURE

Type: ERROR

Category: SUPPORT

☑ Only one alarm associated to this tag can be active simultanuously

Tag filter (an incoming monitor message is associated to this tag, if all Tag filter-terms match): OIL PUMP

Solve filter (an alarm is treated as solved, if all filter-terms match): OK

Forward text (this text will replace the original incoming message):

Reply text (text to reply back to the monitor):

A ‘Tag filter’ can be used to distinguish between different messages. A ‘Tag Filter’ may contain multiple fragments separated by spaces. Only messages containing the filter fragments (regardless of their order in the text) will be associated with this tag. If a fragment itself contains spaces, it can be enclosed within double quotes to prevent the fragment from being seen as multiple fragments.

Tags will be processed on their ‘Tag filters and fragments’ in descending alphabetical order, for instance to prevent a message containing ‘OIL PUMP’ to be associated with a tag filtering on ‘PUMP’ only, while there is a tag specifically filtering on ‘OIL PUMP’.

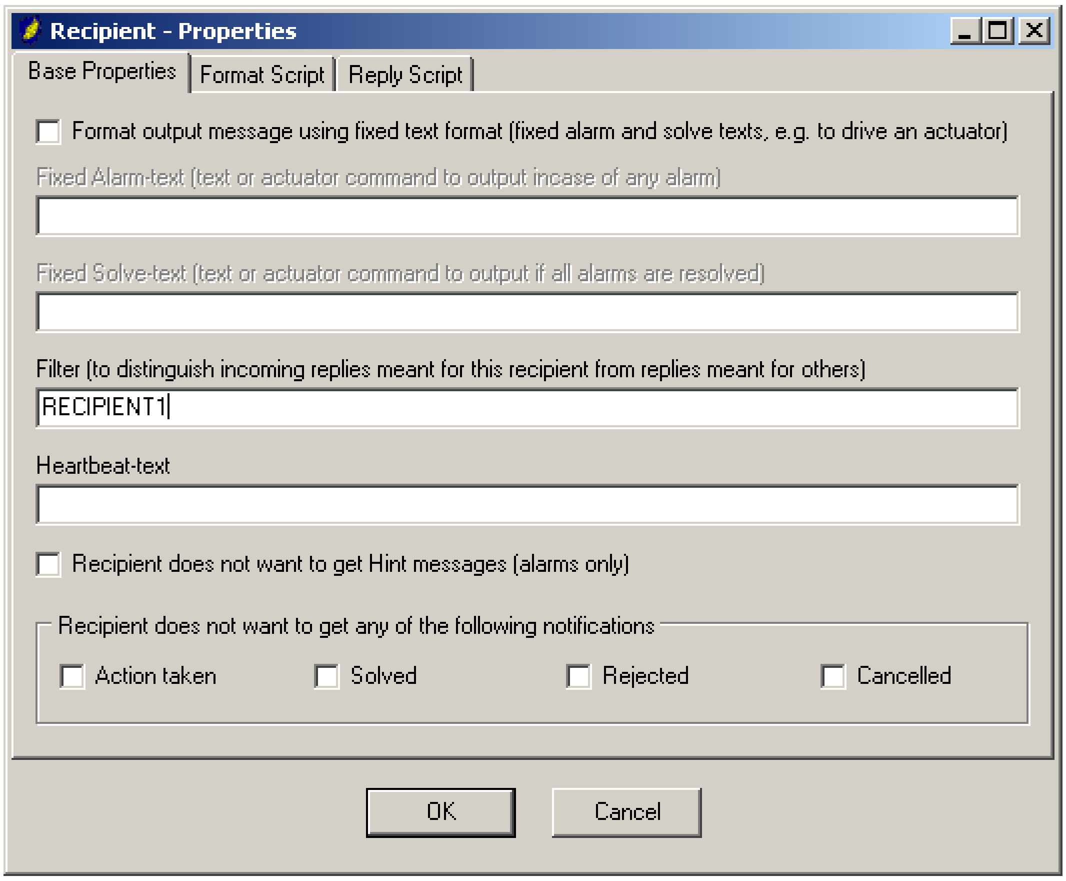
Messages that cannot be associated with a tag are treated as alarms of type ‘Error’ with category ‘Support’. To change this default behavior, you could at a tag with a blank ‘Tag filter’.

If the alarm transmitter is able to send solve messages, the ‘Messenger’ can recognize them, if you specify a ‘Solve filter’. The ‘Solve filter’ may contain, like the ‘Tag filter’, multiple fragments separated by spaces. Only messages matching both the ‘Tag filter’ and ‘Solve filter’ will be treated as a ‘Solve’ message.

If the message text is rather cryptic, it is possible to replace the original text with a ‘Forward text’.

A ‘Reply text’ can be specified if the alarm transmitter requires one.

Recipient properties


Recipient - Properties

Base Properties | Format Script | Reply Script

☐ Format output message using fixed text format (fixed alarm and solve texts, e.g. to drive an actuator)

Fixed Alarm-text (text or actuator command to output incase of any alarm)

Fixed Solve-text (text or actuator command to output if all alarms are resolved)

Filter (to distinguish incoming replies meant for this recipient from replies meant for others)

RECIPIENT1

Heartbeat-text

☐ Recipient does not want to get Hint messages (alarms only)

Recipient does not want to get any of the following notifications

☐ Action taken ☐ Solved ☐ Rejected ☐ Cancelled

OK Cancel

Fixed output message

Instead of transmitting the alarm message itself, it is possible to send fixed text messages, e.g. to signal an actuator. When an alarm is dispatched to 'this recipient', the 'Fixed alarm text' will be send instead and as soon as all alarms dispatched to 'this recipient' are solved, then the 'Fixed solve text' will be send to 'this recipient'.

Filter

A 'Filter' can be used to distinguish between replies from different recipients. A 'Filter' may contain multiple fragments separated by spaces. Only messages containing the filter fragments (regardless of their order in the text) will be seen as a valid reply. If a fragment itself contains spaces, it can be enclosed within double quotes to prevent the fragment from being seen as multiple fragments.

Heartbeat-text

If 'this recipient' is supposed to get heartbeat messages, the heartbeat text to send to 'this recipient' should be defined.

Format Script

JScripting can be used to (re) format outgoing messages. The 'Format-Script'-page contains a checkbox to enable JScripting. It contains an edit window as well, that can be used to define a (re) format function, to format the text of the outgoing message.

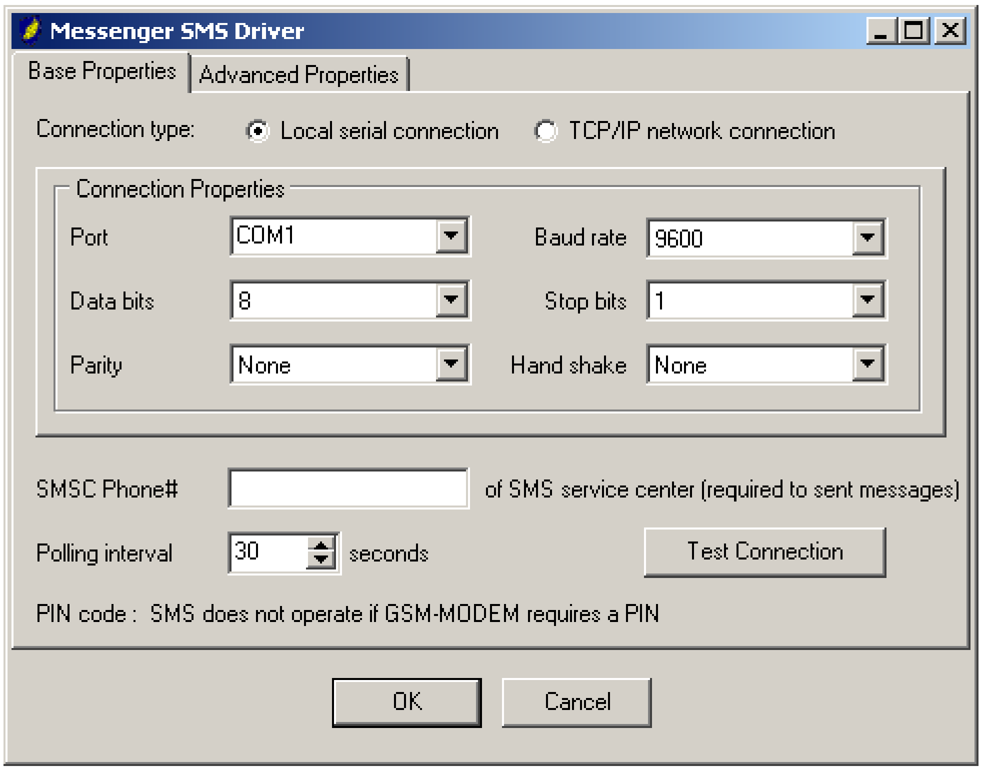
Reply Script

JScripting can be used to pre-process incoming replies. The 'Reply Script'-page contains a checkbox to enable JScripting. It contains an edit window as well, that can be used to define a pre-process function, to transform the text of an incoming reply into another.

PM_SMS (exchange text messages through SMS)

This driver can be used for exchange of text-based message with mobile phones/devices through. The driver can be used to accept/send messages from/to SMS (GSM) compatible alarm transmitters/recipients. JScripting can be used to pre-process incoming and (re) format outgoing messages, which makes it possible to exchange message with a variety of systems.

Driver properties


Messenger SMS Driver

Connection type: [✓ Local serial connection] TCP/IP network connection

Connection Properties

Port	COM1	Baud rate	9600
Data bits	8	Stop bits	1
Parity	None	Hand shake	None

SMSC Phone# ______ of SMS service center (required to sent messages)

Polling interval 30 seconds

PIN code : SMS does not operate if GSM-MODEM requires a PIN

Connection Properties

In case the GSM-Modem is connected to the computer with a serial communication port physically or virtually (USB) you have to specify the used communication port, baud rate etc. If the GSM-Modem is connected to the computer through an RS232-TCP/IP converter, the IP-address and TCP-port# of the converter should be specified.

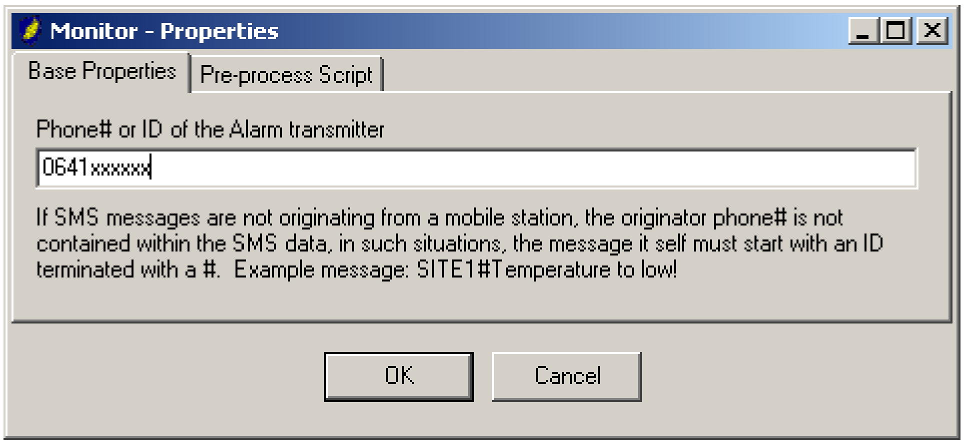
SMSC Phone#

It the ‘Messenger’ is supposed to send SMS-messages, it is required to specify the phone number of an SMS Message Center corresponding to the SIM installed in the GSM-Modem.

Poll Interval: The interval between GSM-Modem readouts. The lower this value the more responsive the system will be, but it could cause an unacceptable network load (in case of using a RS232-TCP/IP converter). The default value is 30 seconds.

Jscript

JScripting can be used, if a recipient requires an e-mail in a specific format or if an incoming e-mail needs pre-processing. The ‘Advanced properties’-page contains a checkbox to enable JScripting. It contains an edit window as well, that can optionally be used to define global variables or functions (within the scope of the driver instance) that can be used within the specific monitor/recipient pre-process and format functions.


Monitor - Properties

Phone# or ID of the Alarm transmitter

0641xxxxxx

If SMS messages are not originating from a mobile station, the originator phone# is not contained within the SMS data, in such situations, the message it self must start with an ID terminated with a #. Example message: SITE1#Temperature to low!

Monitor properties

Phone#

The voice phone number used by the SMS alarm transmitter.

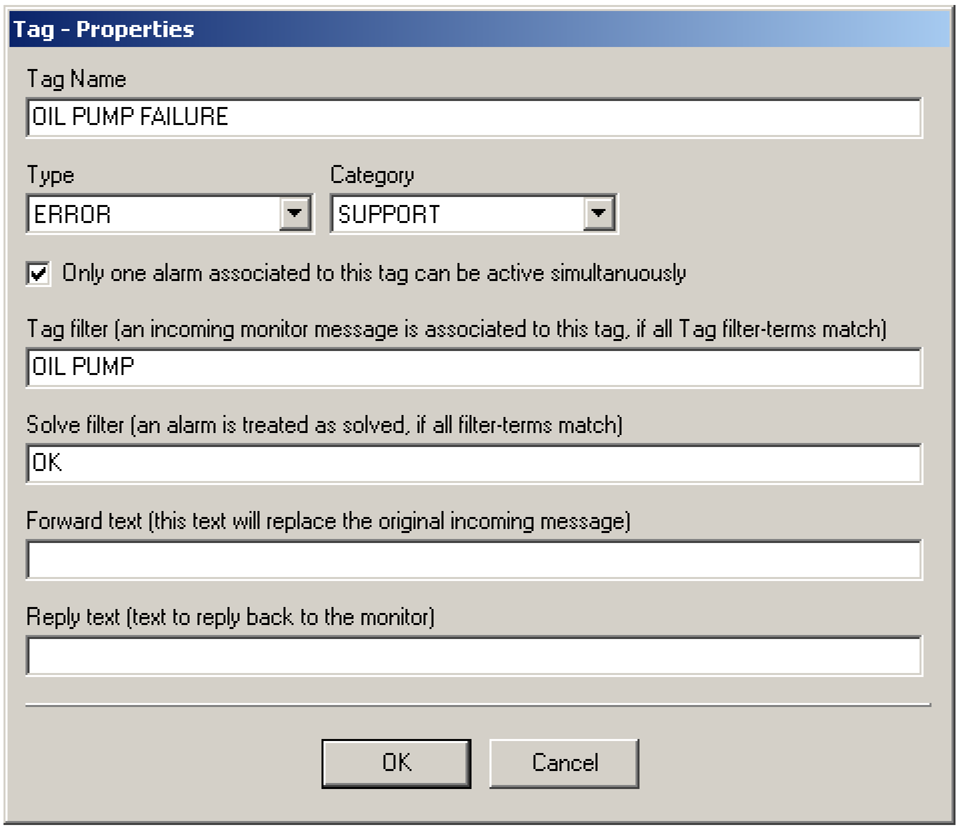
> **Note:** If the SMS alarm transmitter is using an analog phone connection, instead of a mobile phone network to transmit SMS messages, it could be that the phone# is not with the SMS-message. In such situations the message text itself should start with a phone# or any other ID-text separated from the actual message text by a non-alphanumeric symbol.
>
> **Example:** SITE1#Alarm, Temperature tank 1 to low!
>
> In this case you should enter SITE1 as the phone#

Pre-process Script

JScripting can be used to pre-process incoming messages. The ‘Pre-process Script’-page contains a checkbox to enable JScripting. It contains an edit window as well, that can be used to define a pre-process function, to transform the text of an incoming message into another.

Tag properties

A ‘tag’ describes a condition to distinguish between incoming messages that should be treated differently! Some messages could be warnings about a rising critical situation, while others are just hints. Tags cannot be defined, if the monitor is another ‘Messenger’.


Tag - Properties

Tag Name

OIL PUMP FAILURE

Type: ERROR

Category: SUPPORT

[✓] Only one alarm associated to this tag can be active simultanuously

Tag filter (an incoming monitor message is associated to this tag, if all Tag filter-terms match): OIL PUMP

Solve filter (an alarm is treated as solved, if all filter-terms match): OK

Forward text (this text will replace the original incoming message):

Reply text (text to reply back to the monitor):

A ‘Tag filter’ can be used to distinguish between different messages. A ‘Tag Filter’ may contain multiple fragments separated by spaces. Only messages containing the filter fragments (regardless of their order in the text) will be associated with this tag. If a fragment itself contains spaces, it can be enclosed within double quotes to prevent the fragment from being seen as multiple fragments.

Tags will be processed on their ‘Tag filters and fragments’ in descending alphabetical order, for instance to prevent a message containing ‘OIL PUMP’ to be associated with a tag filtering on ‘PUMP’ only, while there is a tag specifically filtering on ‘OIL PUMP’.

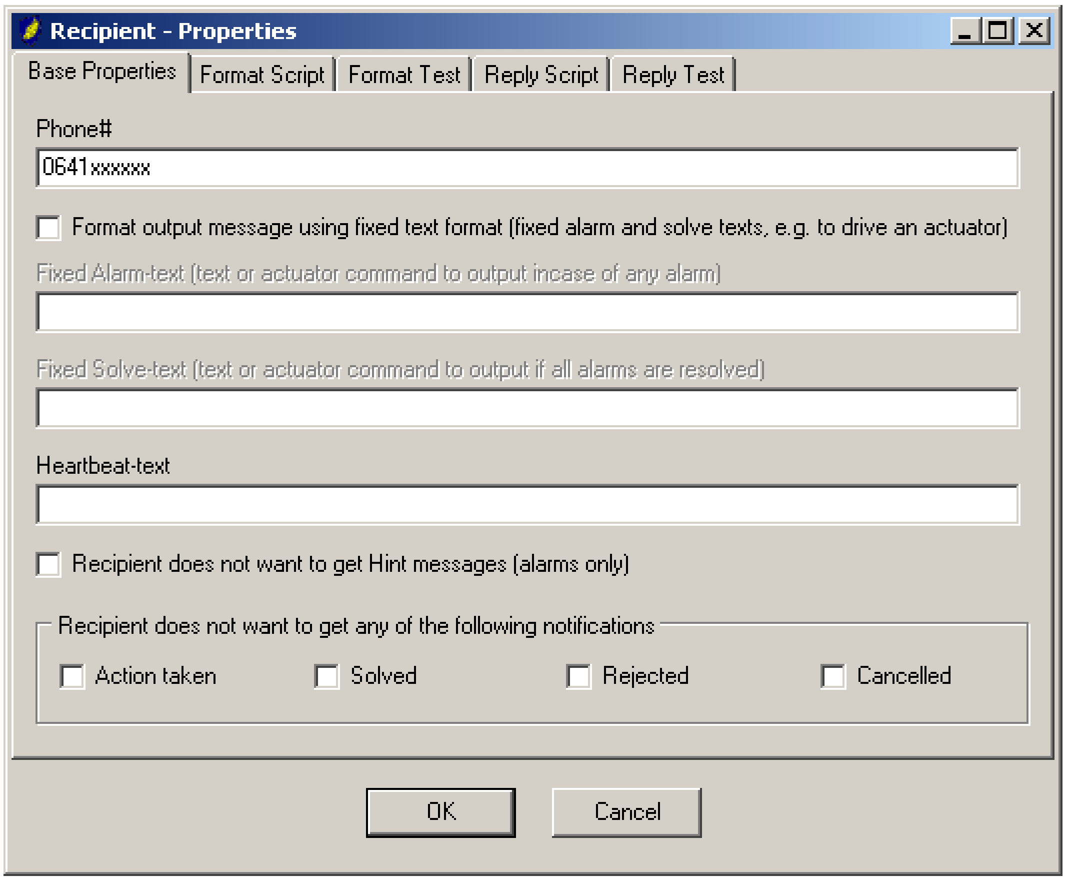
Messages that cannot be associated with a tag are treated as alarms of type ‘Error’ with category ‘Support’. To change this default behavior, you could at a tag with a blank ‘Tag filter’.

If the alarm transmitter is able to send solve messages, the ‘Messenger’ can recognize them, if you specify a ‘Solve filter’. The ‘Solve filter’ may contain, like the ‘Tag filter’, multiple fragments separated by spaces. Only messages matching both the ‘Tag filter’ and ‘Solve filter’ will be treated as a ‘Solve’ message.

If the message text is rather cryptic, it is possible to replace the original text with a ‘Forward text’.

A ‘Reply text’ can be specified if the alarm transmitter requires one.

Recipient properties


Recipient - Properties

Base Properties | Format Script | Format Test | Reply Script | Reply Test

Phone#

0641xxxxxx

☐ Format output message using fixed text format (fixed alarm and solve texts, e.g. to drive an actuator)

Fixed Alarm-text (text or actuator command to output incase of any alarm)

Fixed Solve-text (text or actuator command to output if all alarms are resolved)

Heartbeat-text

☐ Recipient does not want to get Hint messages (alarms only)

Recipient does not want to get any of the following notifications

☐ Action taken ☐ Solved ☐ Rejected ☐ Cancelled

OK Cancel

Phone#

The phone number of the recipient.

Fixed output message

Instead of transmitting the alarm message itself, it is possible to send fixed text messages, e.g. to signal a SMS driven actuator. When an alarm is dispatched to ‘this recipient’, the ‘Fixed alarm text’ will be send in stead and as soon as all alarms dispatched to ‘this recipient’ are solved, then the ‘Fixed solve text’ will be send to ‘this recipient’.

Heartbeat-text

If ‘this recipient’ is supposed to get heartbeat messages, the heartbeat text to send to ‘this recipient’ should be defined.

Format Script

JScripting can be used to (re) format outgoing messages. The ‘Format-Script’-page contains a checkbox to enable JScripting. It contains an edit window as well, that can be used to define a (re) format function, to format the text of the outgoing message.

Reply Script

JScripting can be used to pre-process incoming replies. The ‘Reply Script’-page contains a checkbox to enable JScripting. It contains an edit window as well, that can be used to define a pre-process function, to transform the text of an incoming reply into another.