

Operation Manager 0.27.0 (full version)

Technical documentation

Katarzyna Wladyszewska, Hadden Sp.J.

Operation Manager 0.27.0 (full version): Technical documentation

by Katarzyna Wladyszewska

Published April 2010

Copyright © 2003-2010 Hadden Sp.J.

HADDEN MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MANUAL, INCLUDING, BUT NOT LIMITED TO, THE WARRANTY OF DESIGN, MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

All rights reserved. No part of this document may be photocopied, reproduced or translated to another language without the prior written consent of Hadden Sp.J.

All trademarks included in this document are the property of their respective owners.

Table of Contents

1. Conventions	1
2. General information about David system	2
2.1. General	2
2.2. David system architecture	3
3. Terminology	6
3.1. Authorization process made by David system products	6
3.2. David system terminology used in the documentation	6
4. Installation	7
4.1. The main configuration file of David system	7
4.2. Dedicated account for service of David system	7
4.3. Directories of David system	8
4.4. Configuration of syslogd daemon	8
5. Operation Manager requirements	9
6. Installation	10
6.1. Installation from the RPM package	10
6.2. Installation from the script	10
7. General information	11
7.1. Functionality	11
7.2. Description	11
7.3. Related articles	11
8. Graphic Notifications Server (dgnsd)	13
8.1. General	13
8.2. Synopsis	13
8.3. Options	13
8.4. Description	14
8.5. Related articles	15
9. Sound Client (sndc)	16
9.1. General	16
9.2. Synopsis	16
9.3. Options	16
9.4. Description	16
9.5. Related articles	17
10. Access to the SNMP Trap Interface (damsnmpti)	18
10.1. General	18
10.2. Synopsis	18
10.3. Options	18
10.4. Description	19
10.5. Related articles	19
11. SNMP Trap Analyser (damsnmptaud)	20
11.1. General	20

11.2. Synopsis	20
11.3. Options	20
11.4. Configuration file format	21
11.5. Description	22
11.6. Related articles	22
12. Access to the SNMP Data Interface (damsnmpdi)	23
12.1. General	23
12.2. Synopsis	23
12.3. Options	23
12.4. Description	24
12.5. Related articles	24
13. SNMP Data Analyser (damsnmpdaud)	25
13.1. General	25
13.2. Synopsis	25
13.3. Options	25
13.4. Configuration file format	27
13.5. File format of minimal levels definitions	28
13.6. Description	28
13.7. Related articles	31
14. Cases Database Unit (damadbud)	32
14.1. General	32
14.2. Synopsis	32
14.3. Options	32
14.4. Configuration file format	33
14.5. Description	34
14.6. Related articles	34
15. Cases Service Unit (damcsud)	36
15.1. General	36
15.2. Synopsis	36
15.3. Options	36
15.4. Configuration file format	37
15.5. Description	38
15.6. Running actions (programs)	39
15.7. Input parameters passed on running actions (programs)	39
15.8. Data return by running actions (programs)	41
15.9. Destroying obsolete cases	42
15.10. Related articles	42
16. Buttons the most often used in Web applications	43
16.1. The buttons meaning	43
17. Recorded Operation Browser	45
17.1. General	45
17.2. Description	45
17.2.1. Specyfication of searching criterions	45
17.2.2. Generated report	47
17.3. Related articles	48

18. Pending Operation Browser	50
18.1. General	50
18.2. Description	50
18.2.1. The view of active cases	50
18.2.2. The view of collected SNMP Data	53
18.3. Related articles	57
19. Graphic Notifications Presenter (xdgnp)	58
19.1. General	58
19.2. Synopsis	58
19.3. Options	58
19.4. Description	58
19.4.1. Starting up and terminating the application	58
19.4.2. Main view work	59
19.4.3. List of logged in users on dgnsd server	61
19.4.4. xdgnp configuration	63
19.5. Related articles	66

List of Tables

1.1. The typographical conventions used in this manual	1
2.1. David system products	3
8.1. dgnsd options	13
9.1. sndc options	16
10.1. damsnmpti options	18
11.1. damsnmptaud options	20
12.1. damsnmpdi options	23
13.1. damsnmpdaud options	25
14.1. damadbud options	32
15.1. damcsud options	36
15.2. Input parameters passed on running actions (programs)	39
15.3. The list of arguments	40
15.4. Arguments for snmptrap message	40
15.5. Arguments for snmpdata message	41
16.1. The buttons the most often used in Web applications	43
17.1. Recorded Operation Browser - fields description of Cases group	46
18.1. Pending Operation Browser - meaning fields of Collected data group	54
19.1. xdgnp options	58
19.2. xdgnp - modes of work	59
19.3. xdgnp - columns description	60
19.4. xdgnp - description of the buttons	61
19.5. The buttons correspond to Edit and View menu	61
19.6. dgnsd server - description of the dialog buttons	62
19.7. Columns description	62

Chapter 1. Conventions

The following typographical conventions are used in this manual:

Table 1.1. The typographical conventions used in this manual

Font	What the font represents	Example
<i>Italic</i>	Environment variables.	The name is kept in environmental variable <i>\$DAVIDPRIVDIR...</i>
<i>Italic</i>	Synopsis options.	<i>[-l,--log-facility log_facility]</i>
Bold	Names of programs and products.	damcsud is a part of Operation Manager-a .
Computer	Names of options and menus.	There is Show tool bar option in View menu.
Computer	Names of files and directories.	... reads its configuration file <code>.damadbudrc</code> .
Computer	Names of windows and dialog fields.	In A sessions property window, in Sticking string field, you can write...
Computer	Names of buttons.	Pressing Apply button lets you apply changes.
Computer Bold	Math formulas.	$\exp(-x)$, when $a = 0$ $1 / \text{pow}(a, a) * \text{pow}(x, a) * \exp(-x + a)$, when $a > 0$.
Computer Bold	Terms used in David system terminology.	SNMP Data - a kind of data...
Computer Bold	Contents of configurations files.	action { ... }

Chapter 2. General information about David system

2.1. General

David system is a network management system. It is a packet of applications (modules) that allows computer network to be monitored and managed in real-time through the Internet. There is only one condition that managed devices must meet. Each device must provide SNMP (Simple Network Management Protocol) service. SNMP is the most common management protocol in the Internet so that requirement shouldn't be difficult to meet. Here is the list of typical devices that can be monitored:

- IP routers,
- ATM switches,
- manageable ethernet switches,
- UPSes with a SNMP adapter,
- TV-SAT modems that allow IP devices to work in TV cable networks,
- computers.

One of the most important feature of **David system** is its architecture. It's built of high level configurable and independent from one another modules. This principle is the most essential rule of the project. In consequences, in th metter of speaking, the same modules may build different management system. Here are the main features of **David system**:

- general thinking in information flow controlling that come form high level independence of modules of the system,
- high level configureability of the system modules that allows a special configuration of **David system** to reach end-user expectations so close as it's only possible,
- the system scalability, so you can build up the system adding additional modules in very easy way; note that these modules needn't to be part of **David system** at all; adding another monitored devices to the system is a very easy procedure,
- using shell scripts in information processing is opportunity for modeling information and influence on processing it,
- all configuration files of **David system**, files with input/output data and log files are text files,

- using SNMPv1, SNMPv2C and SNMPv3 to communicate with monitored devices.

2.2. David system architecture

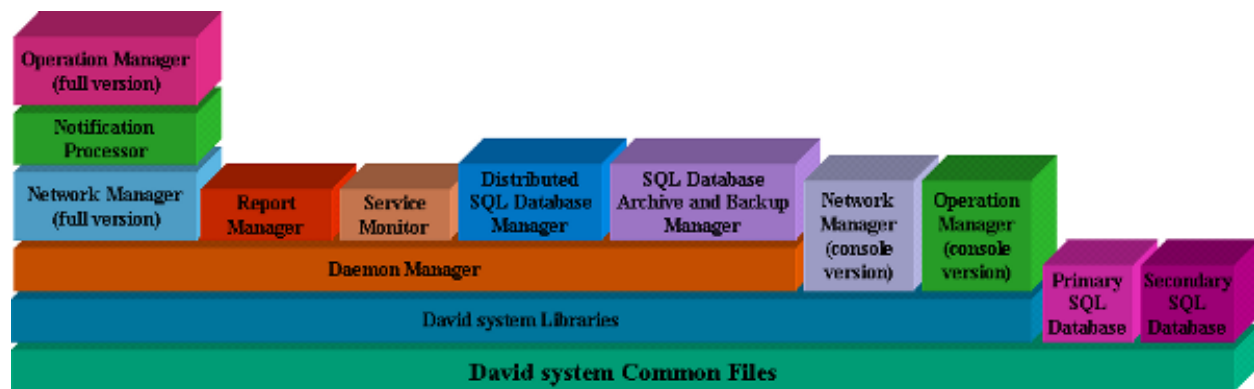
Table 2.1. David system products

Product	Description
David system Common Files	The product, during its installation, prepares the rudimentary directory tree for other products of David system . It also contains some essential and common files for all the products. Thus, this is a fundamental product of David system required by other its products.
Primary SQL Database	The product installs the primary SQL database of David system . Every single installation of David system must have only one the primary database.
Secondary SQL Database	The product installs the secondary SQL database of David system . Each installation of David system may have many secondary databases or none. It allows to distribute the SQL database of David system among many servers.
David system Libraries	This product provides libraries of David system required by its applications. Many other products of David system require that one.
Daemon Manager	It engages in running and terminating daemons of David system as well as monitoring of their work.
Network Manager (full version)	The product using SNMP protocol allows to visualise a topology of monitored networks and auto-discover devices in managed networks. The state of monitored devices also is visualized. The product also collects data from monitored devices using SNMP protocol and allows you to manage user accounts.
Network Manager (console version)	The product, through a graphic application, allows to visualize a topology of monitored networks and shows states of monitored resources. It allows you to control daemons monitoring devices as well as that ones gathering data. Currently, most of functions of that application is obtainable through web applications.
Notification Processor	The product chiefly engages in processing SNMP Trap notifications coming from monitored devices to management stations. The received messages can be formatted to the human readable forms, and then recorded as well. The processed notifications can be passed on to future processing.
Operation Manager (full version)	It can run specified actions on the basis of received data. Sophisticated estimation process depends on information coming from other products of David system and correlation of that information. It tries to build more intelligent and useful notifications then just simple reactions to incoming

General information about David system

Product	Description
	events. The graphic application displays notifications about received events and allows to play audio files as well as reading messages by an outer speech synthesizer.
Operation Manager (console version)	The product contains a graphic application displaying notifications about events and allowing to play audio files as well as reading messages by an outer speech synthesizer.
Report Manager	The product processes recorded SNMP Trap notifications, entries about pending operations and entries about state changes of monitored devices (ping objects, network interfaces and BGP peers), and generates reports on the basis of them. Reports can be viewed using a Web application.
Service Monitor	The product monitors selected network services on application level. In order to do this it monitors selected TCP ports of specified hosts. It checks both availability of ports and a correct reaction for a few selected network protocols (HTTP, SMTP, FTP). It also can verify correctness of work of selected services by verification of received data. Results of its work can be viewed as reports and graphs made available by a Web application.
SQL Database Archive and Backup Manager	It archives the SQL Database used by David system applications.
Distributed SQL Database Manager	It allows to divide the database of David system into one primary database and many secondary ones. Such step boosts performance of the system and decreases load of the servers where daemons of David system work. The migration takes place during the routine work of the system. Such division may be altered many times.

Dependencies between the **David system** products are shown on the following chart..



David system functionality can be very large and it depends on particular configuration a lot. The most important features of **David system** are:

- discovering and visualization of monitored networks topology including visualization of states of

particular nodes;

- possibility of building control panels to monitored devices (they must support SNMP protocol), regardless of device providers;
- formatting and recording SNMP Traps sent by agents working on monitored devices;
- automatic reaction to specified SNMP Traps received from monitored devices;
- possibility of identification of an operator that has received an alert from the system about a problem;
- collecting data concerning parameters of monitored devices;
- automatic reaction to incorrect values of data that were found during data collecting;
- recording pending cases, processed by the system, which have been created as responses for events detected by the system in a monitored network;
- monitoring selected network services on application level.

Chapter 3. Terminology

3.1. Authorization process made by David system products

The modules of David system which need to do an authorization of message senders (i.e. **damsnmpdaud**, **dnmmsd**, **dgnsd**), use the library, that checks whether an IP address of a sender matches with any record found in the file `.known.host`. The library expects to find the file in a directory pointed by a variable `confdir` in the file `/etc/system-david.conf`.

Records in the file `.known.host` are regular expressions specifying acceptable IP addresses.

3.2. David system terminology used in the documentation

There is an explanation of some terms, that are used in David system and its documentation:

- **messages (information)** - data received by interfaces of **Operation Manager**, its data analysers and **Cases Database Unit** of the product.
- **notifications** - the term often is used in the products: **Notification Processor**, **Operation Manager** and **Report Manager**; There are mostly data, that a source are SNMP agents working on network monitored devices.
- **events** - the term often is used in the products: **Operation Manager** and **Report Manager**; and it describes a being, that a source is SNMP Trap or SNMP Data; an **event** is always a part of a **case**;
- **cases** - the term often is used in the products: **Operation Manager** and **Report Manager**; and it describes a group of events connected one another; one **event** at last must be included in a **case**;
- **SNMP Trap** - a kind of data of **Operation Manager** product, which a source are received responses from SNMP agents; SNMP Traps aren't answers on the requests sent by a management station, but they are sent by agents managing network interfaces and processed by **Notification Processor** product;
- **SNMP Data** - a kind of data of **Operation Manager** product, which a source are received responses from SNMP agents on request which a management station sent to them by **Network Manager**.

Chapter 4. Installation

4.1. The main configuration file of David system

The essential configuration file of David system is `/etc/david-system.conf`. It contains entries as pairs: `key = value`. Basically, except the entry `default_email_recipient`, there is no such need to modify any record in that file. All necessary modifications are made during installation processes of particular David system products. Below, there is a list of all entries along with their descriptions that may occur in this basic configuration file.

- `user` - a name of the user with which rights all daemons of David system works;
- `default_email_recipient` - the default e-mail address where messages from David system applications are sent;
- `bindir` - the directory containing David system applications (default: `/usr/bin/david-system`);
- `libdir` - the directory containing David system libraries (default: `/usr/lib/david-system`);
- `incdir` - the directory containing David system headers (default: `/usr/include/david`);
- `confdir` - the directory containing David system configuration files (default: `/etc/david-system`);
- `logdir` - the directory containing log files of David system applications (default: `/var/log/david-system`);
- `sharedir` - the directory containing various files (images, audio files, web files) of David system (default: `/usr/share/david-system`);
- `docdir` - the directory containing various files (images, audio files, web files) of David system (default: `/usr/share/david-system`);
- `vardir` - the directory containing archive files of David system SQL database (default: `/var/lib/david-system`);
- `is_sqldb_installed` - the flag that indicates whether the SQL database of David system has been installed or not.

4.2. Dedicated account for service of David system

There is no need to run any David system module as superuser (usually an account `root` with UID equals 0). Even if some David system daemon requires root rights when starting, there is always possibility to specify, as one of the daemons starting arguments, a user that rights should be taken.

It is a good idea to add a new user to an operating system, under which control David system will work.

4.3. Directories of David system

This hierarchy depends on a particular configuration of David system. In the default system configuration, David system contains the following directories:

- `/usr/bin/david-system` - binaries and shell scripts;
- `/etc/david-system` - configuration files;
- `/usr/share/doc/david-system` - the documentation;
- `/usr/share/david-system` - graphic and audio files, web portal;
- `/usr/include/david` - David system header files;
- `/usr/lib/david-system` - David system libraries;
- `/var/log/david-system` - log files;
- `/var/lib/david-system` - archive files of the David system SQL database;

4.4. Configuration of syslogd daemon

David system modules use `syslog` subsystem available on UNIX platforms. Default configuration of the system modules causes that log messages are sent with `local6` facility. It may be changed for every module during its startup. Its recommended to configure `syslogd` daemon to write all messages from David system modules into one place (one or more files with characteristic name i.e.: `david.log`).

Chapter 5. Operation Manager requirements

The following requirements must be met by a management platform on which **Operation Manager** will work:

- installed, compatible version of **Notification Processor**.

Chapter 6. Installation

6.1. Installation from the RPM package

You must be `root` to install the product. The typical installation looks as this one following below:

- Install the product:

```
rpm -i david-xxx-om-f-yyy.rpm
```

6.2. Installation from the script

You must be `root` to install the product. The typical installation looks as this one following below:

- Uncompress and unpack the archive:

```
gunzip david-xxx-om-f-yyy.i386.tar.gz  
tar xf david-xxx-om-f-yyy.i386.tar
```

The operations create `david-xxx-om-f-yyy.i386` directory in your current directory.

- Change your current directory to `david-xxx-om-f-yyy.i386`:

```
cd david-xxx-om-f-yyy.i386
```

- Read `LICENSE` file from the current directory and **CONTINUE THE INSTALLATION, ONLY WHEN YOU ACCEPT ALL CONDITIONS INCLUDED IN THE LICENSE.**
- Run the installation script:

```
./install
```

Chapter 7. General information

7.1. Functionality

Operation Manager makes possible:

- correlation of information received from **Notification Processor** and **Network Manager**;
- grouping received information into current cases with assigned identifiers, that are unique in the system;
- running actions (specified programs) for processed cases taking into consideration correlation of events included in their case, results of finished actions and time of its last occurrence;
- processing events received from processes that aren't David system modules; it allows to respond to events not originated from SNMP agent;
- getting information about current processed cases, failures and their service;
- learning to recognize potentially dangerous events for monitored networks;
- notifying an operator about received messages, pending cases, and recording user's identifier and his/her reaction time.

7.2. Description

Operation Manager is an important part of the whole system. Other David system products influence on its work but it also influences on their work making some decisions. Using special algorithms, it tries to estimate severity of received information, and also finds connections between received messages. Information received by the product, can be different types (e.g.: data received through SNMP protocol and SNMP-Trap messages).

One of main **Operation Manager** goal is running specified programs or scripts as a response to a current state of monitored devices and also notifying an operator, through e-mails, graphic windows and sound signals, about processed events and pending cases (i.e.: power failures, connectivity breaks between devices, device restarts etc.).

Results of **Operation Manager** work can be viewed through Web applications which also allow to control the product work.

7.3. Related articles

[Access to the SNMP Trap Interface \(damsnmpti\)](#)

[SNMP Trap Analyser \(damsnmptaud\)](#)

[Access to the SNMP Data Interface \(damsnmpdi\)](#)

[SNMP Data Analyser \(damsnmpdaud\)](#)

[Cases Database Unit \(damadbud\)](#)

[Cases Service Unit \(damcsud\)](#)

[Pending Operation Browser](#)

[Recorded Operation Browser](#)

[Graphic Notifications Server \(dgnsd\)](#)

[Graphic Notifications Presenter \(xdgnp\)](#)

Chapter 8. Graphic Notifications Server (dgnsd)

8.1. General

dgnsd is **Graphic Notifications Server** and it is a part of **Operation Manager**. It is a daemon process which works all the time the system is running and it expects requests from clients, which want to display a notification about a running case as a graphic window or to play a sound file or read a text. Next, such notifications are forwarded to registered graphic clients, i.e. the clients that can display notifications as graphic windows or play sound. Responses received from graphic clients are forwarded to a client that sent the notification. A typical client that sends such notifications is **dsi** program, and a typical client, that requests about playing sound is **sndc** program. A graphic client, that can display graphic windows with such notifications and can play sounds is **xdgncp** application.

8.2. Synopsis

dgnsd can be run with the following options: `[-p,--pid-file filename]` `[-l,--log-facility log_facility]` `[-L,--log-level log_level]` `[--nr-of-gc nr]` `[--nr-of-ic nr]` `[--socket-file-4-gc filename]` `[--socket-file-4-ic filename]` `[--port-4-gc port]` `[--port-4-ic port]` `[--ttl seconds]` `[-u,--run-as-user username]` `[--background]` `[-v,--version]` `[-h,--help]`

8.3. Options

Table 8.1. dgnsd options

Option name	Description
<code>-p,--pid-file filename</code>	Write PID to a specified file.
<code>-l,--log-facility log_facility</code>	Choose log facility: daemon user local0 ... local7 (default: local6).
<code>-L,--log-level log_level</code>	Choose log level (on stderr and syslog) i.e. messages of selected level and more important levels will be logged: emerg alert crit err warning notice info debug0 ... debug2 (default: notice).
<code>--nr-of-gc nr</code>	Specify a maximum number of graphic clients that may be served at the same time (default: 10).
<code>--nr-of-ic nr</code>	Specify a maximum number of notification clients that may be served at the same time (default: 20).
<code>--socket-file-4-gc filename</code>	Wait for connections from graphic clients via a specified socket file (default: /tmp/dgns.gc.socket).
<code>--socket-file-4-ic filename</code>	Wait for connections from notifications clients via a specified socket file (default: /tmp/dgns.ic.socket).
<code>--port-4-gc port</code>	Listen to a specified TCP port waiting for graphic clients (default: use a

Option name	Description
	socket file).
<i>--port-4-ic port</i>	Listen to a specified TCP port waiting for notifications clients (default: use a socket file).
<i>--ttl seconds</i>	Specify a maximum TTL (time to live) for outgoing messages (default: 300).
<i>-u,--run-as-user username</i>	Drop root privileges and run server as a specified user.
<i>--background</i>	Go to background after startup.
<i>-v,--version</i>	Display version number on stderr and exit.
<i>-h,--help</i>	Display this help and exit.

8.4. Description

dgnsd is a single-process server. Its clients can be both programs that want to notify an operator about a running problem and programs which can display a notification as a graphic window. Once the server receives from a client a notification about a case, next it is forwarded to registered graphic clients. The message is forwarded only to these clients which have logged in users. So, if a graphic application wants to receive notifications, it must be connected to the server and must pass a successful authentication of its user in David system.

The authentication is made on the base of information contained in **Users' Database** which is stored in SQL Database. A response received from a graphic client is forwarded to a notification client that sent the notification. Because of fact that more then one graphic client may send a response, a simple rule is applied: only that response is respected which comes from a client holding the token. There is always one token in **dgnsd** server. When no user holds the token, the first response matters. A given graphic client may log in many times and every time it may be a different user. A logged in user has rights according to the **Users' Database**. The users may do the following operations:

- to log out;
- to log out users of a lower level;
- to list logged in users;
- to take over the token, if no user has it or its owner is a lower level user;
- to leave the token;
- to pass the token to a lower level user on condition that the token is hold by the user or it is nobody or it belongs to a lower level user.

dgnsd server also gets from [sndc](#) program requests about playing sounds and forwards them to clients [xdgnp](#). There are two kinds of requests:

- a request about reading a text;
- a request about playing a sound file.

The text messages are played with the speech synthesizer's help. When a sound file is played, you should give a path to it. Each path is automatically preceded with a directory in which David system is locally installed (i.e.: /home/david).

The client [xdgnp](#) can be installed on the other computer station then **dgnsd** server. When [xdgnp](#) client receives a request about playing a sound file, it checks that the file is placed in its local filesystem or not. If the program doesn't find the file, it can take it from the station, on which **dgnsd** server is working. All sound files are placed on the computer station with installed **dgnsd** server.

8.5. Related articles

[Graphic Notifications Presenter \(xdgnp\)](#)

[Sound Client \(sndc\)](#)

Notification Processor: Information Recorder (dsi)

Network Manager: User Manager

Chapter 9. Sound Client (sndc)

9.1. General

Program **sndc** is **Sound Client** and it's a part of **Operation Manager**. Its main assignment is to send to [dgnsd](#) server requests about playing sound.

9.2. Synopsis

sndc can be run with the following options: `[-l,--log-facility log_facility] [-L,--log-level log_level] [--socket-file filename] [--host host] [--port port] [--ttl seconds] [--text text] [--file filename] [-v,--version] [-h,--help]`

9.3. Options

Table 9.1. sndc options

Option name	Description
<code>-l,--log-facility log_facility</code>	Choose log facility: daemon user local0 ... local7 (domyślnie: local6).
<code>-L,--log-level log_level</code>	Choose log level (on stderr and syslog) i.e. messages of selected level and more important levels will be logged: emerg alert crit err warning notice info debug0 ... debug2 (default: notice).
<code>--socket-file filename</code>	Connect to the server using a specified socket file (default: /tmp/dgns.ic.socket).
<code>--host host</code>	Connect to the server on a specified host (default: use socketfile).
<code>--port port</code>	Connect to the server using a specified TCP port (default: use socketfile).
<code>--ttl seconds</code>	Specify maximum time in seconds to wait for completion of the connection (default: 60).
<code>--text text</code>	Ask the server for reading this text by its graphic clients.
<code>--file filename</code>	Ask the server for playing this file by its graphic clients.
<code>-v,--version</code>	Display version number on stderr and exit.
<code>-h,--help</code>	Display this help and exit.

9.4. Description

Program **sndc** sends to [dgnsd](#) server requests about playing sound. Server [dgnsd](#) sends the request to one's graphic clients, which are [xdgmp](#) applications. Messages, that are sent by **sndc** can be double:

- a request about reading a text;
- a request about playing a sound file.

Text is played using the speech synthesizer. When a sound file is played, you should give a path to it. Each path is automatically preceded with a directory where sound files of David system are located (default: `/usr/share/david-system/sounds`). Thus giving the file `alarm.wav` causes the graphic client understands that as a file `/usr/share/david-system/sounds/alarm.wav`.

9.5. Related articles

[Graphic Notifications Server \(dgnsd\)](#)

[Graphic Notifications Presenter \(xdgnp\)](#)

Chapter 10. Access to the SNMP Trap Interface (damsnmpti)

10.1. General

damsnmpti is **Access to the SNMP Trap Interface** and it is a part of **Operation Manager**.

10.2. Synopsis

damsnmpti can be run with the following options:: [*-f,--server-socket-file socket_file*] [*-H,--server-host host*] [*-P,--port-on-host port*] [*-l,--log-facility log_facility*] [*-L,--log-level log_level*] [*-i,--trap-severity level*] [*-s,--trap-source IP_adres*] [*-a,--trap-agent IP_address*] [*-c,--trap-community community*] [*-n,--trap-number information_nr_or_oid*] [*-e,--trap-enterprise enterprise*] [*-t,--trap-time-stamp time-stamp*] [*-g,--trap-generic generic_type*] [*-p,--trap-specific specific_type*] [*-d,--trap-id string*] [*-C,--community community*] [*-m,--trap-msg message*] [*-v,--version*] [*-h,--help*]

10.3. Options

Table 10.1. damsnmpti options

Option name	Description
<i>-f,--server-socket-file socket_file</i>	Connect with the server via a specified socket file (default: /tmp/damsnmptid.socket).
<i>-H,--server-host host</i>	Connect with the server on a specified host (default: connect via a socket file: /tmp/damsnmptid.socket).
<i>-P,--port-on-host port</i>	Connect with the server on a specified host using a specified port (default: port 6688 on UDP).
<i>-l,--log-facility log_facility</i>	Choose log facility: daemon user local0 ... local7 (default: local6).
<i>-L,--log-level log_level</i>	Choose log level (on stderr and syslog) i.e. messages of selected level and more important levels will be logged: emerg alert crit err warning notice info debug0 ... debug2 (default: warning).
<i>-i,--trap-severity level</i>	The severity level of this message expressed as a double type number inside range <-1.0;1.0> ((default: 0.0).
<i>-s,--trap-source IP_adres</i>	An IP address of a sender of the notification that will be forwarded to Interfejsowi Danych SNMP Trap .
<i>-a,--trap-agent IP_address</i>	An IP address of an agent which sent this notification.
<i>-c,--trap-community</i>	SNMP community of this notification.

Option name	Description
<i>community</i>	
<i>-n,--trap-number information_nr_or_oid</i>	An identifier (OID) of this notification.
<i>-e,--trap-enterprise enterprise</i>	An enterprise identifier (OID) of this notification.
<i>-t,--trap-time-stamp time-stamp</i>	A time-stamp of this notification.
<i>-g,--trap-generic generic_type</i>	A generic type of this notification.
<i>-p,--trap-specific specific_type</i>	A specific type of this notification.
<i>-d,--trap-id string</i>	An unique ID string that identifies this message in Operation Manager .
<i>-C,--community community</i>	A community assigned to this variable (communities must be separated by ':' character).
<i>-m,--trap-msg message</i>	A human readable string that describes this message.
<i>-v,--version</i>	Display version number on stderr and exit.
<i>-h,--help</i>	Display this help and exit.

10.4. Description

damsnmpti program allows access to [SNMP Trap Interface](#) from shell level. The program serves as a bridge to [SNMP Trap Interface](#). All information about single data (variable) **damsnmpti** receives as input parameters and it forwards them to the server which is [SNMP Trap Interface \(damsnmptaud\)](#).

10.5. Related articles

[SNMP Trap Analyser \(damsnmptaud\)](#)

Chapter 11. SNMP Trap Analyser (damsnmptaud)

11.1. General

damsnmptaud is **SNMP Trap Analyser** and it is a part of **Operation Manager**. It's also **SNMP Trap Interface**. It is a daemon process which works all the time the system is running and it expects SNMP Trap type data (these messages are not TRAP-PDU type notifications as in SNMP protocol). **damsnmptaud** calculates severity of each received message on the basis of its configuration file and then it forwards them. A typical receiver of processed messages is **Associations Database**.

damsnmptaud reads its configuration file `.damsnmptaudrc` during its startup. The program expects to find the configuration file in the directory `/etc/david-system`. **damsnmptaud** daemon won't be run if it can't read its configuration file.

11.2. Synopsis

damsnmptaud can be run with the following options: `[-P,--pid-file filename]` `[-l,--log-facility log_facility]` `[-L,--log-level log_level]` `[-f,--in-socketfile filename]` `[-p,--in-port unsignedtype_number]` `[--out-socketfile filename]` `[--out-port unsignedtype_number]` `[--out-host hostname]` `[-u,--run-as-user username]` `[--severity-balance-factor doubletype_number]` `[--background]` `[--ignore-case-4-severity]` `[-v,--version]` `[-h,--help]`

11.3. Options

Table 11.1. **damsnmptaud** options

Option name	Description
<code>-P,--pid-file filename</code>	Write PID to the specified file.
<code>-l,--log-facility log_facility</code>	Choose log facility: daemon user local0 ... local7 (default: local6).
<code>-L,--log-level log_level</code>	Choose log level (on stderr and syslog) i.e. messages of selected level and more important levels will be logged: emerg alert crit err warning notice info debug0 ... debug2 (default: warning).
<code>-f,--in-socketfile filename</code>	Use a specified socket file for listening (default: <code>/tmp/damsnmptaud.socket</code>).
<code>-p,--in-port unsignedtype_number</code>	Use a specified UDP port number for listening instead of a socket file (default a socket file: <code>/tmp/damsnmptaud.socket</code>).
<code>--out-socketfile filename</code>	Use a specified socket file for output (default: <code>/tmp/damadbud.socket</code>).

Option name	Description
<i>--out-port unsignedtype_number</i>	Send outgoing UDP packets to this port (default: use a socket file '/tmp/damadbud.socket').
<i>--out-host hostname</i>	Send outgoing UDP packets to that host (default: 127.0.0.1).
<i>-u,--run-as-user username</i>	Drop root privileges and run server as a specified user.
<i>--severity-balance-factor doubletype_number</i>	Use a specified severity factor (range <0.0;1.0>) to calculate the final severity of processed message: <code>incoming_severity * this_parameter -+ severity_taken_from_configuration * (1.0-this_parameter)</code> (default: 0,5).
<i>--background</i>	Go to background after startup.
<i>--ignore-case-4-severity</i>	Calculate a processed message severity ignoring a case.
<i>-v,--version</i>	Display version number on stderr and exit.
<i>-h,--help</i>	Display this help and exit.

11.4. Configuration file format

A configuration file consists of objects of `group` type. A definition of a single object is limited by braces `{}`. A `group` keyword that describes type of object must be a single word in a whole line. The same rule is applicable to braces `{}` that begin and end an object definition. Definition of a generic object is shown below:

```
group
```

```
{
```

```
...
```

```
}
```

An object is built from unlimited number of pairs of items. Each item takes two lines of the file. The first line begins from `string` keyword and specifies a string and the second one begins from `value` keyword - item severity. Following examples show how definitions of a `group` object may look like:

```
string=222.222.222.222
```

```
value=0.53
```

```
string=A realy bad news
```

```
value=0.99
```

Numbers describing values are treated as float numbers inside range <0.0;1.0>.

11.5. Description

When the agent, which waits for messages from network clients, receives an UDP packet then checks its IP address. In this case `.known.hosts` file is processed. If `.known.hosts` file can't be opened to read or an IP address of the client doesn't match any record in this file, the UDP packet is dropped. No authorization is made for local clients.

When an accepted message is processed, its severity is calculated, and then the message is forwarded. A typical receiver of processed messages by **damsnmptaud** is **Associations Database**.

The procedure that estimates a severity of a message takes only an identifier of the message into consideration (**damsnmpti** receives that identifier as an argument of its `-d` option). The identifier is unique in all **Operation Manager**. An initial severity of a message is zero. Then for each object of **group** type from the configuration file is checked whether any string after `string=` keyword of this object is included in an identifier of the message. If yes, a maximum value assigned to `string` item will be taken into consideration. This value is added to a current severity of the message. Finally the calculated severity is divided by a number of objects of `group` type defined in the configuration file. The final severity is calculated as follows:

$$s * f - S * (1 - f), \text{ gdy } s < 0$$

or

$$s * f + S * (1 - f), \text{ gdy } s \geq 0$$

where s is severity of a message included in it through its previous processing by other modules of David system. Parameter S is current severity of this message calculated by **damsnmptaud** in the basis of the configuration file. Factor f is an argument of `--severity-balance-factor` option of **damsnmptaud**.

11.6. Related articles

[Access to the SNMP Trap Interface \(damsnmpti\)](#)

[Cases Database Unit \(damadbud\)](#)

Chapter 12. Access to the SNMP Data Interface (damsnmpdi)

12.1. General

damsnmpdi is **Access to the SNMP Data Interface** and it is a part of **Operation Manager**.

12.2. Synopsis

damsnmpdi can be run with the following options: [*-f,--server-socket-file socket_file*] [*-H,--server-host host*] [*-P,--port-on-host port*] [*-l,--log-facility log_facility*] [*-L,--log-level log_level*] [*-a,--data-address IP_address*] [*-o,--data-oid oid*] [*-t,--data-type type*] [*-V,--data-value doubletype_number*] [*-I,--data-id string*] [*-C,--community community*] [*-m,--data-msg message*] [*-p,--data-sender-pid pid*] [*-v,--version*] [*-h,--help*]

12.3. Options

Table 12.1. damsnpdi options

Option name	Description
<i>-f,--server-socket-file socket_file</i>	Connect with the server via a specified socket file (default: /tmp/damsnmpdid.socket).
<i>-H,--server-host host</i>	Connect with the server on a specified host (default: connect via a socket file: /tmp/damsnmpdid.socket).
<i>-P,--port-on-host port</i>	Connect with the server on a specified host using a specified port (default: port 6699 on UDP).
<i>-l,--log-facility log_facility</i>	Choose log facility: daemon user local0 ... local7 (default: local6).
<i>-L,--log-level log_level</i>	Choose log level (on stderr and syslog) i.e. messages of selected level and more important levels will be logged: emerg alert crit err warning notice info debug0 ... debug2 (default: warning).
<i>-a,--data-address IP_address</i>	An IP address of a host that data comes from and it will be forwarded to SNMP Data Interface .
<i>-o,--data-oid oid</i>	An identifier OID of data that will be forwarded to SNMP Data Interface .
<i>-t,--data-type type</i>	Type of data that will be forwarded to SNMP Data Interface . The following types are possible: counter - a value may be only increased (32 bits); gauge - a value may be increased and decreased (32 bits); error - a value with an error; timeout - no answer for this variable; other - a value of other type

Option name	Description
	(IPADDRESS, OCTET, etc.).
<i>-V,--data-value doubletype_number</i>	A value of this data as double precision floating type.
<i>-I,--data-id string</i>	An unique ID string which identifies that data in Operation Manager .
<i>-C,--community community</i>	A community assigned to this variable (communities must be separated by ':' character).
<i>-m,--data-msg message</i>	The human readable message that describes meaning of this data.
<i>-p,--data-sender-pid pid</i>	PID (identifier) of a sender of this message.
<i>-v,--version</i>	Display version number on stderr and exit.
<i>-h,--help</i>	Display this help and exit.

12.4. Description

damsnmpdi allows access to [SNMP Data Interface](#) from shell level. The program serves as a bridge to [SNMP Data Interface](#). All information about single data (variable) the program receives as input parameters and it forwards them to the server which is [SNMP Data Interface \(damsnmpdaud\)](#).

12.5. Related articles

[SNMP Data Analyser \(damsnmpdaud\)](#)

Chapter 13. SNMP Data Analyser (damsnmpdaud)

13.1. General

damsnmpdaud is **SNMP Data Analyser** and it is a part of **Operation Manager**. It also performs the function of **SNMP Data Interface**. It is a daemon process which works all the time the system is running and it expects items of SNMP Data type. Data collected in the system are sent to **damsnmpdaud** and it tries to find these values which are out of a window of average values for that particular data and then it forwards them. A typical receiver of processed items is **Associations Database**.

damsnmpdaud reads its configuration file `.damsnmpdaudrc` during its startup. The program expects to find the configuration file in the directory `/etc/david-system`. **damsnmpdaud** daemon won't be run if it can't read its configuration file.

Next, **damsnmpdaud** reads the file with defined minimum levels. If the levels are exceeded, notifications will be sent to **Associations Database**.

13.2. Synopsis

damsnmpdaud can be run with the following options: `[-f,--socketfile filename]` `[-p,--port unsignedtype_number]` `[--snmpdf-average-len doubletype_number]` `[--snmpdf-time-between-balances unsignedtype_number]` `[--snmpdf-balance-factor doubletype_number]` `[--snmpdf-max-severity-factor doubletype_number]` `[--snmpdf-min-severity-factor doubletype_number]` `[--snmpdf-min-balances4info unsignedtype_number]` `[--snmpdf-ttl seconds]` `[--snmpdf-timeout-text-prefix text]` `[--snmpdf-sender-delay-max unsignedtype_number]` `[--snmpdf-severity-growing-factor-4min unsignedtype_number]` `[--snmpdf-severity-growing-factor-4max unsignedtype_number]` `[--snmpdf-ignored-community text]` `[--out-udp-socketfile filename]` `[--out-udp-port unsignedtype_number]` `[--out-host hostname]` `[-P,--pid-file filename]` `[-l,--log-facility log_facility]` `[-L,--log-level log_level]` `[-u,--run-as-user username]` `[--background]` `[--ignore-case-4-severity]` `[-v,--version]` `[-h,--help]`

13.3. Options

Table 13.1. damsnpdaud options

Option	Description
<code>-f,--socketfile filename</code>	Use a specified socket file for listening (default: <code>/tmp/damsnmpdaud.socket</code>).
<code>-p,--port unsignedtype_number</code>	Use a specified UDP port number for listening instead of a socket file (default socket file: <code>/tmp/damsnmpdaud.socket</code>).

SNMP Data Analyser (damsnmpdaud)

Option	Description
<i>--snmpdf-average-len doubletype_number</i>	Use a specified length of a window of average values for each data; a length of windows for minimum and maximum values is the same and equals $(1.0 - \text{len_of_average})/2$; this parameter must be inside range (0.0;1.0) (default: 0.9).
<i>--snmpdf-time-between-balances unsignedtype_number</i>	Use a specified number of seconds that must occur between balancing of each data (default: 10800).
<i>--snmpdf-balance-factor doubletype_number</i>	Use a specified balancing factor to learn about limits of windows of average values (default: 0.3).
<i>--snmpdf-max-severity-factor doubletype_number</i>	Use a specified severity factor to decide whether an incoming message should be forwarded when its value is above limit of a window of its average values about length of that window multiplied by that factor (only if autobalance is ON for that value, default: 2.0).
<i>--snmpdf-min-severity-factor doubletype_number</i>	Use the specified severity factor to decide whether an incoming message should be forwarded when its value is below limit of a window of its average values about length of that window multiplied by that factor (only if autobalance is ON for that value, default: 0.5).
<i>--snmpdf-min-balances4info unsignedtype_number</i>	Send outgoing UDP packets when number of balancing procedures for a given item is equal or greater then a specified value (default: 36).
<i>--snmpdf-ttl seconds</i>	Destroy item that its modification time is older then a specified number of seconds (default: 172800).
<i>--snmpdf-timeout-text-prefix text</i>	Specify a prefix of a text that will precede (in ID of a message) an IP address of a host which doesn't respond on SNMP requests (default: 'Timeout for: ').
<i>--snmpdf-sender-delay-max unsignedtype_number</i>	Specify a maximum number of seconds after that a new sender will be assigned as a new trustworthy sender of that data (default: 10).
<i>--snmpdf-severity-growing-factor-4min unsignedtype_number</i>	Specify a severity growing factor for values below windows of average values (default: 4).
<i>--snmpdf-severity-growing-factor-4max unsignedtype_number</i>	Specify severity growing factor for values above windows of average values (default: 4).
<i>--snmpdf-ignored-community text</i>	Specify a community to stop forwarding some type of data; data contains (it's case insensitive) that community will not be forwarded (default: ignore).
<i>--out-udp-socketfile filename</i>	Use a socket file for outgoing packets (/tmp/damadbud.socket).
<i>--out-udp-port unsignedtype_number</i>	Send outgoing UDP packets to this port (default: use socket file '/tmp/damadbud.socket').

Option	Description
<i>--out-host hostname</i>	Send outgoing UDP packets to that host (default: 127.0.0.1).
<i>-P,--pid-file filename</i>	Write PID to the specified file.
<i>-l,--log-facility log_facility</i>	Choose log facility: daemon user local0 ... local7 (default: local6).
<i>-L,--log-level log_level</i>	Choose log level (on stderr and syslog) i.e. messages of selected level and more important levels will be logged: emerg alert crit err warning notice info debug0 ... debug2 (default: warning).
<i>-u,--run-as-user username</i>	Drop root privileges and run server as the specified user.
<i>--background</i>	Go to background after startup.
<i>--ignore-case-4-severity</i>	Calculate a processed message severity ignoring case.
<i>-v,--version</i>	Display version number on stderr and exit.
<i>-h,--help</i>	Display this help and exit.

13.4. Configuration file format

A configuration file consists of objects of `group` type. A definition of object is limited by braces `{ }`. A `group` keyword that describes type of object must be a single word in a whole line. The same rule is applicable to braces that begin and end an object definition. An generic object definition is shown below:

```
group
{
...
}
```

An object is built from an unlimited number of pairs of items. Each item has two lines of a file. The first line begins from `string` keyword and specifies a string and the second one begins from `value` keyword - its severity. Following examples show how definitions of `group` object may look like:

```
string=222.222.222.222
```

```
value=0.53
```

```
string=A really bad news
```

```
value=0.99
```

Numbers describing values are treated as float numbers inside range <0.0;1.0>.

13.5. File format of minimal levels definitions

The configuration file consists of objects of `level` type. A definition of an object is limited by braces `{}`. A `level` keyword that describes type of object must be a single word in a whole line. The same rule is applicable to braces that begin and end an object definition. An generic object definition is shown below:

```
level
{
...
}
```

Fields occurred between braces are exactly specified like an order of them. Each field in an object definition takes whole line and has got below form:

```
słowo_kluczowe=
```

The first field is `minvalue4info`. It describes minimal value as a number of `double` type which must be exceeded that a message may be send. Another and the same time the last field is `name`. It may occur one or more time and it defines text which must be included in an identifier of checked data that this definition may be applied to it.

13.6. Description

When the agent, which waits for messages from network clients, receives an UDP packet then checks its IP address. In this case `.known.hosts` file is processed. If `.known.hosts` file can't be opened to read or an IP address of a client doesn't match any record in this file, the UDP packet is dropped. No authorization is made for local clients.

Processing of the next accepted item is begun with checking whether processed data exists in **SNMP Data Analyser Database**. Two any items are the same then and only then when their identifiers (i.e. [damsnmpdi](#) receives that identifier as an argument of option `-I`) are equal. If it doesn't exist, it is stored in the Database with the started values such as: creation and modification time, number of appearances etc. If the item has already occurred in the Database, its modification time and a counter of appearances is updated. If the item is `error` or `timeout` type, a counter of appearances is increased and then data is put to a procedure which sends it to the output of **SNMP Data Analyser**. When the item is `counter` type, increase of value of this item is calculated since last modification of that data. In this case that increase is tested in further procedures instead of a real value. When the item is `gauge` type, a real value

of that data is tested. Then, if specific data appear, information about the message can be sent to output of **SNMP Data Analyser**.

An item may be dropped before actualization of appropriate item in the Database when specific conditions occur. It takes place when an identifier of a sender of the item is different then an actual sender of it. After some amount of occurring of such item (an argument of [--snmpdf-sender-delay-max](#) option), while an actual sender is still silent, an identifier of a new sender will be assigned to this item.

Next step in processing of an item is comparing its value with a window of average values for this data. If its value is lower then minimum limit of that window or greater then maximum limit of it, the item is passed to the procedure which sends data to output of **SNMP Data Analyser**.

The balancing procedure is run independently of received and processed data. It checks whether balancing process should be run for a given item (fixed limits of a window of average, i.e. accepted, values may be set for any data and that limits will never be changed). The balancing process will be repeated for that item if specified time is elapsed since the last balancing procedure for that item. At the first balancing procedure, limits of a window of average values (`window_min`, `window_max`) are calculated according to the following formula:

```
delta = (max - min) * (1.0 - len) / 2.0 * factor
```

```
window_min = min + delta
```

```
window_max = max - delta
```

where `max` is a maximum value of processed item since its beginning, `min` is a minimum value of that item. Parameter [--snmpdf-average-len](#) option of **damsnmpdaud** and `factor` is an argument of [--snmpdf-balance-factor](#) option of it. During next balancing procedures, limits of a window of average values are calculated as follows:

```
delta1 = (window_max - window_min) * ((1.0 - len) / 2.0 - hits_min / hits_after_balance) * factor
```

```
delta2 = (window_max - window_min) * ((1.0 - len) / 2.0 - hits_max / hits_after_balance) * factor
```

```
window_min = window_min + delta1
```

```
window_max = window_max + delta2
```

where `hits_min` is a number of appearances of values below minimum limit of a window of average values since the last balancing procedure and `hits_max` is similarly a number of appearances of values above maximum limit of a window of average values. Parameter `hits_after_balance` is a number of appearances of this data since the last balancing procedure. After updating of limits of the window, the

hit counters of particular ranges are reset and counter of all hits since the last balancing procedure is reset too.

The procedure which sends data to the output of **damsnmpdaud**, checks (when sending data is other type than `timeout` or `error`) whether the balancing procedure has occurred a specified number of times for that item (when `autobalance` is on for that data). If that verification is succeeded and when `autobalance` is on for this data and it is not `timeout` or `error` type, then it is checked whether limits of the window of average values are exceeded significantly. Value of a given item is below a specified level for that item when its identifier matches to some defined object of `level` type (i.e. all strings of `name` fields in this objects are included in this data identifier) and arithmetical average of limits of a window of average values for that data and absolute value of its value are lower then value of that `level` object. In order to check whether the value exceeds a limit of a window of average values for that item, a following value is calculated:

```
window_min * min_severity_factor
```

when a value of data is below minimum limit of a window of average values and it's `counter` type or:

```
window_min - (window_min - min) * min_severity_factor
```

where a value of data is below minimum limit of a window of average values and it's `gauge` type, and according to a formula:

```
window_max + (window_max - window_min) * max_severity_factor
```

where a value of data is above maximum limit of a window of average values. Parameter `min_severity_factor` is an argument of [--snmpdf-min-severity-factor](#) option of **damsnmpdaud** and `max_severity_factor` is an argument of [--snmpdf-max-severity-factor](#) option. In two first cases the item won't be sent when its value is greater or equal calculated quantity. In the last case the item won't be sent, when its value is lower or equal calculated quantity.

When the item isn't dropped by the sending procedure, its severity is calculated and the item is sent to output of **damsnmpdaud**. A typical receiver of such messages is **Associations Database**.

The procedure that estimates severity of items takes only an identifier of message into consideration (i.e. [damsnmpdi](#) receives that identifier as an argument of [-I](#)). That identifier is unique in all **Operation Manager**. An initial severity of item is set to zero. Next for each item from the configuration file is checked whether string after `string=` keyword is included in an identifier of a processed item. If yes, a maximum value, assigned to `string` item, is taken into consideration. That value is added to current calculated severity of the item. At last the calculated severity of item is divided by number of objects of `group` type which are defined in the configuration file. When the item is neither `timeout` nor `error` type, it will be calculated about how many percent a limit of a window of average values will be exceeded

in the ratio to extreme values. An extreme lower value is minimum of two numbers: the smallest value that has occurred for that item or zero.

For a value below the window, the severity is calculated as follows:

```
factor * (1.0 - (x - min) / (window_min - min))
```

or

```
factor * (1.0 - (x - min) / (max - min))
```

where x is a processed value, $window_min$ is a minimum limit of the window of average values and min is a minimum while max is maximum limit of a value. The parameter `factor` is an argument of [--snmpdf-severity-growing-factor-4min](#) option. The value is counted according to a second formula when a length between $window_max$ and $window_min$ is less then 1% of a length between max and min .

For a value above the window, the severity is calculated as follows:

```
factor * (1.0 - (max - x) / (max - window_max))
```

or

```
factor * (1.0 - (x - min) / (max - min))
```

where x is a processed value, $window_max$ is a maximum limit of the window of average values and max is a maximum while min is a minimum limit of a value. Parameter `factor` is an argument of [--snmpdf-severity-growing-factor-4max](#) option. The value is counted according to a second formula similar as the value below the window. The severity is limited to one. At the end the severity gets a negative sign.

The serviced procedure of the Database is run every some period of time and it deletes data which from some period of time (an argument of [--snmpdf-ttl seconds](#) option) wasn't updated (modified). Only this data will be modified which has got an attribute allowing to this operation.

13.7. Related articles

[Access to the SNMP Data Interface \(damsnmpdi\)](#)

[Cases Database Unit \(damadbud\)](#)

Chapter 14. Cases Database Unit (damadbud)

14.1. General

damadbud is **Cases Database Unit** and it is a part of **Operation Manager**. It is a daemon process which works all the time the system is running and it expects data from [SNMP Data Analyser](#) and [SNMP Trap Analyser](#). On the basis its configuration, **damadbud** builds connections between received messages. Every received message is forwarded further by the daemon. Dedicated recipient of these messages is [Cases Service Unit](#).

damadbud reads its configuration file `.damadbudrc` during its startup. The program expects to find its configuration file in the directory `/etc/david-system`.

14.2. Synopsis

damadbud can be run with the following options: `[-f,--in-socketfile]` `[-p,--in-port unsignedtype_number]` `[-P,--pid-file filename]` `[-l,--log-facility log_facility]` `[-L,--log-level log_level]` `[-u,--run-as-user username]` `[--background]` `[-v,--version]` `[-h,--help]`

14.3. Options

Table 14.1. damadbud options

Option	Description
<code>-f,--in-socketfile</code>	Use a specified socket file for listening (default: <code>/tmp/damadbud.socket</code>).
<code>-p,--in-port unsignedtype_numbe</code>	Use a specified UDP port number for listening instead of a socket file (default socket file: <code>/tmp/damadbud.socket</code>).
<code>-P,--pid-file filename</code>	Write PID to a specified file.
<code>-l,--log-facility log_facility</code>	Choose log facility: <code>daemon</code> <code>user</code> <code>local0</code> ... <code>local7</code> (default: <code>local6</code>).
<code>-L,--log-level log_level</code>	Choose log level (on stderr and syslog) i.e. messages of selected level and more important levels will be logged: <code>emerg</code> <code>alert</code> <code>crit</code> <code>err</code> <code>warning</code> <code>notice</code> <code>info</code> <code>debug0</code> ... <code>debug2</code> (default: <code>notice</code>).
<code>-u,--run-as-user username</code>	Drop root privileges and run server as a specified user.
<code>--background</code>	Go to background after startup.
<code>-v,--version</code>	Display version number on stderr and exit.
<code>-h,--help</code>	Display this help and exit.

14.4. Configuration file format

The configuration file consists of objects that type is specified by a keyword used inside their names and their specification is limited by braces `{ }`. Currently only one type of objects is permitted and its keyword is `association`. A keyword that describes type of object must be a single word in all line. The same rule is applicable to braces that begin and end an object definition. An generic object definition is shown below:

```
association
```

```
{  
  
...  
}
```

Fields occurred between braces are exactly specified like an order of them. Each field in a object definition takes whole line and has got below form:

```
keyword=
```

A first field is `var` that must be followed by a string describing name of variable which may be used in the further part of the definition. A field `var` may occur zero or more times and in that way it may define many variables. Two last fields of the definition are: `string1` and `string2`. They describe identifiers of messages that is going to be associated each other. An example definition may have following form:

```
association
```

```
{  
  
var=$1  
  
string1=link $1 up  
  
string2=link $1 down  
}
```

and according to that following two messages:

```
information: link serial1/1 up in router  
  
information: link serial1/1 down in router
```

match each other but these ones:


```
information: link serial1/2 up in router
```

```
information: link serial1/1 down in router
```

don't match each other.

14.5. Description

When the agent, which waits for messages from network clients, receives a UDP packet then checks its IP address. In this case `.known.hosts` file is processed. If `.known.hosts` file can't be opened to read or a IP address of a client don't match any records in this file, the UDP packet is dropped. No authorization is made for local clients.

For each processed message, the list of active messages is searched to check if that message is already on the list. Two messages are treated as the same one then and only then, when they are the same type (SNMP Data or SNMP Trap) and meet specific conditions for given type. Their identifiers must have be equal for SNMP Trap messages but for SNMP Data messages they also must have the same sending reason by [SNMP Data Analyser](#) (the reason must be one of: a value below minimum limit of a window of average values, a value above maximum limit of this window, error or no responding for requests). If the message is on the list of active messages, it will be updated and next, information about it is sent on out of **damadbud**.

In case of the message isn't found on the list of active messages, all messages are searched. If the message is found on the list, it will be updated and will stay active. If the message isn't be found, it will be added to the list of messages as active. The last operation on the processed message is sent it on out of **damadbud**.

When a new item is added to the Database, it will be checked if the item should be associated with some items from the Database according to configuration from `.damadbudrc` file, which was loaded during the startup. In this case each message from the Database is processed with no matter, if the message is on the list of active message or not. Each such pair is checked, whether their identifiers match to definition of any `association{...}` object, that can be found in the configuration file. In this case each of defined objects is browsed. If `string1` occurs in the first message and `string2` in the second one when selected parts of identifiers are treated as variables according to fields `string1` and `string2` definitions, the messages will be associated with each other.

Program **damadbud** also browses active messages. These of them, that weren't updated for last 15 minutes, will be set as passive state. However, a procedure which deletes passive messages, is run every some minutes.

14.6. Related articles

[SNMP Trap Analyser \(damsnmptaud\)](#)

[SNMP Data Analyser \(damsnmpdaud\)](#)

[Cases Service Unit \(damcsud\)](#)

Chapter 15. Cases Service Unit (damcsud)

15.1. General

damcsud is **Cases Service Unit** and it is a part of **Operation Manager**. It is a daemon process which works all the time the system is running and it expects data from [damadbud](#) and services Cases. On the base of data collected in **Associations Database** it tries to consolidate received messages creating cases, that are serviced by the system. **damcsud** removes obsolete items from **Cases Database**. Its work consists in deleting obsolete data from **Cases Database** and running, if need be, specified programs for them. **damcsud** reads its configuration file `.damcsudrc` during its startup. The program expects to find the configuration file in the directory `/etc/david-system`. **damcsud** daemon won't be run if it can't read its configuration file.

15.2. Synopsis

damcsud can be run with the following options: `[-P,--pid-file filename]` `[-l,--log-facility log_facility]` `[-L,--log-level log_level]` `[--max-load-level double_number]` `[--max-period-4-load-checking seconds]` `[--case-ttl seconds]` `[-u,--run-as-user username]` `[--background]` `[-v,--version]` `[-h,--help]`

15.3. Options

Table 15.1. damcsud options

Option name	Description
<code>-P,--pid-file filename</code>	Write PID to a specified file.
<code>-l,--log-facility log_facility</code>	Choose log facility: daemon user local0 ... local7 (default: local6).
<code>-L,--log-level log_level</code>	Choose log level (on stderr and syslog) i.e. messages of selected level and more important levels will be logged: emerg alert crit err warning notice info debug0 ... debug2 (default: warning).
<code>--max-load-level double_number</code>	Suspend running actions when underlying OS average load is higher then that value (default: 20.0). After death actions and only once run actions and also that one which repeating time is greater then an argument of <code>--max-period-4-load-checking</code> option.
<code>--max-period-4-load-checking seconds</code>	Don't check underlying OS average load for actions with repeating time is greater then that value (default: 60).
<code>--case-ttl seconds</code>	Set Time to Live (in seconds) for each case from its modification time (default: 180).
<code>-u,--run-as-user username</code>	Drop root privileges and run server as a specified user.
<code>--background</code>	Go to background after startup.

Option name	Description
<code>-v,--version</code>	Display version number on stderr and exit.
<code>-h,--help</code>	Display this help and exit.

15.4. Configuration file format

The configuration file consists of objects that type is specified by a keyword used inside their names and their specification is limited by braces `{ }`. Currently only one type of objects is permitted and its keyword is `action`. A keyword that describes type of object must be a single word in a whole line. The same rule is applicable to braces that begin and end an object definition. An generic object definition is shown below:

```
action
```

```
{  
...  
}
```

Fields occurred between braces are exactly specified like an order of them. Each field from first six ones in an object definition takes a whole line and has got below form:

```
keyword=
```

The first field is `id` that must be followed by an identifier (number of unsigned long type) of that object. On the second place is `delaytime` field, that sets a number of seconds which must lapse from creation time of a case, that a given action can be run. The next fields `repeatingtime` indicate about what period of time the action can be repeated. A next field `period` helps to specify repeating and occurring of the action. That field is enumeration type. The following values are permissible:

```
after_death
```

```
after_join
```

```
after_split
```

```
once
```

Every other value of that field (also empty) is treated as `always` value.

The last field is `program_spec` object. It may occur more than one time and its definition is limited by braces `{}`. The braces and `program_spec` keyword must occur alone in separate lines. The first field of the object is `program` which shows path to a program that should be run for the action when the given conditions will be met. The last field which can occur more than one, obligatory time is `name`. It specifies a string which must occur at least in one event of a case that the given action may be run for that case. When `name` fields are more than one, they all must occur in a single event that an action can be run.

A pattern of definition of each object is shown below:

```
action
{
id=
delaytime=
repeatingtime=
period=
program_spec
{
program=
name=
...
}
...
}
```

Meaning of individual fields in an object definition is discussed in section "Description" of **damcsud**.

15.5. Description

In the moment of new item receiving the program checks if the item doesn't occur as an event in any of existed cases. If it occur, an appearances counter of a case is increased and an event state changes on active (can't be any changes because an event state could be active earlier). If no event consistent with an original is found, a new event is created. The appearances counter of a case is set as one, an event state is given as active. Next all events are checked in next cases whether any of them isn't related with a

processed message, i.e. if a relation like that was create through [damadbud](#) program. In case of the relation is found, the processed event is added to a given case. If not, a new case will be created and it will include one event for the time being.

The first step of a service procedure is checking whether a given case shouldn't be deleted, i.e. removed from the Database. Otherwise it's checked whether for that case should be run any actions (programs) and they will be run if it's true.

Individual procedures are shown below:

15.6. Running actions (programs)

Checking whether some action should be run for a case is a procedure depends on many parameters. For a given case all defined actions are browsed (they are built according to the configuration file, which is loaded during startup of **damcsud**) and the conditions for each action are checked whether that action should be run for that case. The action won't be run, when a seconds number that elapses from the creation time of a case is lower than a value of `delaytime` parameter. Next procedure, checking if a given action can be run for the case, is a procedure checking whether all values of `name` parameters, for any `program_spec` object, occur in ID of at least one active event of that case. When the action is going to be run as the last for that case, an occurrence of `name` parameter values of the action is performed for each event (also inactive) of that case. No action will be run if all `name` parameter values don't occur for any `program_spec` object. Next the program checks whether a specified period of time described by `repeatingtime` parameter has elapsed since creation of that case. If not, the action won't be run. Next compatibility of `period` parameter is checked with a current state of the case. If the action isn't going to be the last action for that case, it will checked whether the action is running right now for this case. If yes, it won't be run another time. If a `period` parameter of a given action is defined as `once` and it had been run earlier for that case, the action won't be run. If from the last running time of a given action for this case, time defined `repeatingtime` parameter elapsed, the action will be run again. A given action also will be run when it's going to be run for the first time, or it is the last action for that case.

15.7. Input parameters passed on running actions (programs)

A program, that will be run, as its input parameters gets (besides parameters specified in a `program` field of a given action) information concerning a whole case and each event which are included in it. Kinds of parameters for each event included in a case depends on its kind i.e. if its source information is a type of data: SNMP Data or SNMP Trap. Nevertheless all parameters are preceded by keywords which begin double or single dash. Following parameters are given to all programs:

Table 15.2. Input parameters passed on running actions (programs)

Cases Service Unit (damcsud)

Parameter name	Description
--flow-severity severity	Internal severity of a case.
--flow-id ID	Number of a case in a given day.
--action-id ID	Number of an action (value of id field of a given action).
--opened	Information about opening a case (running programs decides what it means; It's usually an notification of an operator). That parameter will occur if a case is marked as open.
--community community	Sum of communities included in each event of the case.
--closing-reason description	Note about closing reason of a case. It must be specified by previously occurred actions. It is always a specified parameter for an action that is run after death of a case. In that case, not specified closing reason, is described as "Not specified".
--closed-by uid	An identifier of a person closing a case. The parameter will not occur if it hasn't been specified during previously run actions. It will never be empty for actions run after death of a case. Its default value is "-1" number.
--action-done numer_akcji:kod_powrotu	Number of an action and a code of its last return (an argument may occur zero or more times).
--flow-ctime seconds	Creation time of a case (in sec. since beginning of UNIX epoch).
--flow-mtime seconds	Modification time of a case (in sec. since beginning of UNIX epoch).

All arguments below occur for each event including in a given case. The arguments can be more then one full set given below:

Table 15.3. The list of arguments

Argument name	Description
--event-hitsnumber number	The appearances number of the event during the case is working.
--event-state state	Event state, where <i>state</i> parameter must be one of: active, passive.
--event-id number	Unique number that identifies that event.
--event-ctime seconds	Creation time of an event (in sec. since beginning of UNIX epoch).
--event-mtime seconds	Modification time of an event (in sec. since beginning of UNIX epoch).
--event-type type	Type of message that an event is originated from. The value must be one of: snmptrap, snmpdata.

When type of a message is `snmptrap`, following parameters will occur:

Table 15.4. Arguments for snmptrap message

Argument name	Description
-T seconds	Receiving time of a message (in sec. since beginning of UNIX epoch).
-s IP	Source address of a message.
-a IP	IP address of a SNMP agent that sent a message.
-c community	SNMP community of a message.
-n OID	Identifier of a message in MIB.
-e OID	Identifier of enterprise in MIB.
-t timestamp	Timestamp of a message.
-g generic	Generic number of a message.
-p specific	Specific number of a message.
-d string	Identifier of a message in Operation Manager .
-m message	A human readable message that explains meaning of a message.

When type of a message is `snmpdata`, following parameters will occur:

Table 15.5. Arguments for snmpdata message

Argument name	Description
-f type	Reason of sending of a message to Associations Database must be one of: min - a value below minimum limit of a window of average values of that data, max - a value under maximum limit of a window of average values of that data, error - error for this data, timeout - No responding for request for that data.
-T seconds	Receiving time of data (in sec. since beginning of UNIX epoch).
-s IP	Address of a polling device.
-o OID	Data identifier in MIB.
-t type	Type of data that must be one of: counte, gauge, error, timeout, none. other - none of above types.
-v value	The last value of data.
-a value	Minimum limit of a window of average values of that data.
-A value	Maximum limit of a window of average values of that data.
-d string	Identifier of that data in Operation Manager .
-m message	A human readable message that explains meaning of that data.

15.8. Data return by running actions (programs)

damcsud expects that each running action (program) can print a result of its work on its stdout. If nothing is printed on stdout, the action can't affect on a case state, for which it was run.

The work result of the action is printed as text lines. Each line corresponds with the single event included in a case or if the line consists of a single word: `opened`, it means, that a case should be marked as opened (next actions, that will be run for this case will receive an [--opened](#) argument). Another two exceptions are the lines beginning from the words `closing-reason` and `closed-by`. In the first case the text following after keyword will be remembered and next, it will be passed successive actions as an argument of [--closing-reason](#) option. Similar situation is in the second case, where the remembered identifier will be an argument of [--closed-by](#) option for successive actions. Each line included in a single event consists of two items: an event identifier (an argument of each `--event-id` option) and a code, that is a value of `long` type (32-bit signed number). If the code is a positive value or zero, the event is set as the [active](#) state. Otherwise its state is set as [passive](#). From all codes for each event which an action has generated, the maximum code is selected but it isn't lower than -1 and it's treated as a return code of this action.

If the return code is a positive number or zero, a modification time of a given case is set as current time from which a number of seconds equals a return code of the action is added (i.e. a modification time of a given case is elder then current time about number of seconds equals a return code of the action).

At the end of a return code of the action is saved for a given case.

15.9. Destroying obsolete cases

The procedure checking whether a given action can be destroyed, calculates the time elapsed since the last modification of a case. The case won't be destroyed if that value is lower than an argument of [--case-ttl](#) option of **damcsud** or when any action will be run for that case.

Just before destruction of a given case, the last action, which fulfills specific conditions of this case, can be run for it. Only one action like this can be run.

15.10. Related articles











[Cases Database Unit \(damadbud\)](#)

Chapter 16. Buttons the most often used in Web applications





16.1. The buttons meaning

There are the buttons, in the chart below, that occur the most often in Web applications. Their function in particular applications is similar and even identical sometimes. Some of the buttons can have additional functions, that were described during descriptions of the particular applications.

Table 16.1. The buttons the most often used in Web applications

Button	Description
	It allows you to recover to a previous page.
	It deletes an item i.e.: it closes a case, sets an event in a passive state etc.
	It allows you to get to an edition of a given item.
	It confirms an operation and makes it (i.e.: generating of a report using selected criterions).
	It allows you to get to a detailed view.
	It allows you to get to a higher level of item hierarchy.
	It opens a new window with data which are prepared for a printout.
	It allows you to get to a presentation of the graph with data for a given item (Collection Browser).
	It reloads a page view.
	It accepts changed values as current one.

Buttons the most often used in Web applications

Button	Description
	It allows you to get to a report for a given item (Node Reporter).
	It lets you get to a Trap browser for a given item (Trap Browser).
	It lets you get to a report browser (about cases) for a given item (Recorded Operation Browser).
	It saves changes, that were done by a user.

Chapter 17. Recorded Operation Browser

17.1. General

Recorded Operation Browser is a Web application and it is a part of **Operation Manager**. It allows you to browse registered information about cases, that are run through **Operation Manager**.

17.2. Description


17.2.1. Specification of searching criterions

17.2.1.1. Default view of the application



Recorded Operation Browser is accessible through Reports tab. It is a Cases group of the tab. If you give searching criterions in Cases group, you'll get a list of registered information about processed cases.

17.2.1.2. Cases group

Cases 

Text

Community

Source device

Show/hide matched

View type

Time range

Date and period

Results per page

Each bar occupies

Graph's width

Graph's height

Limit graph results (case view only)

The fields included in the group and their meaning are presented in the chart below.

Table 17.1. Recorded Operation Browser - fields description of Cases group

Field	Description
Text	A human readable meaning of a case.
Community	A label of the case consisted of labels of events, that included in the case (individual labels are separated from themselves ':' mark).
Source device	A device which is a source of the case.
Show/hide matched	The field means, if the cases fulfilling searching criterions are shown or hidden.
View type	Two modes of presenting of a registered cases list are possible: choosing <code>As cases</code> mode causes grouping of entries on the subject on the same case in one string of information; choosing <code>As log entries</code> mode causes treating of each entry independently and it is rather less, natural mode of information browsing about existed cases.
Time range	Time range specification through choosing of a period of time which lasts till now.
Date and period	Time range specification thorough selecting of specific date and time in combination with a specific period of time (i.e. 1 day, 2 days, 1 week etc.).

Recorded Operation Browser

Field	Description
Results per page	It describes a maximum number of entries, that are shown at once.
Each bar occupies	You can set here, how much time a single period of time will included on a graph schedule of case occurrence in time.
Graph's width	Width of a graph
Graph's height	Height of a graph.
Limit graph results	For As cases view, it sets if generated graph of particular case duration is including all selected cases or current presented one (limited by Results per page parameter).

17.2.2. Generated report

17.2.2.1. As cases view



The view of As cases type is a natural way for browsing information about occurrence cases. Main rows including serial numbers give you information about the first entry of a given case. If entries are more then one, additional row appears and it will includes information about the last entry of the case. ID column includes a case identifier. Links of Title column allow you to get to more detailed information about a given case beginning from a list of all entries concerned this case. Duration column gives time beginning from the first, visible entry of a given case to the last (often it is a duration of the case). Percent of period column shows a percentage participation of Duration field quantity of a

[Associations Database Unit \(damadbud\)](#)

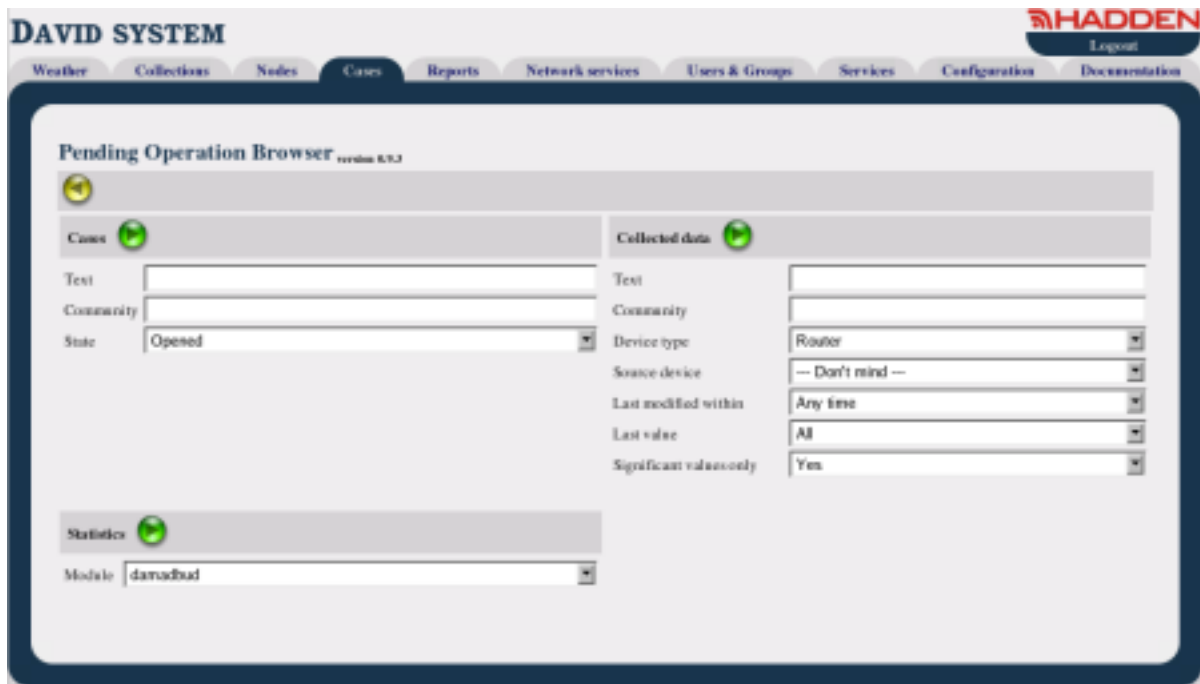
[Cases Service Unit \(damcsud\)](#)

Chapter 18. Pending Operation Browser

18.1. General

Pending Operation Browser is a Web application and it is a part of **Operation Manager**. It provides an access to current proceeding cases and to other data of the running subsystem.


18.2. Description



Pending Operation Browser is accessible through **Cases** tab. The tab is divided into groups, that make possible for access to particular kinds of data. **Statistics** group allows you to see statistic data of particular modules, that are a part of the product, and it has a diagnostic sense. **Cases** group gives access to current proceeded cases, whereas **Collected data** group makes access to data collected through [SNMP Data Analyser](#).

18.2.1. The view of active cases

18.2.1.1. Specyfication of searching criterions

The group includes `Text` field allowing to give a searching text, that shows in a description of pending cases. `Community` field allows to browse cases according to a text content in their `Community` fields. `State` field allows to choose these cases, that are currently in a chosen state. Clicking the button  runs a searching process of the cases.

18.2.1.2. The list of active cases

No.	State	ID	Updates	Description
1	🟡	20070717-59	🟢	Filter violation on interface...
2	🟡	20070717-337	🟢	Filter violation on interface...
3	🟡	20070717-338	🟢	Filter violation on interface...
4	🟡	20070717-340	🟢	Connection (eth0)...
5	🟡	20070717-341	🟢	Link down...


The list is divided into columns, that characterize the particular cases. `State` column shows a current state of a each case, while `ID` gives its identifier. The column `Updates` shows if sending updates is blocked or not. `Description` column shows a description of each case. Clicking on `ID` or `Description` field casuses displaying details about the selected case.



There is also a list of events included in the case and a list of actions, that were or are run for this case.

The list of events is divided into columns describing an event state (State column), a number of occurrence of a given event since the case was created (Hits column) and its meaning (Description column).


The list of actions is divided into columns: ID - an action identifier which comes from a configuration file of [damcsud](#) module, State - a current action state (DONE or RUNNING) and Calls - a number of calls of a given action.

The button  which is placed on the toolbar, above description of a given case, lets you close the case.

Clicking on Description field shows a detailed view of selected event included in a given case.




The `Case` field describes a case in brief to which a given event belongs. Next fields describes properties of particular event parts, that were created in a process of event creation starting off a message SNMP Trap or SNMP Data.

The button  which is placed on the toolbar, above a given event description, lets you to set an event state as passive.

18.2.2. The view of collected SNMP Data

18.2.2.1. Specification of searching criterions

Collected data 

Text

Community

Device type

Source device

Last modified within


Last value

Significant values only

The fields of Collected data group and their meaning are shown in the chart below.

Table 18.1. Pending Operation Browser - meaning fields of Collected data group

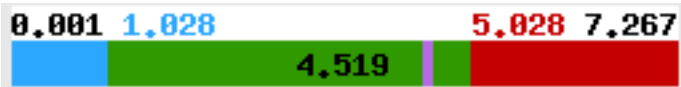
Field	Description
Text	A human readable meaning of data.
Community	A label assigned to a given data
Device type	A device type which data comes from.
Source device	A device which is a sender's data.
Last modified within	A period of time in which data has been modified lately.
Last value	A range of values to which data can be included (in the range of average values, besides this range, below the range, above the range).
Significant values only	It tells whether only significant values should be shown (data with values below the predefined threshold are ignored).

Clicking on the button  runs searching process of collected data, that correct values [SNMP Data Analyser](#) learns.

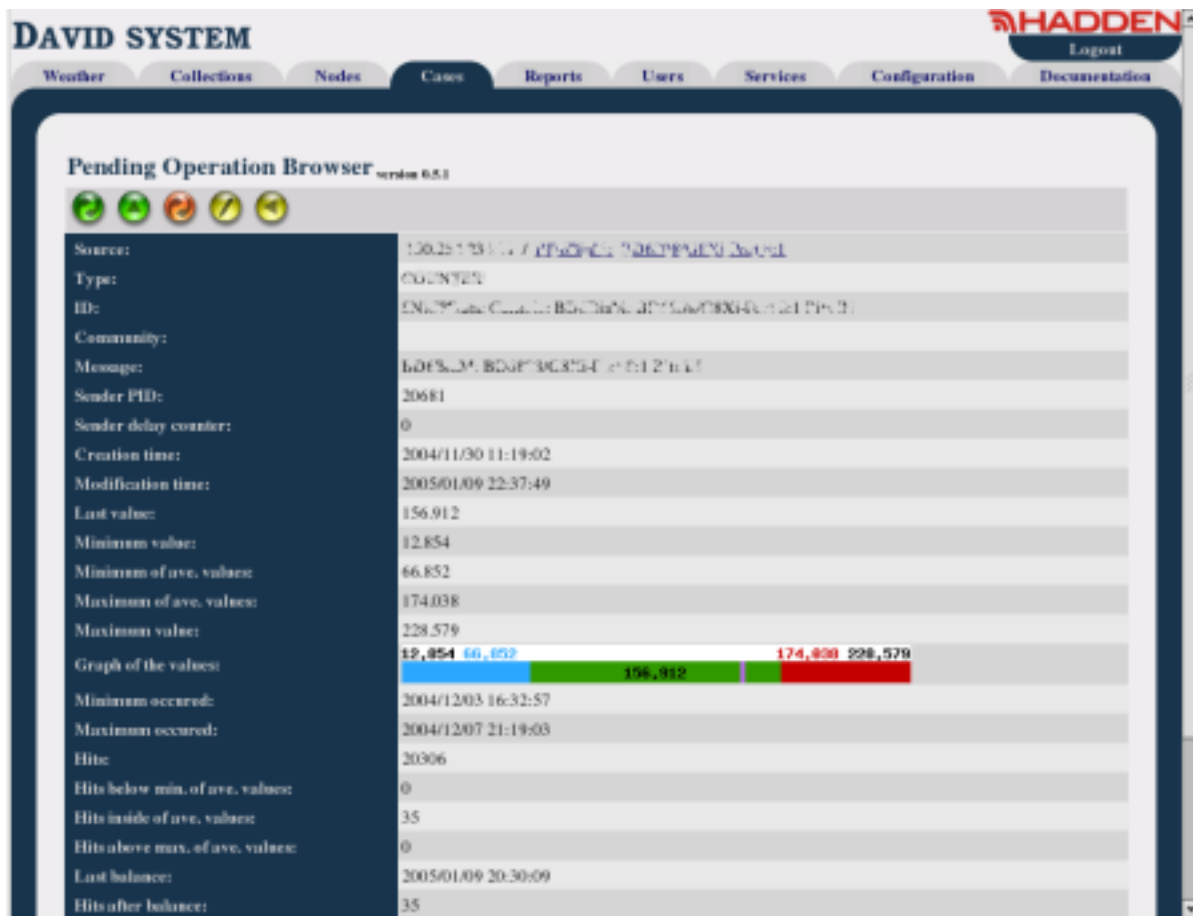
18.2.2.2. The list of SNMP Data, that corresponds with searching criterions

No	State	Source	Description	Modification time	Last value
1	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:IN	3 min. 14 sec. ago	37,854 89,972 174,888 228,574
2	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:OUT	3 min. 14 sec. ago	0,000 13,218 81,774 27,556 329,481
3	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:IN	3 min. 14 sec. ago	0,000 11,710 38,888 58,288
4	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:OUT	3 min. 14 sec. ago	27,549 49,972 142,868 284,720
5	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:OUT	14 sec. ago	0,000 18,418 64,118 48,888 48,888
6	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:IN	14 sec. ago	0,000 18,418 64,118 48,888 48,888
7	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:OUT	14 sec. ago	0,000 29,702 182,888 284,872
8	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:IN	14 sec. ago	0,000 22,867 182,888 218,374
9	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:IN	1 min. 9 sec. ago	0,999 28,888 57,776 88,774
10	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:OUT	1 min. 9 sec. ago	0,000 2,517 18,418 25,274
11	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:IN	1 min. 9 sec. ago	0,000 2,517 18,418 25,274
12	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:OUT	1 min. 9 sec. ago	4,812 5,517 88,888 185,558
13	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:OUT	25 sec. ago	0,000 27,833 84,888 145,812
14	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:IN	25 sec. ago	0,000 29,474 84,888 118,888 148,888
15	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:IN	25 sec. ago	0,448 24,933 182,888 284,812
16	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:IN	25 sec. ago	0,000 35,578 88,888 285,288
17	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:OUT	25 sec. ago	37,885 37,231 282,888 282,888
18	🟢	BDsBaM: BDsBaM:BDsBaM:Port 8.1	BDsBaM: BDsBaM:BDsBaM:Port 8.1:Port 8.1:OUT	25 sec. ago	0,000 4,137 22,813 48,123


The list is divided into columns. State column shows if the last data value is below the range of average values (violet), if it includes in the range (green), or if it is above the range (red). Source column shows a source data item (i.e.: a device and network interface which data belongs to). Modification time column presents last modification time of data. Last value column shows a location of last data value in the range of its average values. The graphs presents values: minimum and maximum in black color, minimum out of average values in blue color, and maximum in red color and additionally drawn as violet line.

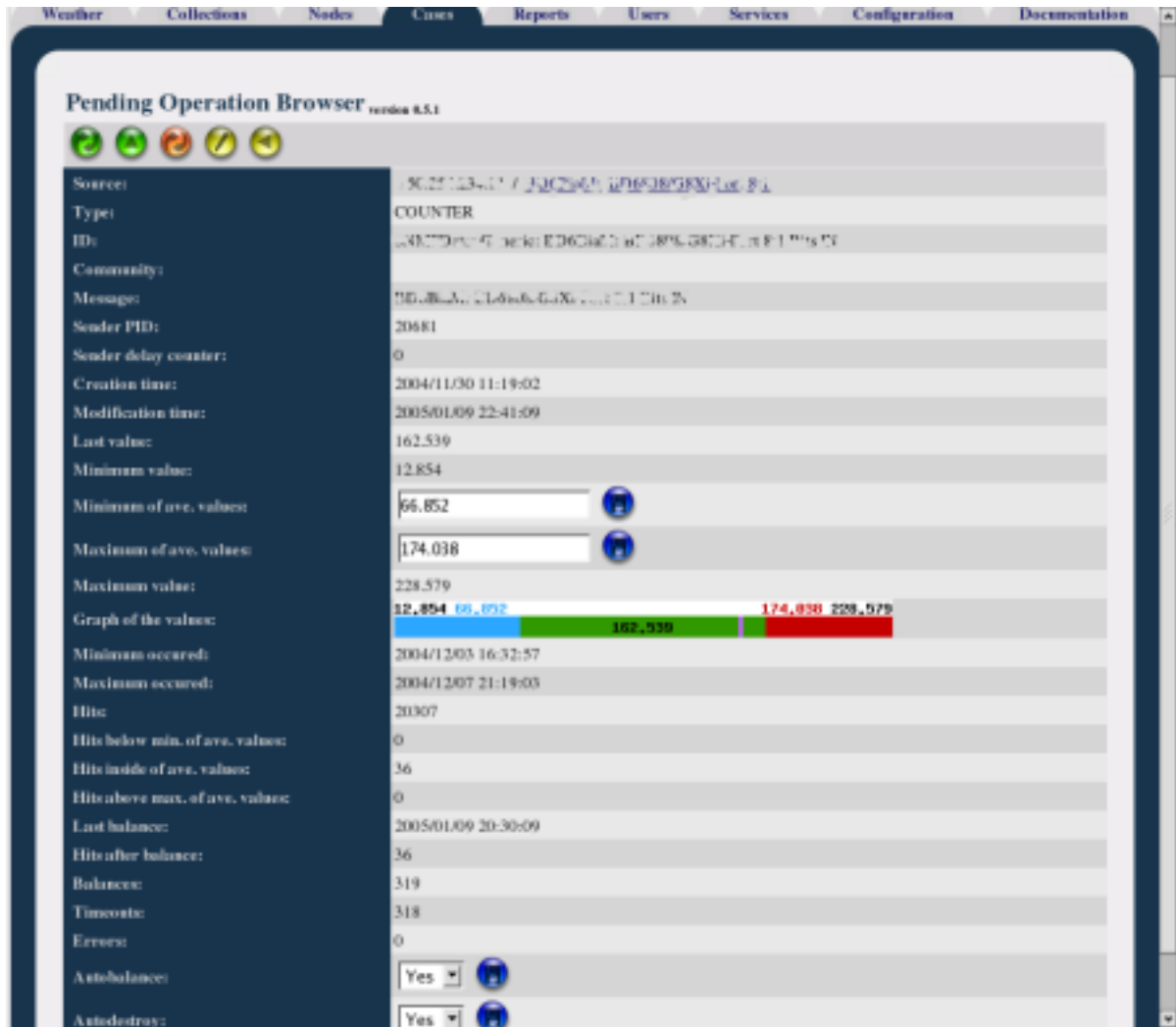


Description column describes a human readable meaning of data and it includes a link. When you click it, you'll see detailed information about selected item.



There are detailed values of particular fields of SNMP Data and a visualisation of the average values window, that ranges the system learns during its work. The system configuration allows to set not to learn average values for selected data by the system, but the values can be set by a user.

Clicking the button  allows to go to edition mode of selected SNMP Data. A user can define limits of the average values window.



Clicking on either of these buttons  allows to save any changes.

18.3. Related articles

[Access to the SNMP Trap Interface \(damsnmpti\)](#)

[SNMP Trap Analyser \(damsnmptaud\)](#)

[Access to the SNMP Data Interface \(damsnmpdi\)](#)

[SNMP Data Analyser \(damsnmpdaud\)](#)

[Cases Database Unit \(damadbud\)](#)

[Cases Service Unit \(damcsud\)](#)

Chapter 19. Graphic Notifications Presenter (xdgnp)

19.1. General

xdgnp application is **Graphic Notifications Presenter** and it is a part of **Operation Manager**. The application allows to display graphic notifications concerning cases about which full information is received from [dgnsd](#) server and react an operator to these notifications. It can play also sound files and read text.

19.2. Synopsis

xdgnp can be run with the following options: [[-l,--log-facility log_facility](#)] [[-L,--log-level log_level](#)] [[-v,--version](#)] [[-h,--help](#)]

19.3. Options

Table 19.1. xdgnp options

Option name	Description
-l,--log-facility log_facility	Choose log facility: daemon user local0 ... local7 (default: local6).
-L,--log-level log_level	Choose log level (on stderr and syslog) i.e. messages of selected level and more important levels will be logged: emerg alert crit err warning notice info debug0 ... debug2 (default: warning).
-v,--version	Display version number on stderr and exit.
-h,--help	Display this help and exit.

19.4. Description

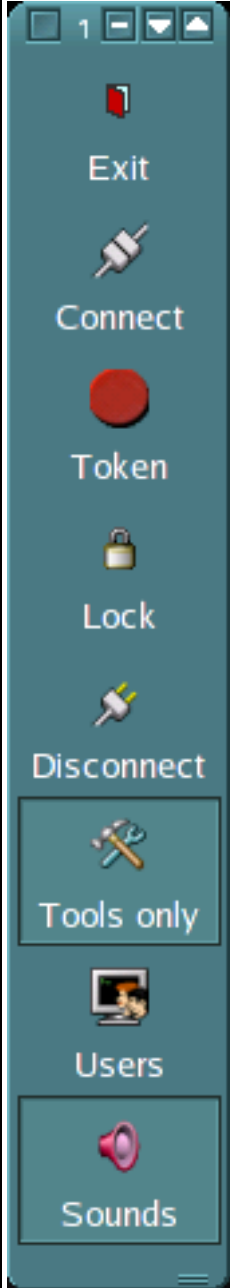
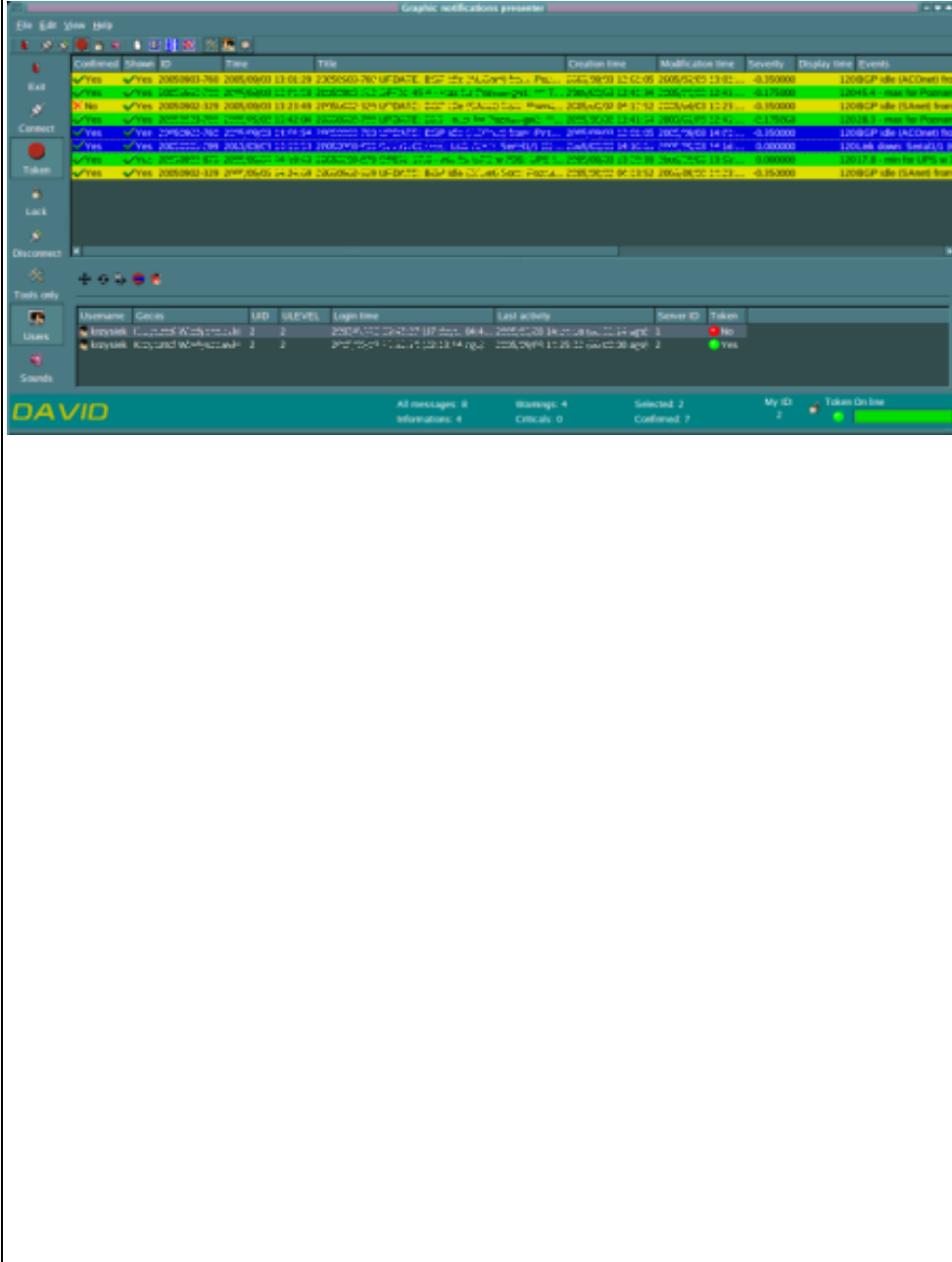
19.4.1. Starting up and terminating the application

xdgnp reads its configuration parameters from `.xdgnprc` file during its startup. These parameters concern appearance of the program and other working parameters. The application expects to find its configuration file in a directory which name is kept in environmental variable `$DAVIDPRIVDIR`. When such file doesn't exist, the application begins its work with its default settings.

19.4.2. Main view work

The application may work in two view modes. Default mode of view is a full view of the application with the toolbar, the status bar and the list of received notifications. The second mode is only view of the tool panel and in this way the application doesn't occupy so much place on the desktop. It's very easy to switch between these two modes of view during the application work.



Table 19.2. xdgnp - modes of work

A toolbar mode	A full view mode
 <p>The toolbar contains the following elements from top to bottom:</p> <ul style="list-style-type: none"> Window control icons (minimize, maximize, close) and a '1' icon. Exit button with a red power icon. Connect button with a plug icon. Token button with a red circle icon. Lock button with a padlock icon. Disconnect button with a plug icon. Tools only button with a wrench and screwdriver icon. Users button with a computer monitor icon. Sounds button with a speaker icon. 	 <p>The main window displays a list of notifications with the following columns: Confirmed, Show, ID, Time, Title, Creation time, Modification time, Severity, Display time, and Events. Below the list is a 'Tools only' section with columns for Username, Genus, UID, SUSEVEL, Login time, Last activity, Server ID, and Token. The status bar at the bottom shows 'DAVID' and summary statistics: All messages: 8, Informations: 4, Warnings: 4, Critical: 0, Selected: 2, Confirmed: 7, My ID: 7, and Token On line.</p>

The full view in its central part shows a list of received notifications. Meaning of particular columns is explained below:

Table 19.3. xdgnp - columns description

Column name	Description
Confirmed	Whether a notification window has been closed by an operator and thereby receiving of the message has been confirmed.
Shown	Whether a notification window has been shown or reason of failure otherwise.
ID	An unique identifier of a case.
Time	Notification receiving time.
Creation time	Case creation time.
Modification time	Case modification time.
Severity	Severity of a case.
Display time	Maximum time the notification window could be displayed.
Title	A title of a case.
Events	Events included in a case.

The notification list counters are displayed on the status bar. These counters show: a number of all messages, a number of information messages, a number of warnings, a number of critical notifications, a number of selected items and a number of confirmed notifications. Next, an identifier of this application assigned by [dgnsd](#) server is displayed. Next there are the buttons  and .

Next, there is a control light which signals the application owns the token on [dgnsd](#) server. The last item on the status bar is a connection state indicator.







The tool panel which buttons mostly correspond to buttons on the toolbar is displayed on the most left side of the application. `Tools only` button is the only one button located only on the tool panel and not placed on the toolbar. It switches the application between both kinds of its view.

Pressing the right mouse button above the list of received notifications shows the all menus accessible also on the top of the application. Pressing the left mouse button over an item of the list selects or unselects a given item. The list allows you to select many items at the same time.

19.4.2.1. Main view buttons









Buttons on the toolbar allow you to control the application work. The first five buttons from the left side correspond with the options of `File` menu.

Table 19.4. xdgnp - description of the buttons

Button	Description
	It lets you exit the application.
	It lets you connect to dgnsd server.
	It lets you close the connection (logout).
	It allows you to take the token.
	It allows you to lock access to the application. Locking the application causes appearing of <code>Authorization</code> , dialog in case of any mouse move or any mouse button or any keyboard key pressing. In this dialog you should enter a user name and his/her password to unlock access to the application and login to the server again (you needn't login as the same user).
	Turn on/off a sound service.

The next buttons correspond to `Edit` and `View` menu.

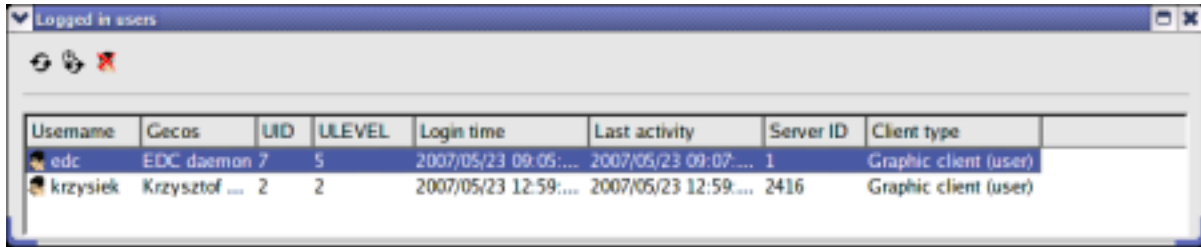
Table 19.5. The buttons correspond to Edit and View menu

Button	Description
	It lets you delete all items of the notification list.
	It lets you delete selected items.
	It lets you reverse selection of the notification list.
	It lets you unselect all previously selected items.
	It allows you to show or hide the tool panel.
	It lets you open the edition panel of comments to received notifications.
	It lets you open the list of logged in users on dgnsd server.
	It allows you to configure parameters of the application.

Additionally, in `View` menu two options are placed - `Show tool bar` and `Show status bar`. They allow you to show or hide the toolbar and the status bar.






Help menu may inquire about a version and a creation time of the application.

19.4.3. List of logged in users on dgnsd server



The dialog with a list of logged in users on **dgnsd** server allows to make some actions on these users. A set of buttons available on this dialog is assigned to that. The buttons are described in the chart below:

Table 19.6. dgnsd server - description of the dialog buttons

Button	Description
	It allows you to change the dialog into the window docked at the main view of the application and vice versa.
	It refreshes the list of logged in users.
	It turns on/off auto-refreshing of the list.
	It lets you grant the token a selected user of the list.
	It lets you logout the selected user.

The rules of granting the token and logging out the users are described in the chapter [Description of dgnsd server](#).


Meaning of particular columns of the list is explained below:

Table 19.7. Columns description

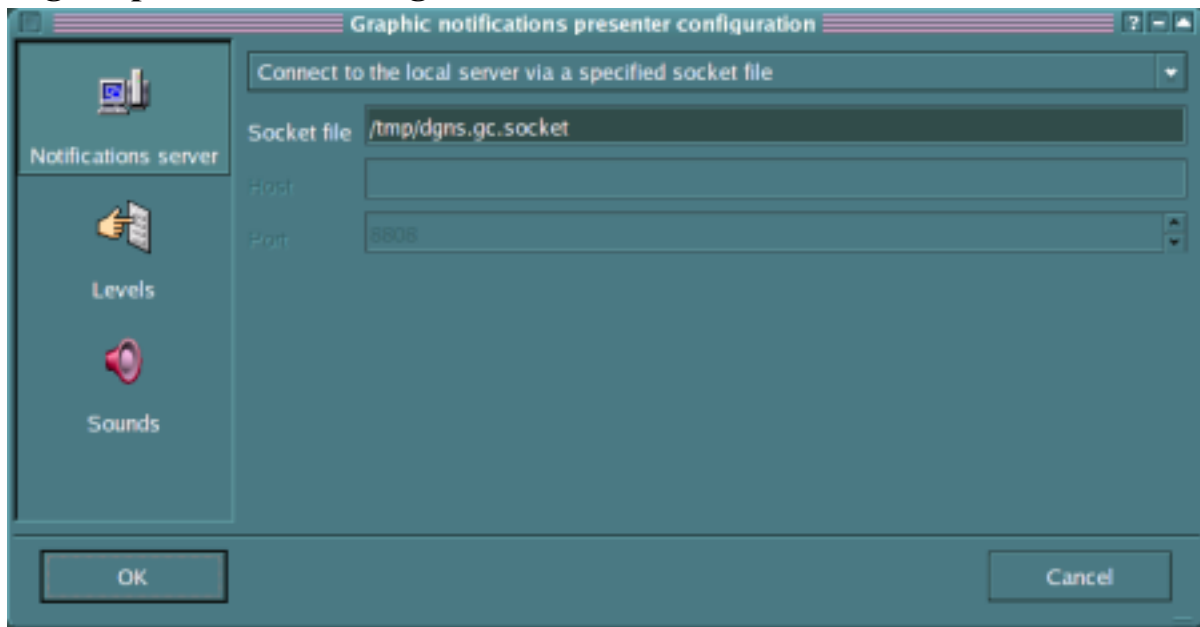
Column	Description
Username	User's name.
Gecos	Account description.
UID	User's identifier (number).
ULEVEL	Level of user's rights.
Login time	Login time as a specified user (it needn't be equal the connection time in case the authentication has been done more then once).
Last activity	Time of last activity of a user (this instance of the application) on dgnsd server understood as sending any message to the server.
Server ID	An identifier of a given instance of the application on dgnsd server (the same user may be logged in running more then one instance of xdgntp).

Column	Description
Token	Indicates whether a given user (exactly: that instance of the application) has the token.

19.4.4. xdgnp configuration

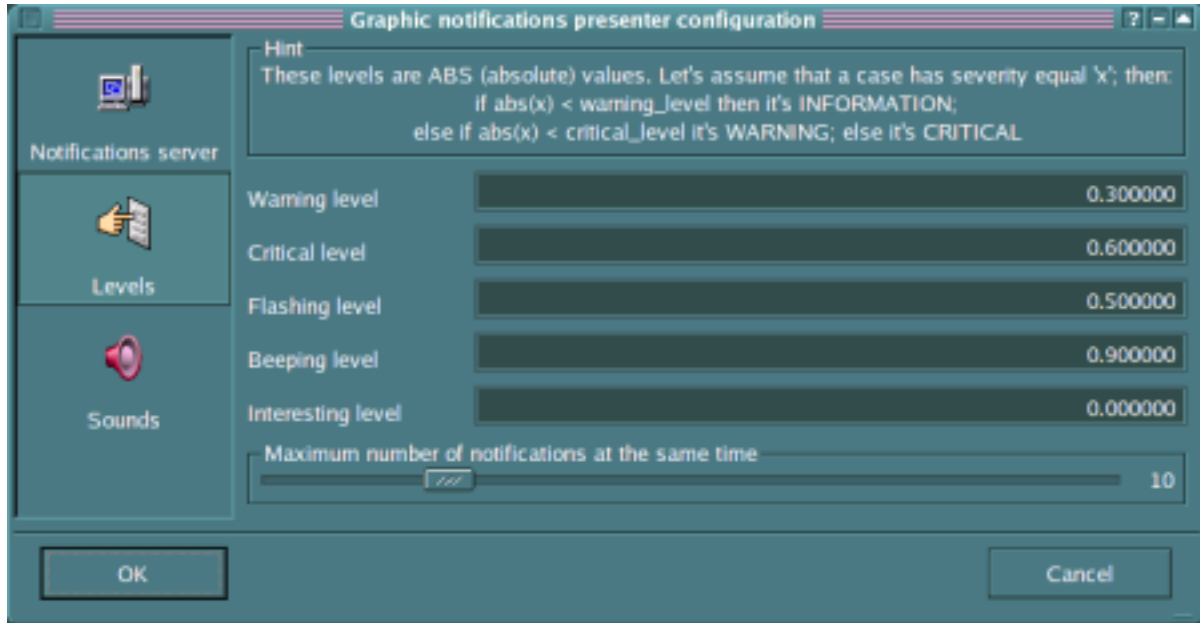
Pressing the button  opens the configuration dialog. The window is divided into three configuration areas.

19.4.4.1. dgnsd parameters configuration



There's a list of two items at the top of the window which lets you choose a type of connection with [dgnsd](#) server. The first option lets you choose the local connection via a socket file that name you may enter below. The second one indicates that you want to make the connection via the network. In this case you should enter a host name or its IP address and a number of TCP port, in which the server listens for.

19.4.4.2. Work levels configuration

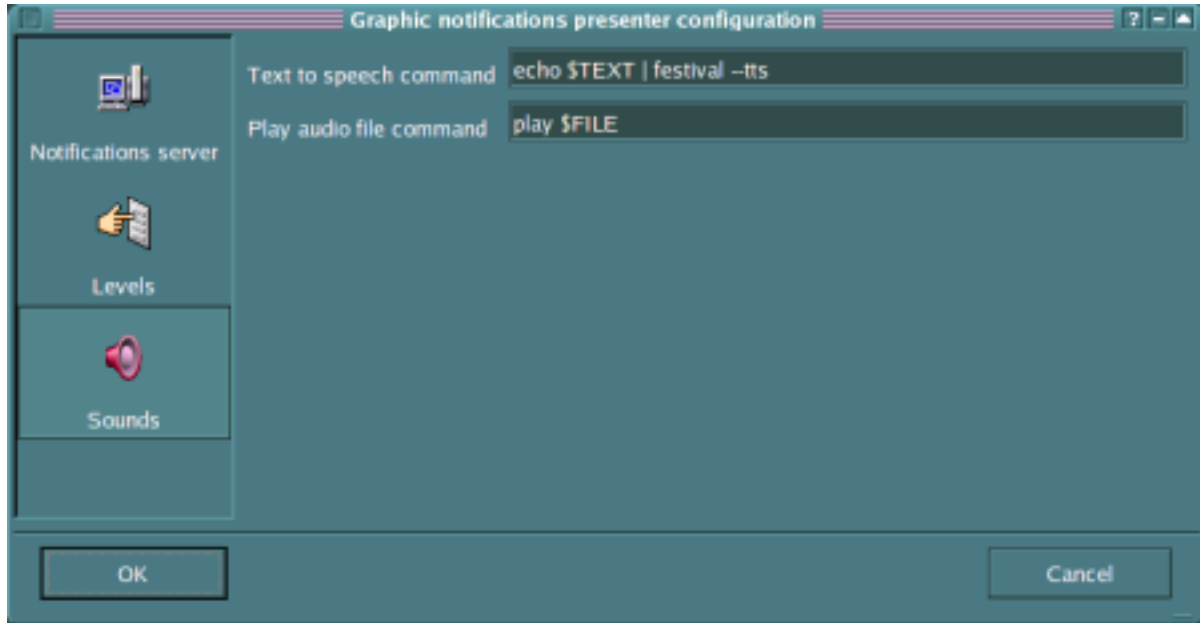


This tab lets you set levels of severities for incoming notifications. All levels are treated as absolute values i.e.: if severity of a given notification equals x , and if

$\text{abs}(x) < \text{warning_level}$

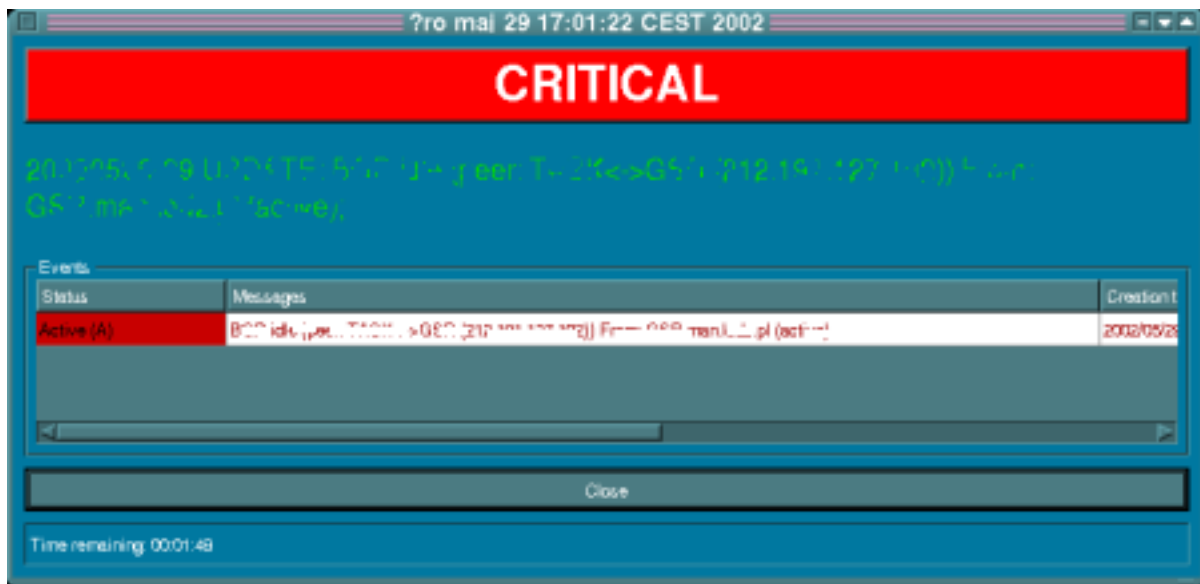
then the notification is recognized as information. Parameters `Warning level` and `Critical level` determine severity of notifications. Parameter `Flashing level` indicates level from which the notification title flashes during displaying a notification window while `Beeping level` value tells you from which severity value the window beeps. Parameter `Interesting level` shows a minimum level of severity below that notifications are not displayed as a notification window. Even then the notification is added as a new item to the received notification list in the main view of application. Maximum number of notifications at the same time group lets you limit number of notification dialogs shown at the same time.

19.4.4.3. Sound configuration in xdgnp application



The tab allows you to specify adequate commands, that are run to play sound files and text messages. In Text to speech command field a command specification is taken place, that runs the speech synthesizer. In Play audio file command a command is specified, that helps to play sound files.

19.4.4.4. A notification window



The graphic window displaying a notification shows on its title bar number of a case that generated the notification. At the top of the window there is a text describing severity of the message (INFORMATION, WARNING, CRITICAL). The severity depends on two parameters: severity of the message set by the client and levels of severities set by the user of **xdgntp** application. Below the text there is a title of the message and next there is a table that displays all events included in the case. Below the table there is Close button that lets you close the window which is equivalent to a confirmation of

receiving of a given notification. At the bottom of the window the time remaining to closing up the window is displayed.

Each row of the table presents a particular event included in the case. Particular columns describe:

- `Status` - state of an event (three possibilities: `Active (A)`, `Passive (H)`, `Not managed here (NM)`). You may change the state from active to passive and vice versa;
- `Message` - human readable description of an event;
- `Creation time` - event creation time (in this case);
- `Modification time` - last appearing time of an event;
- `Hits numbers` - number of hits of an event since its creation time;
- `Successors` - a list of successors (another events) of that event placed one by one.

19.5. Related articles

Notification Processor: Information Recorder (dsi)

[Graphic Notifications Server \(dgnsd\)](#)