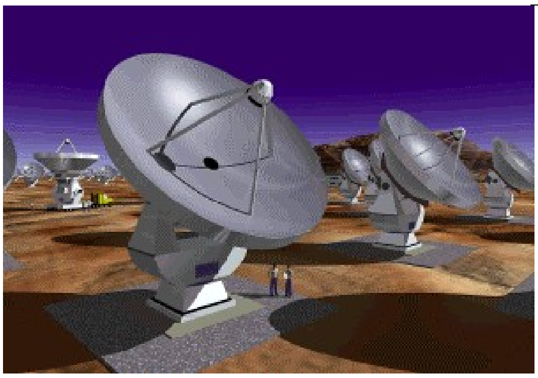


Atacama Large Millimeter	KGB-DOC-02/02 Revision: 1.13 30/04/2009 Software Manual Ales Pucelj

Logging Client User's Manual

Software Manual

Ales Pucelj (ales.pucelj@kgb.ijs.si)

KGB Team, Jozef Stefan Institute

Igor Verstovsek (igor.verstovsek@ijs.si)

KGB Team, Jozef Stefan Institute

Keywords:

Author Signature:	Date:
Approved by: Institute:	Signature: Date:
Released by: Institute:	Signature: Date:

Change Record

REVISION	DATE	AUTHOR	SECTIONS/PAGES AFFECTED
	REMARKS		
1.0	2002-02-15	Ales Pucelj	All
	Created.		
1.1	2002-11-25	R. Georgieva	
	Updated for ACS 2.0. Updated the menu items available in 3.2.		
1.2	2003-11-24	A. Caproni	
	Updated for ACS 3.0		
1.3	2004-08-11	A. Caproni	
	Updated filters and menus		
1.4	2005-05-04	A. Caproni	
	Updated for ACS 4.1		
1.5	2005-07-18	A. Caproni	
	Added the description of the search dialog		
1.6	2005-12-02	A. Caproni	
	Updated for ACS 5.0. Added a better description of Additional Info		
1.7	2006-01-10	A. Caproni	
	Revised the Additional Info section; added a paragraph for the command line parameters		
1.8	2006-07-26	A. Caproni	All
	Revised for ACS 5.0.4		
1.9	2006-30-10	A. Caproni	
	Added the description of the error dialog and the database connection		
1.10	2006-12-05	A. Caproni	
	Added acsLogAssistant		
1.11	2007-27-03	A. Caproni	
	Improved the documentation of filters		
1.12	2007-20-04	A. Caproni	
	Added missing part as requested by SPR		
1.13	2007-07-05	A. Caproni	
	Filters revisited		
1.14	2008-22-07	A. Caproni	All
	Updated fro ACS 7.0.2		

Table Of Contents

1 Introduction

Logging Client (LC) is the ACS specific tool for displaying messages read from the database, a file or received from the CORBA notification channel. Its primary intention is to offer an easy to use interface that simplifies reviewing and monitoring of the logs.

A command line tool, *acsLogAssistant* helps manipulating files of logs.

2 Requirements

Logging Client is written entirely in Java and therefore requires Java Runtime Engine (JRE). Since ACS 7.0.2 the client needs jdk 1.6 and is not anymore compatible with previous java versions.

3 Basics

3.1 Functional overview

Once Logging Client starts it tries to connect to the control system. If the connection failed, the LC works offline: the user can load and browse an xml file previously saved or retrieve logs from the database. If the connection succeeded or the user selects “Connect” from the File menu, LC connects to CORBA Notify Service "LoggingChannel" and creates an event consumer on it. This consumer then sends all the received logs to the GUI part of LC, where the logs are displayed in a table.

While working online, the client monitors the status of the connection every 15 secs. If the connection is lost then a message dialog is shown and the world “Offline” is added to the title of the main window. An icon is also present in the main window showing the status of the connection. The user can connect and disconnect for the logging service at any time by selecting the apposite item of the File menu.

The logs are stored internally in the same order they are received; however they can be displayed using many other criteria. Should any errors or problems occur during use, they will be displayed in the status bar at the bottom. All the features, such as sorting and filtering by any of the fields are described in detail in the following sections.

LC stores the logs using a multi level cache. The first level stores the logs in a file, keeping an index in memory to access the file randomly. In this way the logging client is able to manage long file of logs: it has been successfully tested with more to 1 million logs. To avoid having a file cache too long, such a file is splitted in several files each of which is removed when the application terminates.

The second level is a memory cache to reduce the number of disk operations. A subset of the logs is stored in memory using a policy to enhance the probability to have always in memory the most referenced logs.

The logging client allows to define filters when the logs are received (i.e. in the engine) and when the logs are displayed (i.e. in the table). The logs that do not match with the engine filters are discarded when they are received before being ingested by the application.

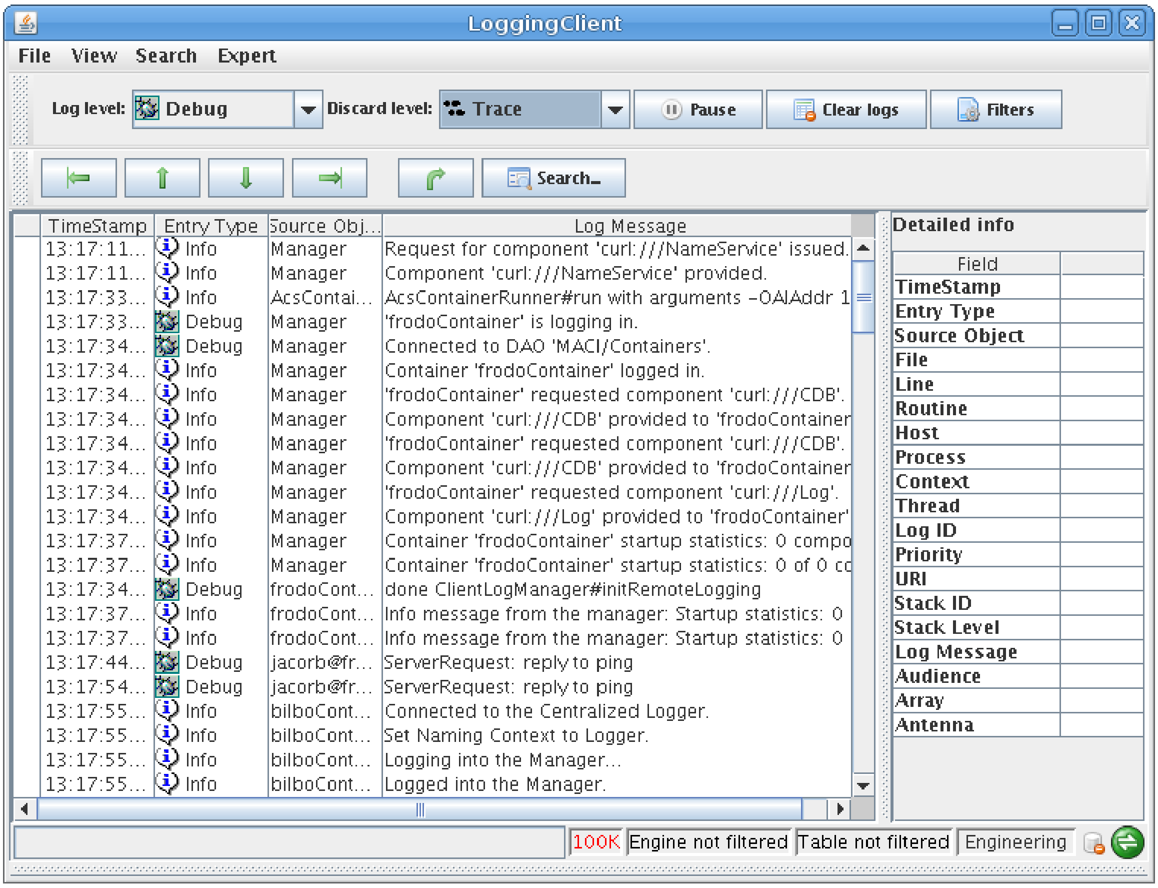
Table filters instead allow to create a view of the logs in the table. With the table filters the logs that do not match filters are removed from the table but are still available and can be displayed by removing or disabling the filters.

The Main Window

The main window of the logging client is divided into three areas:

- the main panel i.e. the table containing the logs
- detailed log information, in the right side
- the status panel at the bottom; this panel is not visible at startup but can be displayed by means of a menu item.

At the top of the application there is a double line toolbar offering the widgets to select the most used functions. At the bottom a status bar shows if there are filters applied together with other useful information that we'll see later in a greater detail.



TimeStamp	Entry Type	Source Obj...	Log Message
13:17:11...	Info	Manager	Request for component 'curl:///NameService' issued.
13:17:11...	Info	Manager	Component 'curl:///NameService' provided.
13:17:33...	Info	AcsContai...	AcsContainerRunner#run with arguments -OAIAddr 1
13:17:33...	Debug	Manager	'frodoContainer' is logging in.
13:17:34...	Debug	Manager	Connected to DAO 'MACI/Containers'.
13:17:34...	Info	Manager	Container 'frodoContainer' logged in.
13:17:34...	Info	Manager	'frodoContainer' requested component 'curl:///CDB'.
13:17:34...	Info	Manager	Component 'curl:///CDB' provided to 'frodoContainer
13:17:34...	Info	Manager	'frodoContainer' requested component 'curl:///CDB'.
13:17:34...	Info	Manager	Component 'curl:///CDB' provided to 'frodoContainer
13:17:34...	Info	Manager	'frodoContainer' requested component 'curl:///Log'.
13:17:34...	Info	Manager	Component 'curl:///Log' provided to 'frodoContainer'
13:17:37...	Info	Manager	Container 'frodoContainer' startup statistics: 0 compo
13:17:37...	Info	Manager	Container 'frodoContainer' startup statistics: 0 of 0 co
13:17:34...	Debug	frodoCont...	done ClientLogManager#initRemoteLogging
13:17:37...	Info	frodoCont...	Info message from the manager: Startup statistics: 0
13:17:37...	Info	frodoCont...	Info message from the manager: Startup statistics: 0
13:17:44...	Debug	jacorb@fr...	ServerRequest: reply to ping
13:17:54...	Debug	jacorb@fr...	ServerRequest: reply to ping
13:17:55...	Info	bilboCont...	Connected to the Centralized Logger.
13:17:55...	Info	bilboCont...	Set Naming Context to Logger.
13:17:55...	Info	bilboCont...	Logging into the Manager...
13:17:55...	Info	bilboCont...	Logged into the Manager.

Field	
TimeStamp	
Entry Type	
Source Object	
File	
Line	
Routine	
Host	
Process	
Context	
Thread	
Log ID	
Priority	
URI	
Stack ID	
Stack Level	
Log Message	
Audience	
Array	
Antenna	

The size of the three areas can be adjusted, by dragging the separators. The two lines of the toolbar can be detached from the main window by pressing the mouse button over the grayed area at their left side.

3.2 Menu

The menu bar is composed of four items: `File`, `View Search` and `Expert`.

- `File/Connect and File/Diconnect`: this menu item allows the user to connect/disconnect for the server. If the client is not connected then Connect is shown otherwise Disconnect is shown. The progress of the process will be displayed in the status panel. If LC is not connected to the server the client works offline; in this case the title of the main window has the Offline world appended at the end (see the picture above). The icon in the bottom right side of the main window is green when the application is connected to the logging channel and becomes red in case of disconnection.

- `File/Auto reconnect`: if the check box is selected LC tries to reconnect to the logging channel if the connection has been lost (defaults unchecked).

- `File/Load from File`: loads an xml file containing logs into the log table.
- `File/Load from URL`: loads the logs from a given URL
- `File/Load from database`: load the logs querying the database
- `File/Save File As`: saves the log entries that are currently displayed in an xml file.
- `File/Clear logs`: clears the log table from the logs.
- `File/Exit`: disconnects from the server and closes the application.
- `View/Toolbar`: shows or hides the toolbar
- `View/Detailed log info`: shows or hides the detailed log info panel at the right side of the main window
- `View/Status ares`: shows or hides the status area panel at the bottom of the main window
- `View/Fields`: opens the window to choose the visible columns
- `View/Short date format`: the date in the time stamp column of the table of logs are displayed as hh:mm:ss.mmm; the long format comprises the date and the time.
- `View/Show log type description`: if selected the icon together with the full description of the type of each log is displayed in the entry type column, otherwise the icon is only displayed
- `View/Filters`: opens the filters configuration window
- `View/Error browser`: opens the dialog with the logs of an error stack trace
- `View/Statistics`: opens the statistic dialog
- `View/Error log`: show the error dialog reporting the errors detected by jlog during its functioning
- `Expert/Mode`: allow to select the operating mode between Engineering and operator.; the active mode is visible in the status bar.
- `Expert/Suspend`: the suspend menu item causes a suspension of the ingestion of logs. When LC is suspended it does not listen from incoming logs (i.e. the logs loaded or received from the logging channel are discarded). Setting LC in suspend mode do not cause the disconnection from the logging channel.
- `Expert/Preferences`: open the dialog to setup the preferences. It allows to select the max number of logs to keep in memory (default is 100.000).
- `Expert/Engine filters`: allow to set filters at engine level.

3.3 Log table

This is the central area of the Logging Client. Any log received by the application is displayed here. The view is highly configurable to allow filtering and sorting. The default view on startup is set to be unsorted and unfiltered.

The table shows the logs one per row. If the user selects a line, all the field of the logs are displayed in the Detailed info panel at the right side.

If a log has additional informations added programmatically or by the logging client user, these informations are also displayed in the Detailed Info panel . See the section Additional Info for further info about this topic.

It is possible to order the logs by pressing on one of the column header. Pressing twice changes the ordering from ascending to descending and vice versa. A column can be placed to a different position by pressing the right mouse button over the column header and dragging the column to the desired position.

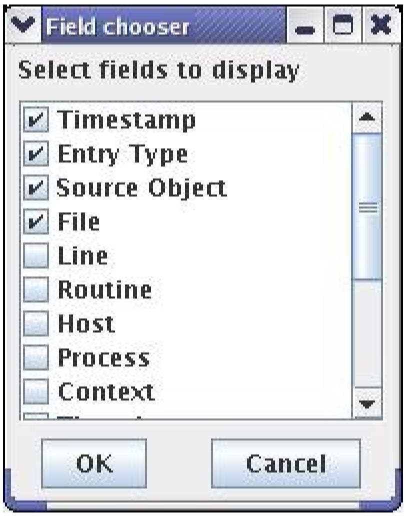
When the user selects one or more log entries and presses the right mouse button, a popup menu allows to:

- show the error trace in the error browser (if the log has a valid stack ID)
- Copy the selected logs in the clipboard[1]
- Save the selected logs in a file
- Add additional info to a log entry
- Add the content of a field to the additional info of the selected log (the same can be done simply pressing selecting a line in the table too)

3.4 Field Chooser

The field chooser dialog is opened by selecting the View\Field menu item. Note that the same dialog is displayed by pressing the right mouse button over the table header of the error browser.

[1] By copying the logs in the clipboard the user can past the logs in another application, for example a text editor. The logs are pasted in XML format.



☑ Timestamp
☑ Entry Type
☑ Source Object
☑ File
☐ Line
☐ Routine
☐ Host
☐ Process
☐ Context

Field chooser is used to show or hide different log entry fields i.e. colmns of the table. When a column is added to the table it is always placed at the right and if the column is removed, all remaining will shift automatically to fill its position. In the field chooser checking a check box before the column name means it will be shown, and un-checking it means it will be hidden. All changes occur after pressing the OK button.

A full explanation of all the fields may be found in Logging and Archiving (http://www.eso.org/projects/alma/develop/acs/Releases/ACS_4_1/Docs/Logging_and_Archiving.pdf)

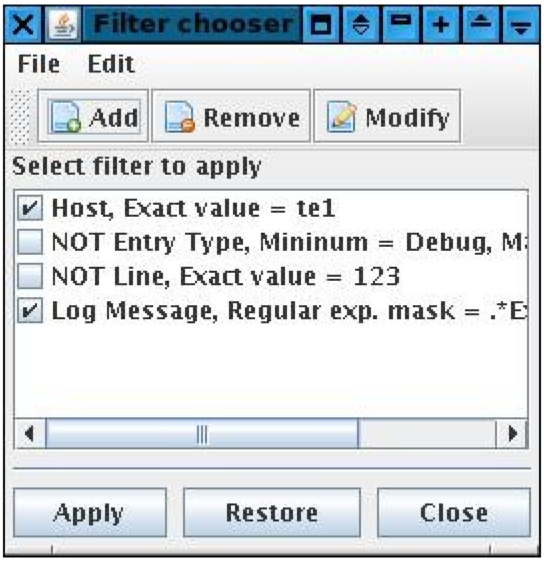
3.5 Filters

A button in the toolbar and the View/Filters menu item show the filters configuration dialog that allows to load, save, edit, configure the filters. This dialog is not modal i.e. it is possible to edit filters while the logs are

Filters allow displaying only log entries which match specified criteria. The engine does not discard the messages that do not match with the filters. It instead hides those messages to the user in such a way when the filters are disabled all the messages will be displayed again in the table. The content of each log is compared in AND with the defined filters i.e. an entry in the table for that log is shown only if the log matches all the filters.

The dialog owns a table in the center containing the summary of all defined filters. It contains one entry for each filter consisting of a check box and a brief description of the filter. Each filter can be enabled or disabled with such a check box. In this way the definition and the activation of a filter happens in two separate steps.

Caution should be used when enabling filters, since some combinations may filter out all the entries and the main table will appear empty. As said before, the filters on the entry type are evaluated in AND between them and with the log level selected in the toolbar. The statistic dialog shows the the number of logs in memory and the number of logs filtered out.



Select filter to apply

☑ Host, Exact value = te1
☐ NOT Entry Type, Mininum = Debug, M:
☐ NOT Line, Exact value = 123
☑ Log Message, Regular exp. mask = .*E

The toolbar allows the user to add, remove or edit a selected filter. The menu bar contains two entries: File, to load and save filters; Edit to enable, disable or clear all the filters.

The filter control panel manages the filters locally until the user presses the Apply button. Only at that moment the filters are applied to the logs displayed in the main window. This is done to allow the user to perform several changes to the filters before seen their effect in the main table. There is another reason for that: performance. In fact, when the Apply is pressed, the entire set of logs is regenerated and such operation can be slow depending on the number of logs and filters.

The functioning of the filter control panel is the following:

- when the panel is shown, it shows the active filters in the table
- the user can then edit, add, remove, activate, deactivate the filters visible in the dialog (these changes do not reflect to the main application)
- the user can save the filters on a file
- the user can apply the filters to the table of logs. At that moment the filters are applied to the table. If the user does not press such a button, the filters are not applied and can be lost if he/she presses the close button (however, to avoid such a risk, an apposite warning is shown before closing).
- The restore button, delete all the changes done before showing the panel (i.e. It will restore the situation at startup).

Another way to explain the functioning of the filter panel is to think that the panel works on a different copy of the filters used by the main application. When the user presses Apply, the filters in the panel are passed to the main application (causing an invalidation of the displayed logs).

When the user presses Restore, the active filters are copied back in the dialog allowing the user to undo the changes he/she made since the filter dialog was shown.

If the user forget to press the Apply, the filters in control panel will not be passed to the main application. If the user closes the dialog without pressing the apply button, an apposite confirmation dialog is shown. However, such a dialog is not shown if the user save the filters before closing.

The buttons in the toolbar are:

- Add: presents a dialog window to define a new filter.
- Remove: remove the selected filter
- Modify: show a panel to edit the selected filter

3.5.1 Loading and saving filters

A set of filters may be saved and loaded in a XML file. The file menu shows the usual items for this kind of operations: Load, Save and Save As...

By selecting Load or Save As menu item a file chooser dialog is shown to allow the user to select an existing file or to create a new one.

The Save item is enabled only after a file has been chosen for loading or saving.

By selecting the Load item and if a set of filters is already defined, a dialog box asks the user if he/she wants to merge the filters defined in the file with the existing ones.

The state of the filter (enabled/disabled) is saved together with the filters. If the state is not specified in the file, then the filter is loaded as enabled. This situation can happen while loading old file of filters that do not store the state. The state of the filters is saved by default starting from ACS 6.0.3.

3.5.2 Creating and modifying filters

Each log entry is composed of several fields each of which is or is not shown in the table of logs depending of the user preferences. We can say that each log is shown in a row of the table. Each field of a log is displayed in the relative column.

The user can define which columns he/she want to see in the table. The searching allows the user to look for specific values in the fields (i.e. the columns) of the logs. The filters allow the user to select which logs he wants to see in the table depending on the values of their fields (i.e. the values in their columns).

Each filter checks the value of the specified field of the log (i.e. the value of a given column) against a pattern that can be defined in different ways depending on the type of the data contained in the field itself. It means that each filter only checks the value of one field. There are no filters checking the content of more then one field of the logs.

The filtering mechanism allows defining more then one filter regardless if they check the content of the same field or not. The user must be very careful defining filters to avoid filtering out all the logs. If more then one filter is defined, they are checked in AND. If one filter is used to hide filters depending of the content of one of its fields, more filters allow checking the content of several fields.

The engine checks each log to display in the table against the filters. If the log matches with one filter then it is compared to the next one. If it does not match, it is not shown in the table and is not checked against the other filters. If and only if the log matches with all the filters, it appears in the table.

To create a filter, click the Add button in the toolbar of the Filters Chooser dialog. This will open a new window, the Filter properties. To define a filter, simply check the conditions to validate and click OK. Clicking the Cancel button will close this window without making any modifications. The same window is shown when modifying a filter. This can be done by selecting a filter in Filters window and clicking the Modify button. The only difference from creating a new filter is that the fields will already be filled in.

As we said upon, the filters are based on the fields of a log entry. The first step defining a filter is to select which field of the log entry the filter must check. The supported filters are:

- Timestamp
- Entry type
- Source object
- File
- Line
- Routine
- Host
- Process
- Context
- Thread
- Log ID
- Priority
- URI
- Stack ID
- Stack level
- Log message
- Audience

- Antenna
- Array

When the user selects the field to filters, the rest of the dialog adjusts to enter the relevant filtering for such a field. For example, if the user wants to filter by date/time the dialog shows controls to insert date/time. If the user instead selects to filter by entry type, the bottom side of the window shows the list boxes with the different entry type.

The editors for each possible entry type are very intuitive and do not need any further explanation.

There are basically two different types of editors: one allows defining a range of logs while the other allows matching the values of the field. Which one of the two is used depends on the type of the field to filter. For example it is possible to filter the logs generated between a two different dates/times or having the type between Info and Warning. Instead, while filtering for source object, it is more useful to look at the content of a string. It is also possible to match if the content of the field is exactly the same of a given value. It is not possible to define in the same time a range and an exact value.

To select a range of logs, the editor contains two check boxes: From and To. From is the starting of the range while from is the end. One of the two must be selected otherwise the filter has no meaning. If only To is selected, the filter matches all the log up to the value defined in the To field. If only From is selected, the filter matches all the log entry having that value or a greater one in the filtered field.

For example, if the user is creating a filter for the entry type, the following cases apply:

1. he/she selects only the From checkbox, selecting as minimum for example info: the log entries that matches that filter are all the log entries having an entry type equal or greater then Info

2. he/she selects only the To checkbox, selecting for example Error: the log entries that matches that filter are all the log entries having an entry type equal or lower then Error

3. he/she selects the From checkbox, choosing Debug and the To checkbox choosing Warning: the log entries that matches that filter are all the log entries having entry type between Debug and Warning (inclusive).

4. he/she selects the Exact value, selecting Alert in the dropdown list: the log entries that matches that filter are all the log entries having entry type of Alert

The editor contains another checkbox called "*Discard entries matching the filter*". Till now, we have said that each filter allows the user to select all the log entries that do matche the given filter. Sometimes instead, it is useful to select all the log entries that *do not* match the filter. This is done by selecting the "Discard…" checkbox. In short, this checkbox works like a NOT. The effect is that the entries that are filtered out are all the logs that do not match the filter. Let's come back to our example of an entry type filter where the user selected the Exact value checkbox and the Alert in the drop list. We have seen before that the effect of such a filter is to show only the logs having an entry type of Alert. If the user selects the “Discard…” check box, the logic applies on the contrary: the logs shown in the table are all the logs having an entry type *not* equal to Alert.

If there are two filters with the “Discard…” checkbox selected they are applied in AND as usual i.e. the logic is to show all the logs that do *not* match the first filter *and* do *not* match the second filter.

For string values it is also possible to define a regular expression to match. A regular expression is a string, a pattern, used to describe a set of strings. The filter matches all the string that matching to the regular expression. The web is full of documentation about regular expressions (see for example http://en.wikipedia.org/wiki/Regular_expression or http://www.regular-expressions.info/). In the linux environment the man pages for grep contains a description of the syntax of the regular expressions. If you have a KDE installed in your linux box, a program called kregexpeditor allows editing and testing regular expressions.

This is not the right document to present a full description of regular expression syntax but we’ll give some examples of the most used cases.

A regular expression is a case sensitive string describing strings, it is a pattern; we say that all the strings that adhere to the pattern of the regular expression, *match* that regular expression. In the simplest case, a regular expression is composed of a sequence of character. In this case, the set of strings matching such a regular expression is composed of only one string equals to the regular expression. For example, if the regular expression is “jlog” the only string that matches with such a regular expression is “jlog”.

The power of regular expressions is that some characters have a special meaning:

- dot (.) means any character . For example the regular expression “jlog.” Matches with “jlog1”, “jlogA”, “jlog]”, “jlog.” and so on. If you want to match exactly the dot character, you have to escape the dot in the regular expression with a backslash (\[2]). For example the regular expression “jlog\.” Matches only with “jlog.”

- Square brackets mean one of the characters between the brackets. For example the regular expression “jlog[124]” matches only with “jlog1”, “jlog2” and “jlog4”. It is also possible to define a range of characters using the dahs (-) like for example in “jlog[1-3]” that matches “jlog1”, “jlog2” and “jlog3”

- * is a repetition operator that means zero or more instances of the preceding item. For example “jlog*” means “jlo” followed by 0 or more ‘g’ because the * applies to the last character. In this example, the strings matching “jlog*” are “jlo”, “jlog”, “jlogg” and so on.

- + is another repetition operator that means one or more instances of the preceding item. For example “jlog+” means “jlo” followed by 1 or more ‘g’. In this example, the strings

[2] All the special characters must be escaper with a backslash if you want your regular expression matches exactly that character.

matching “jlog+” are “jlog”, “jlogg” and so on. Note that in this case “jlo” does not match.

- ? is another repetition operator that mean none or one instance of the preceding item. For example, the regular expression “jlog1?” matches only with “jlog” and “jlog1”
- Braces, (), groups items in the regular expressions and can be very useful when used together a repetition operator. For example the regular expression “(jlog)+” matches the strings containing a sequence of one or more “jlog” like for example “jlog”, “jlogjlog”, “jlogjlogjlog”…
- Vertical bar | means an alternative. For example, "gray|grey" can match "gray" or "grey". Note that the previous regular expression could be shortened to the equivalent "gr(a|e)y",

The following table contains some common examples.

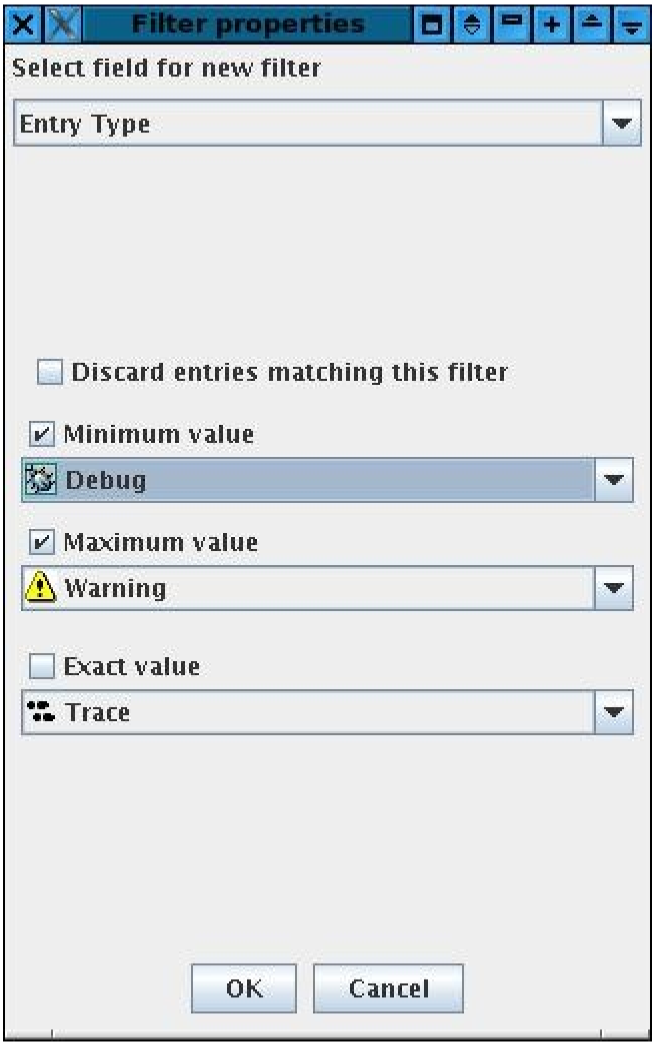
Regular expression	**Examples of matching strings**
.*	Matches with the repetition (*) of any character (.) i.e. any string. Matching example are: “”, “container”, “123”, …
[Cc]ontainer	Matches with “container” and “Container”
.*[cC]ontainer.*	Matches with all the strings containing “container” or “Container” (“container”, “maciContainer”, “macicontainer243”, “ContainerAB”…)
Mount[0-9]+	Matches with the string Mount followed by a number (“Mount9”, “Mount132”…)
maciContainer| bilboContainer	Matches with “bilboContainer” and “maciContainer”

3.5.3 Example of filters

The figure below shows a filter for the entry type of the logs, as you can see in the drop list in the upper side of the dialog.

The max and min checkboxes of the entry type are selected together with the Debug and Info entry types.

The logs shown in the table. i.e. the logs that match this filter, are all the logs having type Debug, Info, Notice and Warning.



Filter properties

Select field for new filter

Entry Type

□ Discard entries matching this filter

☑ Minimum value

Debug

☑ Maximum value

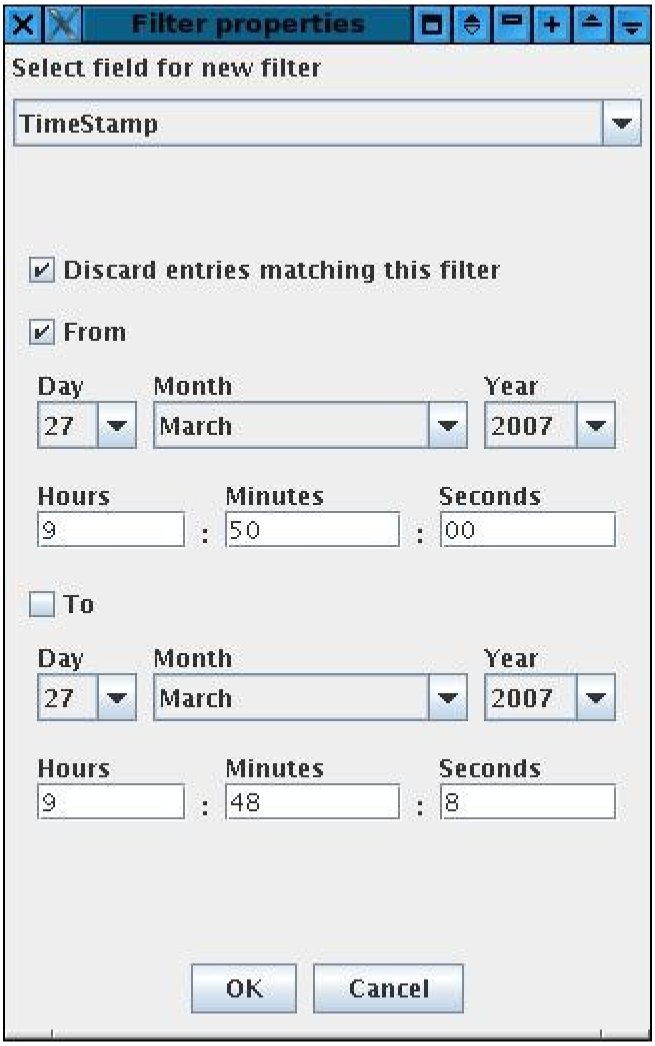
Warning

□ Exact value

Trace

Next picture shows a filter by timestamp. In this case the “Discard…” checkbox is also selected. The filter matches with all the logs having a time stamp starting from the 09:50 of march the 27th 2007, i.e. all the logs generated after that date. However, as the “Discard…” checkbox is selected, the logs shown in the table are those that are *not* generated after that date/time. Or, if you prefer, the logs shown in the table are the logs generated before 27/03/2007 at 09:50.

Note that the same filtering can be obtained deselecting the “Discard..” and the From checkboxes, selecting the To checkbox and setting there the same date now written in the From widgets.



Select field for new filter

TimeStamp

☑ Discard entries matching this filter

☑ From

Day	Month	Year
27	March	2007

Hours	Minutes	Seconds
9	50	00

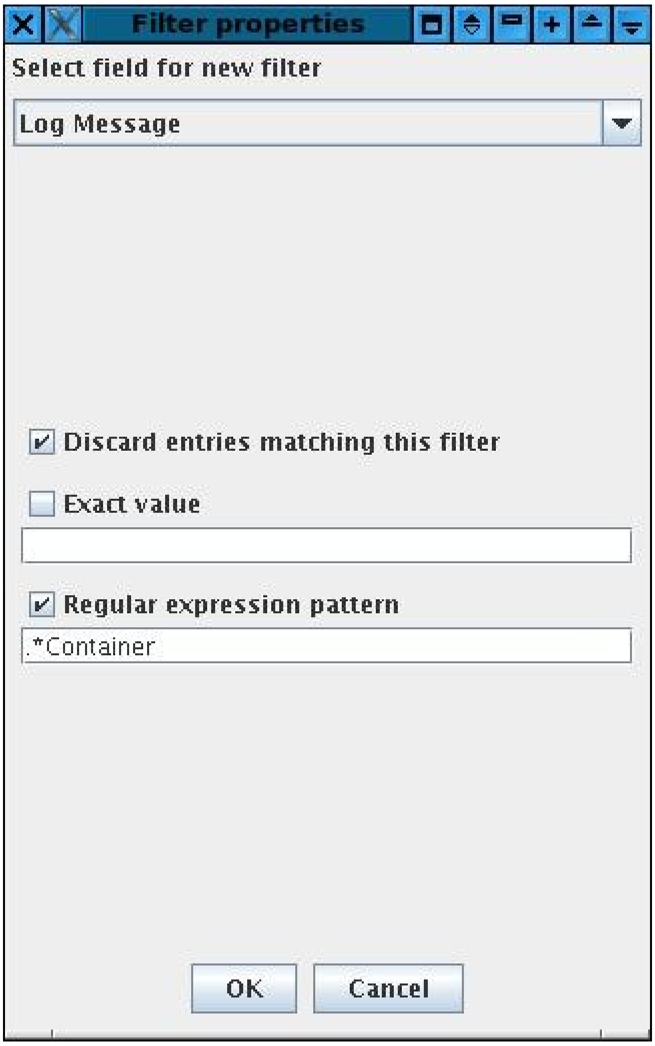
☐ To

Day	Month	Year
27	March	2007

Hours	Minutes	Seconds
9	48	8

The next picture shows a filter for the content of a log message. The “Discard…” checkbox is again selected and the match criteria is a regular expression. We have seen upon that this regular expression matches with all the string terminating with “Container” like “maciContainer”, “bilboContainer” or “Container” itself.

The effect of this filter is to filter out (hide) all the logs whose message does *not* match with the regular expression. For example the log with content “Instantiating MOUNT” will be shown, while the logs “maciContainer started” and “Shutting down bilboContainer” will not be visible.



Select field for new filter

Log Message

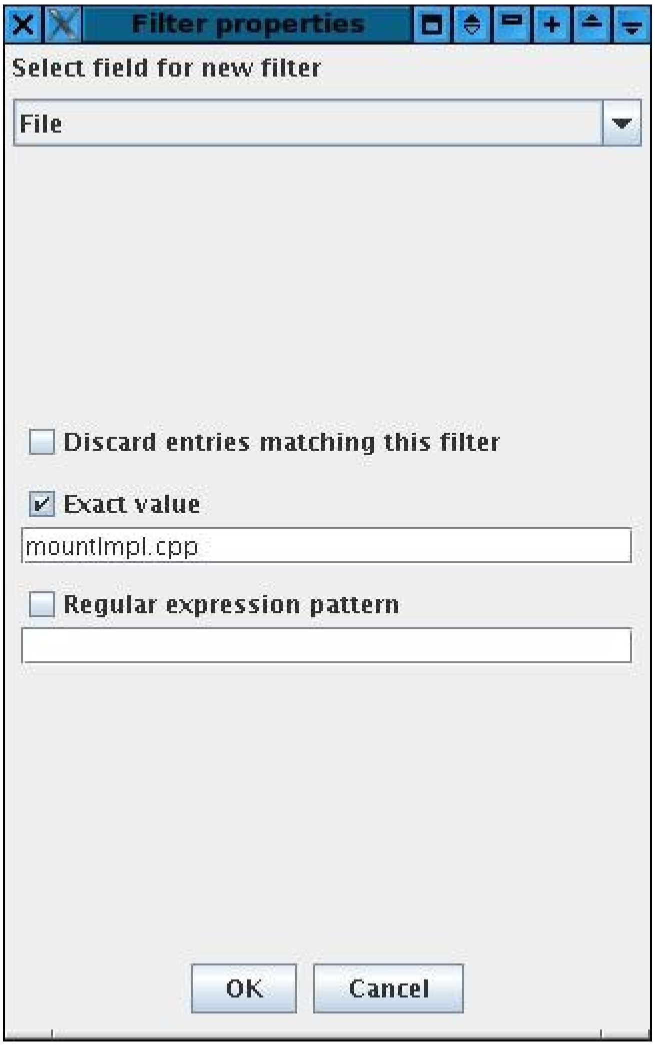
☑ Discard entries matching this filter

□ Exact value

☑ Regular expression pattern

.*Container

The last example shows an exact filter for the File field. In this case, the logs that will be shown in the table are only the logs having “mountImpl.cpp” in the File field.



Select field for new filter

File

□ Discard entries matching this filter

☑ Exact value

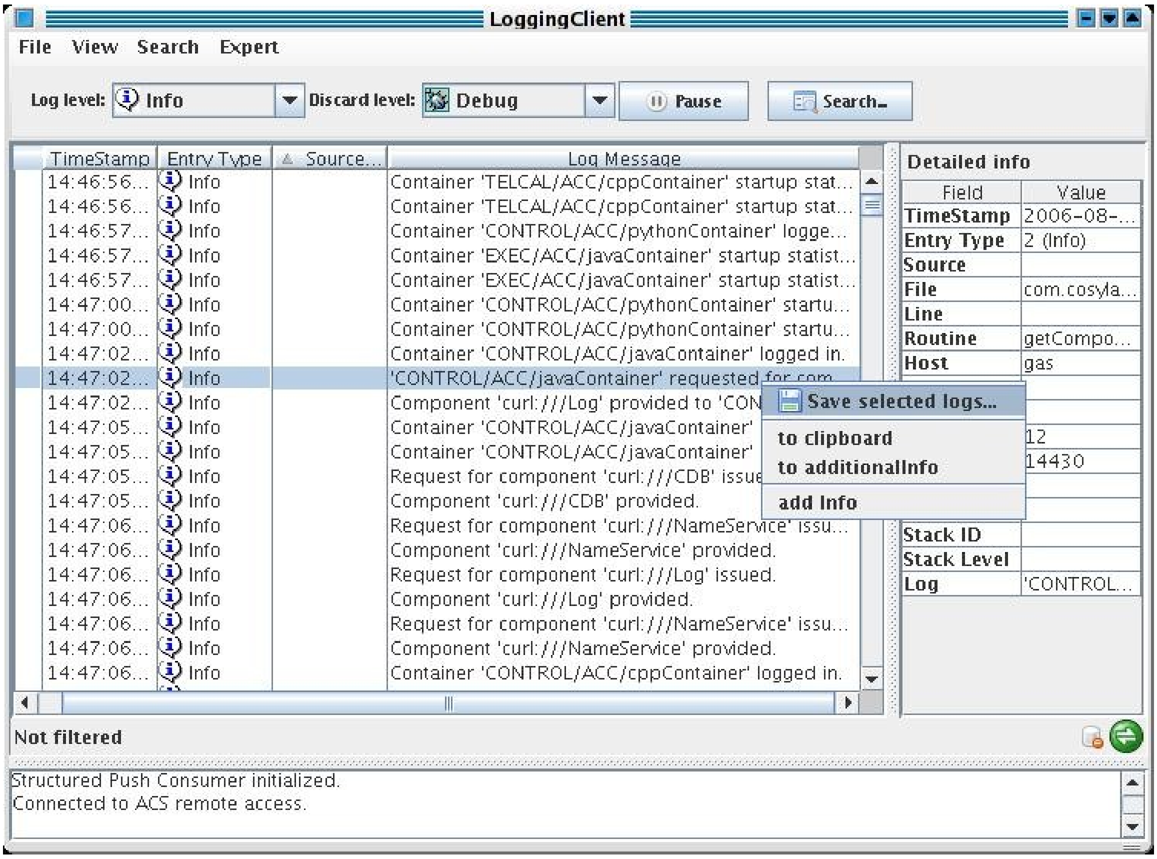
mountImpl.cpp

□ Regular expression pattern

3.6 Additional log information

According to the Logging and Archiving specification (KGB-SPE-01/04) document, each log entry can also contain unspecified amount of additional information. The detailed info panel shows all the fields of a selected log together with its additional informations (if any). The panel itself provides no other functionality.

The ℹ icon appears in the left side of each row whose log entry contains additional information. The purpose of this icon is to signal the user that, by selecting the row, additional info will be displayed in the right panel. In the following picture you can see that both the logs in rows 11 and 18 contain additional info.



TimeStamp	Entry Type	Source...	Log Message
14:46:56...	Info		Container 'TELCAL/ACC/cppContainer' startup stat...
14:46:56...	Info		Container 'TELCAL/ACC/cppContainer' startup stat...
14:46:57...	Info		Container 'CONTROL/ACC/pythonContainer' logge...
14:46:57...	Info		Container 'EXEC/ACC/javaContainer' startup statist...
14:46:57...	Info		Container 'EXEC/ACC/javaContainer' startup statist...
14:47:00...	Info		Container 'CONTROL/ACC/pythonContainer' startu...
14:47:00...	Info		Container 'CONTROL/ACC/pythonContainer' startu...
14:47:02...	Info		Container 'CONTROL/ACC/javaContainer' logged in.
14:47:02...	Info		'CONTROL/ACC/javaContainer' requested for com...
14:47:02...	Info		Component 'curl:///Log' provided to 'CON
14:47:05...	Info		Container 'CONTROL/ACC/javaContainer'
14:47:05...	Info		Container 'CONTROL/ACC/javaContainer'
14:47:05...	Info		Request for component 'curl:///CDB' issue
14:47:05...	Info		Component 'curl:///CDB' provided.
14:47:06...	Info		Request for component 'curl:///NameService' issu...
14:47:06...	Info		Component 'curl:///NameService' provided.
14:47:06...	Info		Request for component 'curl:///Log' issued.
14:47:06...	Info		Component 'curl:///Log' provided.
14:47:06...	Info		Request for component 'curl:///NameService' issu...
14:47:06...	Info		Component 'curl:///NameService' provided.
14:47:06...	Info		Container 'CONTROL/ACC/cppContainer' logged in.

Field	Value
TimeStamp	2006-08-...
Entry Type	2 (Info)
Source	
File	com.cosyla...
Line	
Routine	getCompo...
Host	gas
	12
	14430
Stack ID	
Stack Level	
Log	'CONTROL...

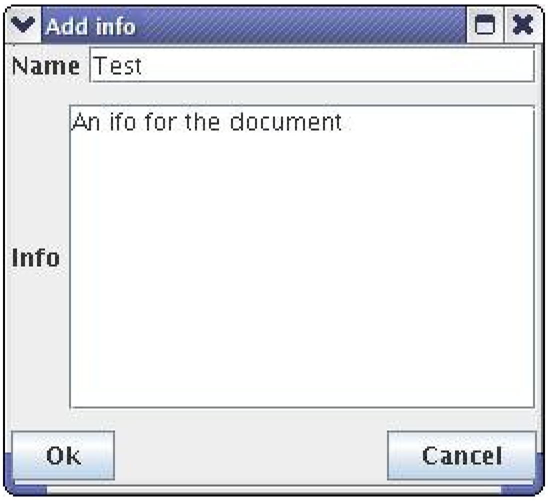
```
Structured Push Consumer initialized.
Connected to ACS remote access.
```

The additional informations are couples <Name, Value> and a log entry may have none or many information of this type. There are three ways to add information to a log entry:

1. programmatically with the addData method specifying the name and the value (the picture show an entry of this type in the first position; the developer added the type of the exception received when the method failed)

2. by pressing the right mouse button over a cell of a log and selecting the toAdditinalInfo item. The content of the cell is added in the Value field and "Generated by jlog" will be the Name (it is the second entry in the picture upon). Note that after inserting this entry from the entry type cell of the selected log, the user hided the Entry type column but the information is still visible in the second line of the Additional Info panel.

3. By pressing the right mouse button over a log and selecting add Info, a new dialog is shown to the user where he can the <Name,Value> pair to be added in the panel. The third line of the Additional info panel shows the result produced by pressing ok in the panel in the picture.



Name: Test

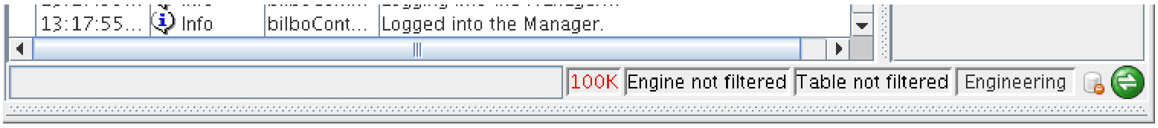
Info: An ifo for the document

The additional infos of each log entry become part of the log itself and thereafter will be saved together with the logs (`File/Save` or `File/Save as..`). This allows the users to add some more details to the snapshot and it is very useful when the same log file is shared by several developers.

In the picture, the detailed info panel is filled by the data of the selected log. In the bottom of that panel the Additional informations are visible, identified by the bold Additional title. The key of the additional data is shown in the first column and the value in the second.

3.7 Status bar

The status bar report some useful information about the current state of the application.



13:17:55... Info bilboCont... Logged into the Manager.

100K | Engine not filtered | Table not filtered | Engineering

At the left side a progress bar is activated when a long lasting operation is performed like for example when sorting the table.

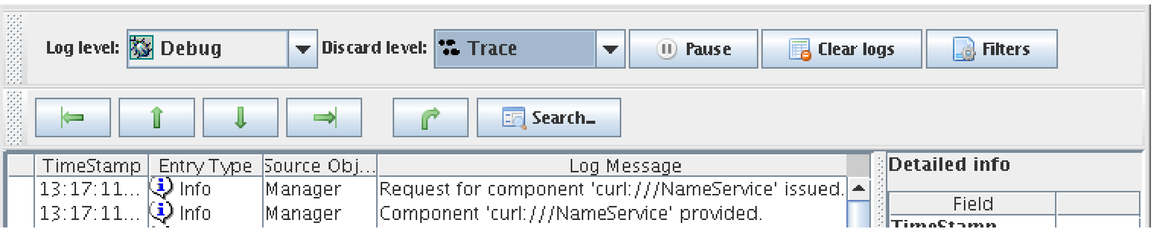
The next field, in red in the picture above, say if the number of logs managed by jlog is limited. If it is the case the number apears in red.

Two fileds shows if there are table and engine filters active. If the filters are active the label related widget is painted in red.

The next widget says the selected operating mode, it can be *Operator* or *Engineering*. Changing the mode causes jlog to apply a defined set of filters at engine level. Operator mode is thought to be used inside the OMC.

3.8 Toolbar

The two-lines toolbar allows to perform some of the most useful and requested operations in a short time. Each of the two lines can be detached from the main window by pressing the left mouse button at the gray area at left side of each line.



Log level: Debug

Discard level: Trace

TimeStamp	Entry Type	Source Obj...	Log Message
13:17:11...	Info	Manager	Request for component 'curl:///NameService' issued.
13:17:11...	Info	Manager	Component 'curl:///NameService' provided.

By selecting an item in the left combo box, the user can select the log level so that only the messages with a higher priority then the selected log level will be shown in the main window. The log level functionality is obtained by defining a filter at system level. The user shall keep in mind that it is also possible for him to define filters based on the same property: those filters will be evaluated in AND with the selected log level. If the log level is selected to trace then all the messages appear in the table.

To reduce the load of the application it is possible to define a discard level: the discard level let jlog discard all the logs having a log level equal or less the the selected discard level. The selection operated in the discard combo box applies directly to the engine.

There is a great difference between the log level and the discard level. The log level works like a filter, showing or hiding the logs in the table depending of their entry type. The discard level instead causes the application to ignore the incoming logs if their entry level is equal or less then the selected discard level.

There is no way to get back a log that has been discarded because of the setting in the discard level combo box. The user should use this option carefully. To disable the discard level, select None.

The log level defaults to Info; the discard level defaults to Debug.

By pressing the Pause button, all the received logs are queued and their insertion in the main table is delayed until the button is pressed again. This feature freezes the table of logs allowing the user to read the status of the system at a given time without dealing with the incoming logs. As the logs are queued, they will be displayed in the table when the user un-pauses the application.

The Suspend item of the Expert menu works in a similar way of the Discard level. Both the features discard logs when they arrive and before being ingested and added in the table of logs. The only difference is that the discard level ignores the logs following a criteria, a kind of filter based on the entry type of each received log. The suspend function instead discard all the logs regardless of their content.

The clear logs button remove all the logs from the table included the hidden logs i.e. the logs filtered out. By pressing this button the cache is also cleared and a lot of memory and disk resources are freed.

The filters button is a short cut for the Filter menu item.

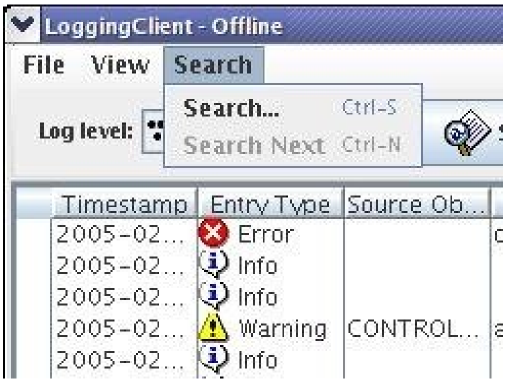
The second line of the toolbar contains widgets to simplify the browsing of logs. The green arrows select the first, previous, next and last line of the table respectively.

The fifth and last arrow make the selected log visible. This can be useful after resorting the logs. Let's suppose you have selected a log and then sorted the table by entry type. When the table is sorted it can happen then the selected log has been moved in a different position and it is not visible. By pressing the arrow the table is scrolled to make such a log visible.

The Search button is in the right side of the toolbar: by pressing this button the search dialog box is shown.

3.9 Search for logs

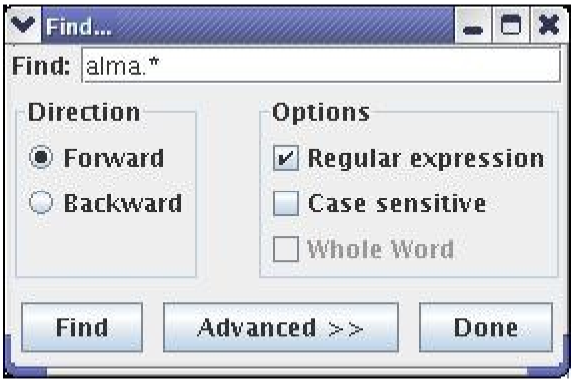
The search feature allows the user to search the logs for a string or a regular expression. The search dialog can be displayed by pressing the button in the toolbar, by pressing CTRL+S in the keyboard or by selecting the `Search...` menu item as shown in the following figure:



Timestamp	Entry Type	Source Ob...
2005-02...	Error	
2005-02...	Info	
2005-02...	Info	
2005-02...	Warning	CONTROL...
2005-02...	Info	

The search dialog box is used to define the parameters of the search. The engine searches for the string or the regular expression inserted in the `Find` text box. The search is performed in forward or backward order depending by the selected item in the Direction box and starting from the selected row in the log table of the main window[3].

[3] If there is no line selected in the table of the logs the engine begins the scan starting from the first or the last row if the direction is forward or backward respectively.



Find: alma.*

Direction
● Forward
○ Backward

Options
☑ Regular expression
□ Case sensitive
□ Whole Word

The option pane contains the following switches:

- `Regular expression`: means that the string inserted in the upper text field is a regular expression instead of a standard string (the picture shows this case)

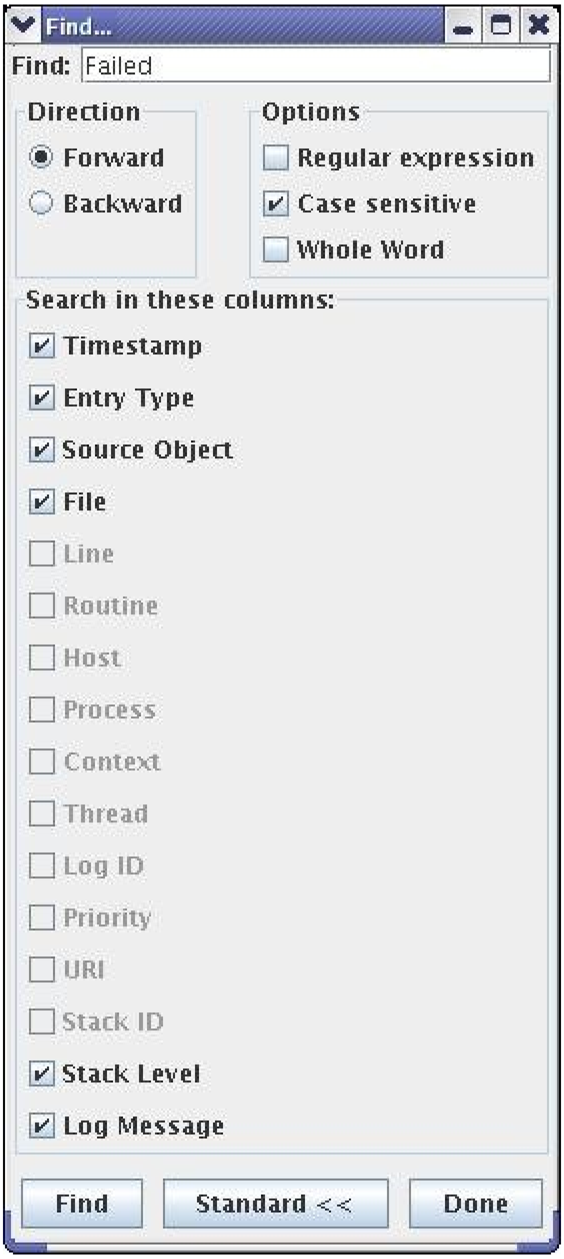
- `Case sensitive`: the search is performed comparing or not comparing the cases of the strings

- `Whole word`: if this item is checked the engine looks for a string equals to the text written in `Find` otherwise it looks for a string containing the string written in the text field. This field is disabled if the string is a regular expression.

When the user presses the `Find` button, the engine starts the search from the selected row in the table of the logs of the main window. The search is done by comparing the string or regular expression in the text field with the string in each visible field of the table. A dialog is shown if no log matching the criteria is found otherwise the row of the table containing the matching log is selected and shown in the table of the main window.

The `Done` button closes the dialog box but the search criteria is not lost i.e. once the user has introduced a search criteria in the dialog box, it is possible to search for a log by pressing CTRL+N even if the dialog box is not visible anymore.

As we said before, the search is performed by looking for the string (or the regular expression) only into the fields of the logs that are visible in the table of the logs: the user can show or hide those fields by selecting the View/Fields menu item. It is possible to explicitly select the fields to look into during the search. This is done by pressing the `Advanced` button: another panel is shown in the search dialog box:



Find...

Find: Failed

Direction

● Forward

○ Backward

Options

□ Regular expression

☑ Case sensitive

□ Whole Word

Search in these columns:

☑ Timestamp

☑ Entry Type

☑ Source Object

☑ File

□ Line

□ Routine

□ Host

□ Process

□ Context

□ Thread

□ Log ID

□ Priority

□ URI

□ Stack ID

☑ Stack Level

☑ Log Message

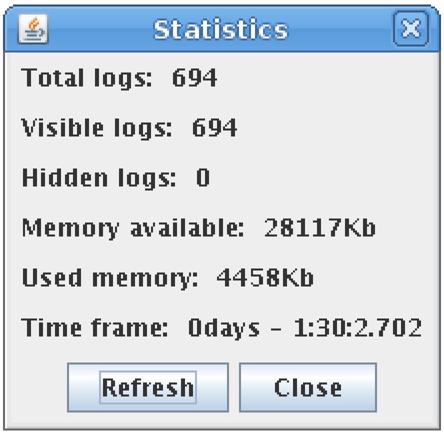
The panel shows all the fields of the logs. The fields that are not visible in the table are disabled because it is not possible to search into these fields. The visible fields instead are all enabled (and checked for default): the string is searched in all and only the checked fields. The user can select the field to search into by checking or unchecking the relative checkbox.

For example, in the figure above if the user wants to search for string into the `Source Object` column only then he has to uncheck all the fields but the `Source Object`. If he instead wants to search a string also in the `Routine` column the he has first to make visible the `Routine` column in the table[4] and then to check the Routine checkbox in the advanced panel.

[4] By selecting the View/Fields menu item and making visible the Routine column in the table of logs, the Routine check box in the advanced panel will be enabled.

4 Statistics

The statistics dialog appears by pressing the `View/Statistics` menu item.



Total logs: 694

Visible logs: 694

Hidden logs: 0

Memory available: 28117Kb

Used memory: 4458Kb

Time frame: 0days - 1:30:2.702

The dialog shows some information about the loaded logs. In the picture upon, you can see that jlog is managing a total of 694 logs. The total and available memory is also shown.

Visible logs are those displayed in the table while hidden logs are the logs filtered out of the table. Note that while reading logs it can happen that the number of hidden logs is not zero even if there are no table filters active. It happens because some logs are still hidden while the application find the position in the table where they need to be placed to respect the selected ordering. This is a temporary situation and the number quickly goes to zero as soon as all the logs found their position.

The time frame is the amount of time visible in the window (i.e. the time between the oldest and the newer log in the GUI).

By pressing the Refresh button, the data displayed are updated.

The dialog can stay open during the normal functioning of the logging client.

5 Connection icon

The status of the connection of the LC is displayed in the status bar as a string. An apposite icon is present in the bottom right side of the application window to show the user the status of the connection with a quick look.

The icon changes the color and/or the picture depending on the status of the connection.

The green icon means that the LC is connected to the logging channel and working properly.

The red icon means that the application is disconnected from the logging client. This is the icon shown while working offline. The green icon becomes red if the connection is lost at run time.

The gray icon is shown when the application is trying to connect to the logging channel. It is usually displayed at startup. It can also be shown if the connection has been lost and the application is trying to reconnect to the channel if the autoreconnect menu item is selected.

The blue icon is shown only when the application is connected to the logging channel and means that there is a delay between the loogs received and the logs displayed in the table. This happen when the number of logs read by the logging channel per second is too high and the application is not able to cope with such a load. The blue icon advise the user that he has to wait some time to see all the received logs displayed in the table.

Please note that even if the blue icon is shown no logs are lost. In case of overload, LC queues the incoming logs in order to insert the logs in the table when the load is acceptable. The icon usually blinks from green to blue during normal operations.

6 Database connection

The LC tries to connect to the database at startup. An icon in the bottom right side of the dialog shows the status of the connection:

- Means that the connection with the database is unavailable
- Mean that the connection with the database is active
- Means that a query is in progress

By selecting the "Load from database" menu item the following dialog allows the user to submit queries to the database.

The user can define a maximum number of logs to load with the query i.e. the number of logs returned by the database.

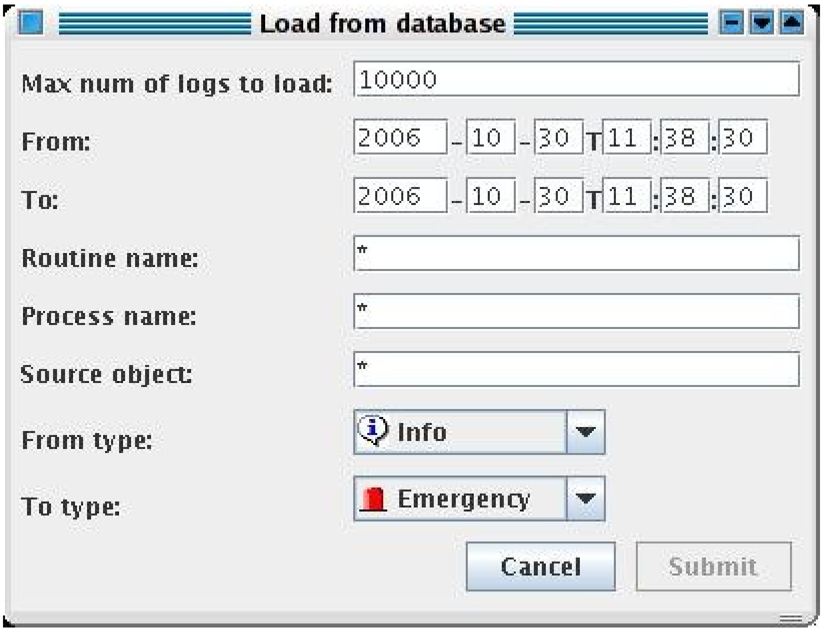
The other fields of the dialog are very intuitive and must be used to submit the desired queries. A "*" for the routine name, process name or source object means that the database will return logs without checking the content of such fields.

When the database is available, the Submit button is enabled and cause the sending of the query to the archive. The icon in the main GUI changes its status to busy until the logs are retrieved and flushed in the main GUI. The status of the icon then reverts to available.

In case of an error submitting the query, the database connection is invalidated and the user can't submit further queries to the database.

In order to use the database, the LC checks at startup if the code from ARCHIVE is available. If the code is found it tries to connect to load the classes. If this operation is successful then the database will be available.

There is a chance that even if the code is loaded the database is not available. Unfortunately in this first version, the only way to discover the problem is to try to submit a query.



Load from database

Max num of logs to load: 10000

From: 2006 - 10 - 30 T 11 : 38 : 30

To: 2006 - 10 - 30 T 11 : 38 : 30

Routine name: *

Process name: *

Source object: *

From type: Info

To type: Emergency

It is also possible to get logs out of the database with a command line tool provided by ARCHIVE: `archiveLoggingCli`.

The documentation about the usage of this tool is available at the following twiki page:

`http://almasw.hq.eso.org/almasw/bin/view/Archive/ArchiveTools`

7 Error dialog

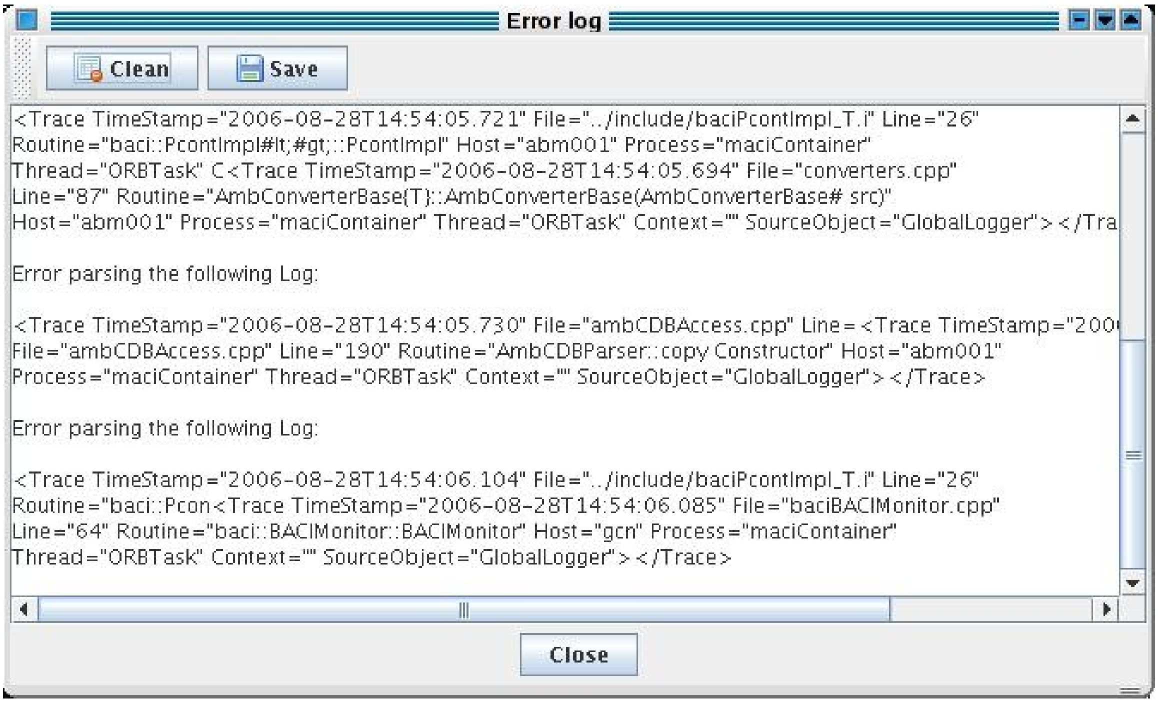
When an error happens at run time, an apposite dialog is shown with a description of the logs. Note that this window shows the errors occurred in the application like for example an error parsing a log and must not be confused with an error trace displayed by the error browser that is generated by looking at the content of the logs in the table.

The normal functioning of the application continues even if the log dialog is visible and the user is browsing its content.

Two buttons allows the user to clear and save the content of the error dialog.

If the user closes the dialog pressing the Close button or the apposite widget, the content of the text is not deleted.

It is possible to show the error dialog at any instant, by selecting the `View/Error log` menu item.



```
<Trace TimeStamp="2006-08-28T14:54:05.721" File="../include/baciPcontImpl_T.i" Line="26"
Routine="baci::PcontImpl#lt;#gt;::PcontImpl" Host="abm001" Process="maciContainer"
Thread="ORBTask" C<Trace TimeStamp="2006-08-28T14:54:05.694" File="converters.cpp"
Line="87" Routine="AmbConverterBase{T}::AmbConverterBase(AmbConverterBase# src)"
Host="abm001" Process="maciContainer" Thread="ORBTask" Context="" SourceObject="GlobalLogger"></Tra

Error parsing the following Log:

<Trace TimeStamp="2006-08-28T14:54:05.730" File="ambCDBAccess.cpp" Line=<Trace TimeStamp="200
File="ambCDBAccess.cpp" Line="190" Routine="AmbCDBParser::copy Constructor" Host="abm001"
Process="maciContainer" Thread="ORBTask" Context="" SourceObject="GlobalLogger"></Trace>

Error parsing the following Log:

<Trace TimeStamp="2006-08-28T14:54:06.104" File="../include/baciPcontImpl_T.i" Line="26"
Routine="baci::Pcon<Trace TimeStamp="2006-08-28T14:54:06.085" File="baciBACIMonitor.cpp"
Line="64" Routine="baci::BACIMonitor::BACIMonitor" Host="gcn" Process="maciContainer"
Thread="ORBTask" Context="" SourceObject="GlobalLogger"></Trace>
```

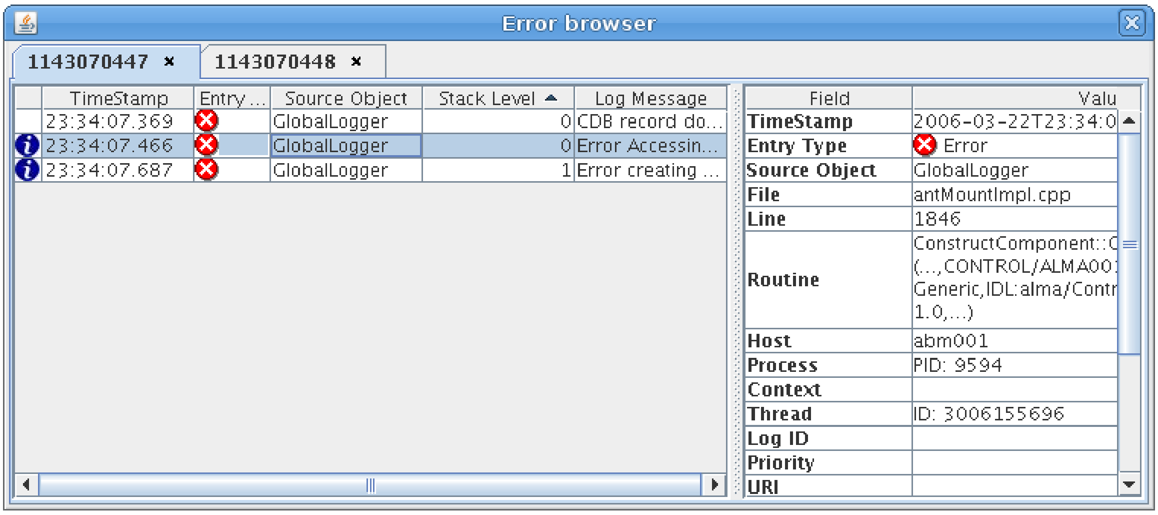
8 Error browser

The error browser dialogs shows an ACS error trace. By reading the ACS documentation about the error system, you can see that all the logs belonging to the same error trace have the same stack ID and a stack level to identify their position in the error stack. At startup the table hides the stack ID and the stack level of the logs but they can be shown by selecting the Fields menu item.

If the selected log has a stack ID defined, the popup menu triggered by pressing the right mouse button over the log entry in the table has the *Show error stack* menu item enabled. By selecting such an item jlog perform the following operations:

- read the stack ID of the log
- shows the error browser dialog
- create a new tab in the dialog having the stack ID as title
- looks for all the logs in memory belonging to the error trace identified by the stack ID
- populate the table of the error stack with all the logs

The following picture shows the error browser dialog with 2 error traces i.e. 2 tabs.



	TimeStamp	Entry ...	Source Object	Stack Level ▲	Log Message
	23:34:07.369	✖	GlobalLogger	0	CDB record do...
ⓘ	23:34:07.466	✖	GlobalLogger	0	Error Accessin...
ⓘ	23:34:07.687	✖	GlobalLogger	1	Error creating ...

Field	Valu
TimeStamp	2006-03-22T23:34:0
Entry Type	✖ Error
Source Object	GlobalLogger
File	antMountImpl.cpp
Line	1846
Routine	ConstructComponent::C (...,CONTROL/ALMA00 Generic,IDL:alma/Contr 1.0,...)
Host	abm001
Process	PID: 9594
Context	
Thread	ID: 3006155696
Log ID	
Priority	
URI	

Each tab is composed of a table populated by all the logs having the same stack ID shown as the title of the tab. By selecting a log of the table, the details table at the right side is populated with the field of the log. To close a tab, press the little widget close to the title.

The table of logs is sortable by pressing the column header, the columns can be moved in the usual way. To select the column displayed by the table you must press the right mouse button over the table header and select the menu item in the popup menu.

The table in this dialog is very close to the table displayed in the main window but it has reduced functionalities. For example it is not possible to add information to a log or to save the selected log. This is done to reduce the amount of resources needed by this fetaure.

The error browser dialog can be closed and displayed again by selecting the View/Error browser menu item. If there are error stacks in the dialog they will be preserved during the closing and reopening.

9 Command line

To launch the logging client use the following command line:

```
jlog [logFileName] [(-f|--filter) filterFileName] [(-d|--discard)
  (NONE|discard level)] [-dnc|--DoNotConnect] [-u||--unlimited]
```

It is possible to specify in the command line a log file as well as a set of filters to load at startup.

If a set of filters is specified, they will be loaded immediately after startup, before connecting to the logging notification channel or loading a file of logs. These filters applies to the table and not to the engine.

If a file of logs is specified then it will be loaded immediately after loading the filters (if any). In this case the logging client does not connect to the CORBA notification channel but works offline. The user can connect to the logging channel using the apposite menu item at any time.

If a filter file name and a log file name are not in the command in the command line, then the client starts with an empty table of logs, an empty set of filters and tries to connect to the logging notification channel for receiving logs.

The -d or –discard switches are present, it is possible to define the initial discard level for jlog. This is very important especially when reading logs from a file: if the discard level is not set with -d then the default, debug, will be used.

The -dnc or –DoNotConnect switch inhibits jlog to connect to ACS at startup.

The -u or –unlimited does not limit the number of logs kept in memory by jlog. It is again very important while loading files from command line. In such a case in fact if the -u switch is missing then the number of logs in memory is limited to 100K as default.

Appendix A: acsLogAssistant

In the real world it happens to have very big files of logs that are very slow or impossible to open with jlog. acsLogAssistant helps manipulating such log files extracting logs or splitting the big file in several shorter files of log.

The command line of the tool is the following:

```
acsLogAssistant [options] command command_params src dest
```

The tool opens a log of files, src, and writes one o more output files, dest, based on the criteria indicated in the command_params.

There are 2 available command: split and extract.

The split command (-split or –p) split the input file in more files having the specified number of logs (-n) or the number of minutes (-m) specified in the command_params. It is possible to select only one between minutes and number of logs.

While splitting, the name of the output files is obtained adding a sequenzial number to the dest parameter. The xml extension is added by the tool.

It the splitting happens by number of minutes, then the date of the first log in each file will appear the name of the file.

For example if you need to split the file all_Logs.xml with a set of files each of which has 50000 logs, you should use the following command line:

```
acsLogAssistant -split -n 50000 all_Logs.xml shortLog
```

The tool produces files named shortLog-0.xml, shortLog-1.xml and so on.

The second way to use the script is to extract logs that match some criteria from a file. The logs are all written in the dest file (again the xml extension is added by the tool). The extraction mode is activated selecting –extract or –x as command.

The criteria for the extraction are all executed in AND. In the command line the –s and –e parameters allows to select the start and end date of the logs you want to select.. The format of the date is ISO without milliseconds (for example 2006-12-05T15:20:45).

The start and end date can be missed and in such a case the tool assumed that the log matches the date so it is possible to extract all the logs generated before or after a defined date passing that date with –e or s.

The following command lines show how to select the logs based on their date:

```
acsLogAssistant -x -s 2006-12-05T10:20:00 -e 2006-12-05T10:30:00
allLogs.xml log
```

A file of filters can be passed to the tool for a more defined extraction. Such file can be created with the logging client. –f tells the tool to use a filter file. The filters are added in AND to the dates while checking for the logs that match the criteria. Filters and dates must be used carefully because it is possible to define time interval also with the filters creating conflicts.

The following example shows how to extract from logs.xml all the logs generated after 200/12/05 at 09:50 and with a given set of filters previously generated with jlog:

```
acsLogAssistant –extract –s 2006-12-05T09:50.00 –f filters.xml
logs.xml myLogs
```

We kindly suggest you to look at the paragraphs for the filters to use this tool as its best.

The tool accepts two options, -csv and –col.

-csv is used to write the out as comma separated values instead of XML. The CSV standard is used quite often and is useful to plot values or read the file as spreadsheet.

-col is used in conjunction with –csv to select the columns and their order in the generated output. For default, the file writes all the columns in the CSV output in their native order.

-col is a string containing chars matching the following regular expression [0-9a-g]*. Each character has the following meaning:

0	Timestamp
1	Entry type (log level)
2	Source object
3	File
4	Line
5	Routine
6	Host
7	Process
8	Context
9	Thread
A	Log ID
B	Priority
C	URI

D	Stack ID
E	Stack level
F	Log message
G	Additional data

The same example shown before write the output in CSV format having only three cols: the message, the date and the time stamp of each log:

```
acsLogAssistant -csv -col f10 -extract -s 2006-12-05T09:50.00 -f
filters.xml logs.xml myLogs
```