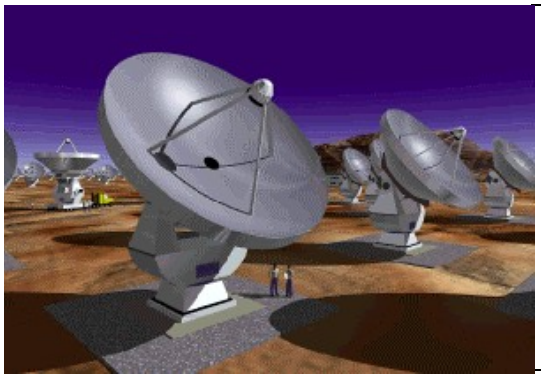


Atacama Large Millimeter

Revision: 1.2

2003-12-08

User's manual

Hernan Raffi

ACS Configuration Database Browser

User's manual

Hernan Raffi

University of Newcastle upon Tyne

Keywords:

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

Change Record

REVISION	DATE	AUTHOR	SECTIONS/PAGES AFFECTED
	REMARKS		
1.0	2003-09-14	Hernan Raffi	All
	Created		
1.1	2003-12-04	Hernan Raffi	All
	Updated according to comments by Gianluca Chiozzi, ESO		
1.2	2003-12-08	G.Chiozzi	All
	Minor changes		
1.3	2004-05-19	A. Caproni	
	Added the Edit section		

Table of Contents

1 Introduction

The Configuration Database (CDB) Browser is a tool used to visualize the tree like structure of any CDB instance and look at its XML configuration files used to maintain and initialize the Components of an ACS system.

The CDB Browser enables developers to edit the attribute values of any Component and switch between different views of the same data. More precisely the Browser shows an ‘XML view’ of the configuration data exactly as it is stored inside the CDB and a ‘Table view’ of the same data, presented in an organized table.

In the current implementation, the ACS CDB support a writable interface and edited values are written in the actual CDB.

2 Application Overview

The ACS CDB is accessed through an IDL interface that provides an abstraction for a hierarchical configuration database and decouples the access to configuration data from the physical storage mechanism.

Configuration data appears logically as XML files stored in a hierarchical directory structure. There exists a one-to-one mapping between the logical path of the XML configuration file, the name of the entity, and the tree path where the data is represented in the CDB tree.
For example, configuration data for the object:

/ALMA/Antenna1/Motor3

can be found in the CDB tree under the logical path:

root->alma->Antenna1->Motor3

Actual implementation of the service can store physically the data in different ways.
ACS provides a default implementation called jDAL based on a simple file-based XML database implemented in Java. This implementation stores XML files on the normal file system. In this way the XML configuration file for the entity described before is located in the file:

$SOMEWHERE/ALMA/Antenna1/Motor3/Motor3.xml

Other implementations used by ALMA are based on a public domain XML database (Xindice) and on a relational database (DB2) but are accessed via the same DAL IDL interface.

At start-up the configuration data will be displayed by the cdbBrowser in a tree like structure. By selecting a node the request goes to the jDAL server to see if the requested XML entry exists. Should the request be successful a XML string will be returned and a tabbed pane will appear on the right side of the window. This will provide two alternative views of the same data, namely the *Table View* and the *XML View*.

The GUI also shows a message window at the bottom that displays text messages depending on the user's action.
At the top of the GUI there is a location bar that shows the current selected path in the CDB tree.

There are also warning dialogs, which pops up every time the user makes an inaccurate action (e.g. user forgets to save changes to a table).

3 Main Window

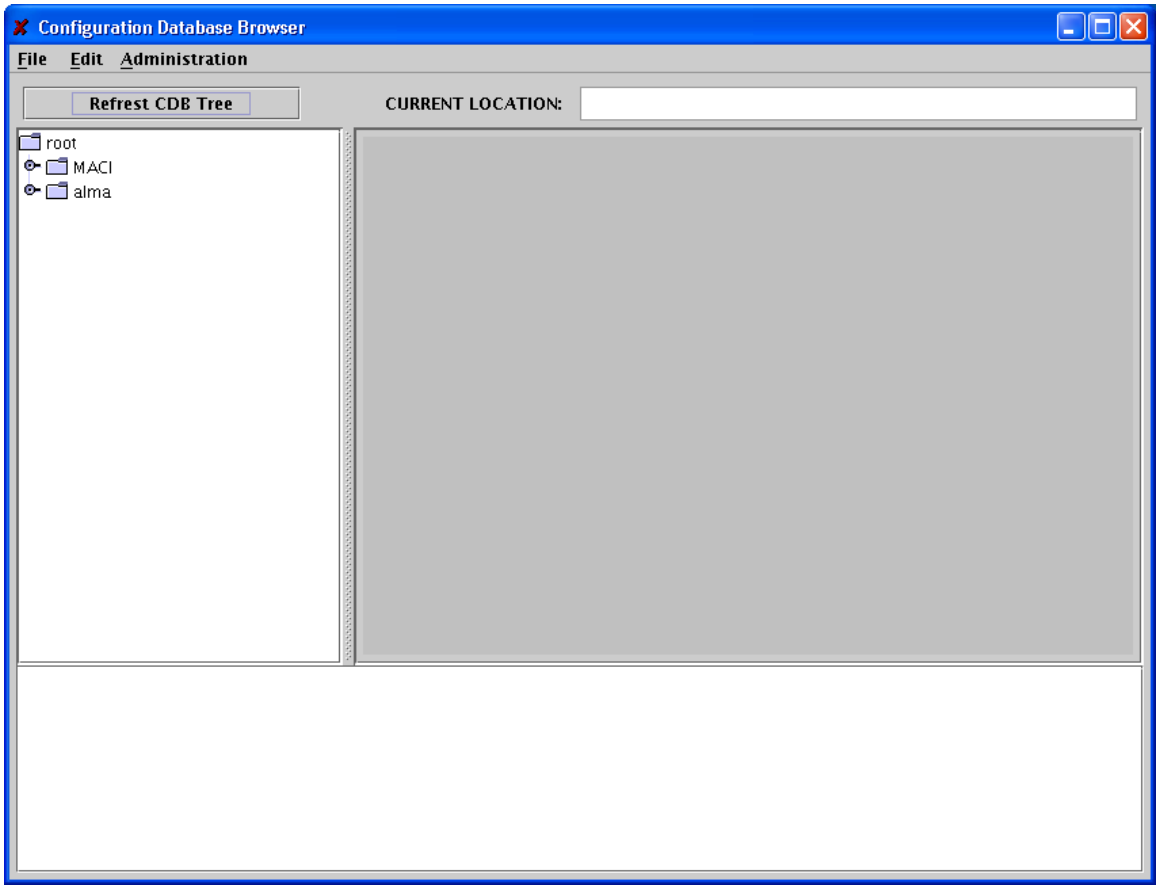


Figure 1 The Configuration Database Browser.

The CDB Browser Window is divided into three sections: the CDB tree (left side), the tabbed pane (right side) and the message text area (bottom), with an additional button (*Refresh CDB Tree*) and display field on top (*location bar*).

4 The GUI

4.1 Menus

4.1.1 File

In the *File* menu you can choose the following:

- *Exit:* The *Exit* option will close the CDB Browser window and exit the program, but only if there are no changes left unsaved. If this is the case a pop up dialog will appear preventing the user to close the Browser (see Figure 7).

- *Save XML file*: The *Save XML file* menu item will open a File Chooser (see Figure below) to navigate the local file system and save the XML document.

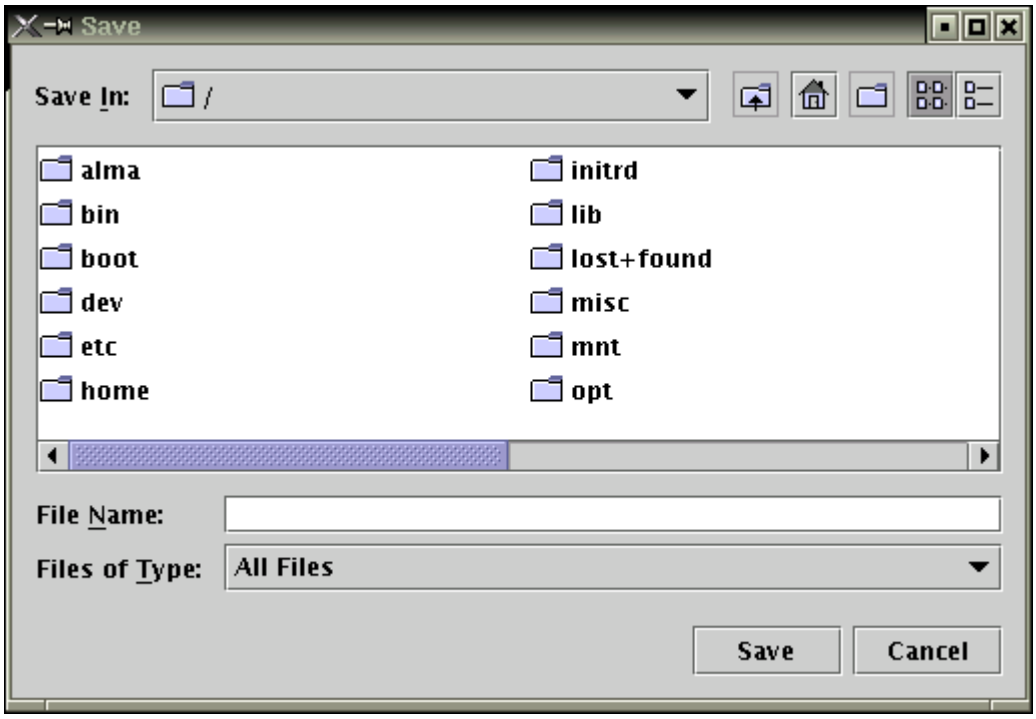
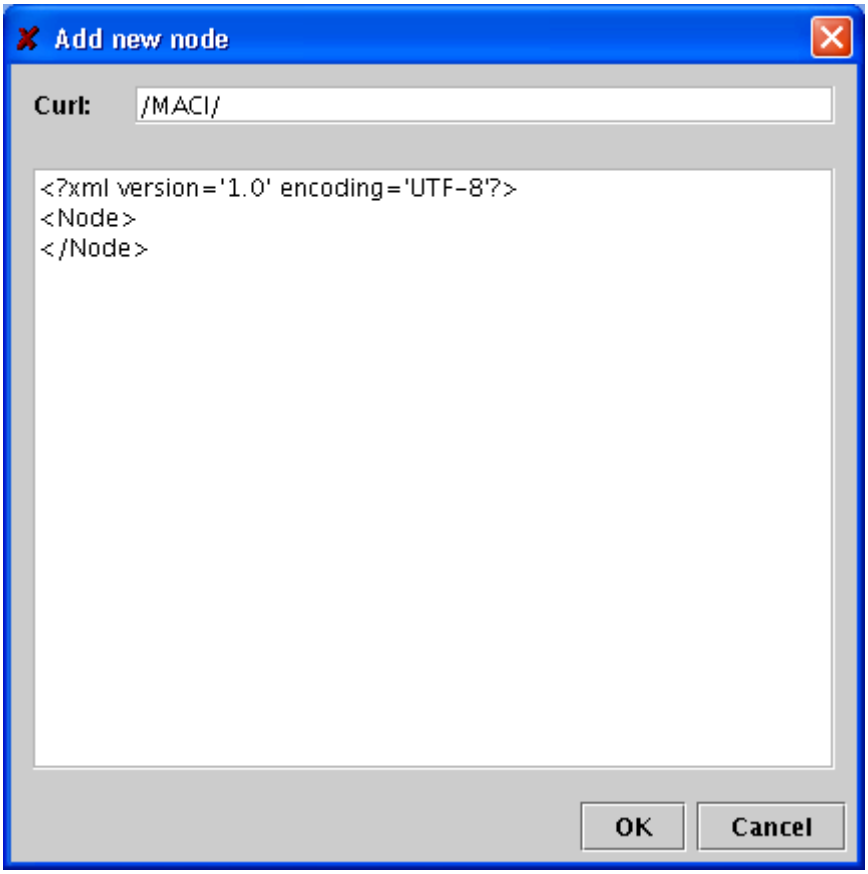


Figure 2 The File Chooser.

The File Chooser will only appear if the *XML View* pane is visible in the right side of the window. If this is not the case a pop up dialog will appear telling the user to select a XML tab (see Figure 6).
If the user edits an XML string and tries to save it before validate it a warning pop up dialog will warn the user that he might save invalid data (see Figure 8).

4.1.2 Edit

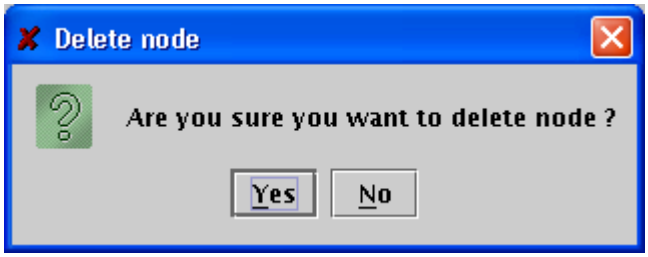
The edit menu allows the user to add and delete new nodes[1] into the CDB. By pressing the Add node menu item the following dialog is shown.



To add a new node the path of the XML file describing the node has to be inserted in the Curl text field.

In the big text field, the full xml file must be written.

The Remove node removes the selected node but after it requests a confirmation showing the following dialog:



[1] You can find a short explanation about what a node is in another section of this document.

4.1.3 Administration

The *Administration* menu provides the *Clear Cache* menu item to clear the CDB cache. This function is specific for the jDAL implementation, that caches in memory data read from the XML files. It will not be selectable for other implementations.

4.2 The CDB Tree

4.2.1 The Refresh CDB Tree button

This button is located on top of the CDB Tree window. By clicking it a new session will start; the CDB tree will be reinitialized by resetting the connection with the DAL server. The message text area and the area containing the tabbed pane will be cleared.

If the user tries to refresh the tree without having saved or reset changes made to a node a warning pop up dialog will appear (see Figure 7) preventing the refresh action to occur.

4.2.2 Nodes

A node in the CDB tree can represent an entity whose configuration is store in the CDB (like a Component, a Manager or a Container); there exists a one-to-one mapping between the name of the entity and the path of the CDB tree that brings to that node. For example the Component alma/LAMP1 is represented by a tree node named LAMP1 whose path is root->alma->LAMP1.

When a node is expanded for the first time and the node is an instance of the 'CDBTreeNode' class, than a request is sent to the DAL server to check if the requested XML entry exists.
If the record does not exist a '*RecordDoesNotExist*' exception is raised.
If the record is found the DAL provides an interface to get access to the corresponding DAO, which returns the requested XML string and a linked hash map containing the attribute names and values of the selected node.

4.3 The Tabbed Pane

The Tabbed pane is located on the right side of the Browsers GUI, it appears every time the user extends a node for the first time, and the jDAL server successfully returns its XML record or if the user re-selects a same node.
The Tabbed pane has two tabs: the *Table View* tab and the *XML View* tab, both standing for the same XML record. The user chooses which component to view by selecting the tab corresponding to the desired component.

4.3.1 The Table View pane

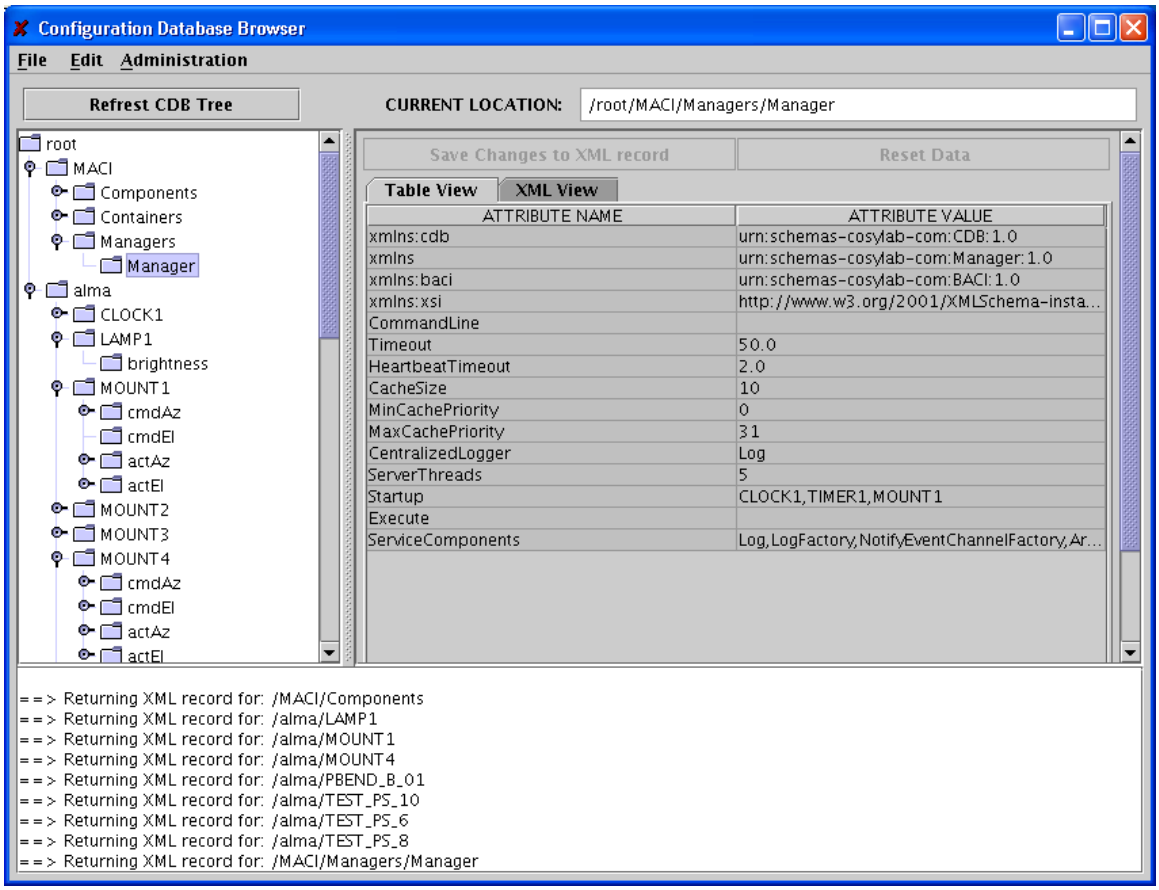


Figure 3 The Table View.

The *Table View* tab shows the Attribute names and values of the selected node.
Only the column containing the attribute values is editable. As soon as the user starts editing a cell the two buttons on top of the tabbed pane (*Save Changes to XML record* and *Reset Data*) get enabled.

The user can edit as many values as required and save them by clicking the *Save Changes to XML record* button. This will copy the changes to the corresponding XML string (see Figure 4) and sent it for validation to the jDAL server.
The user can also decide to reset the table to its original values (since the last time they where saved) by clicking the *Reset Data* button.
During editing of the table the CDB tree gets disabled so that the user has no choice than saving or resetting the changes made.

4.3.2 The *XML View* tab

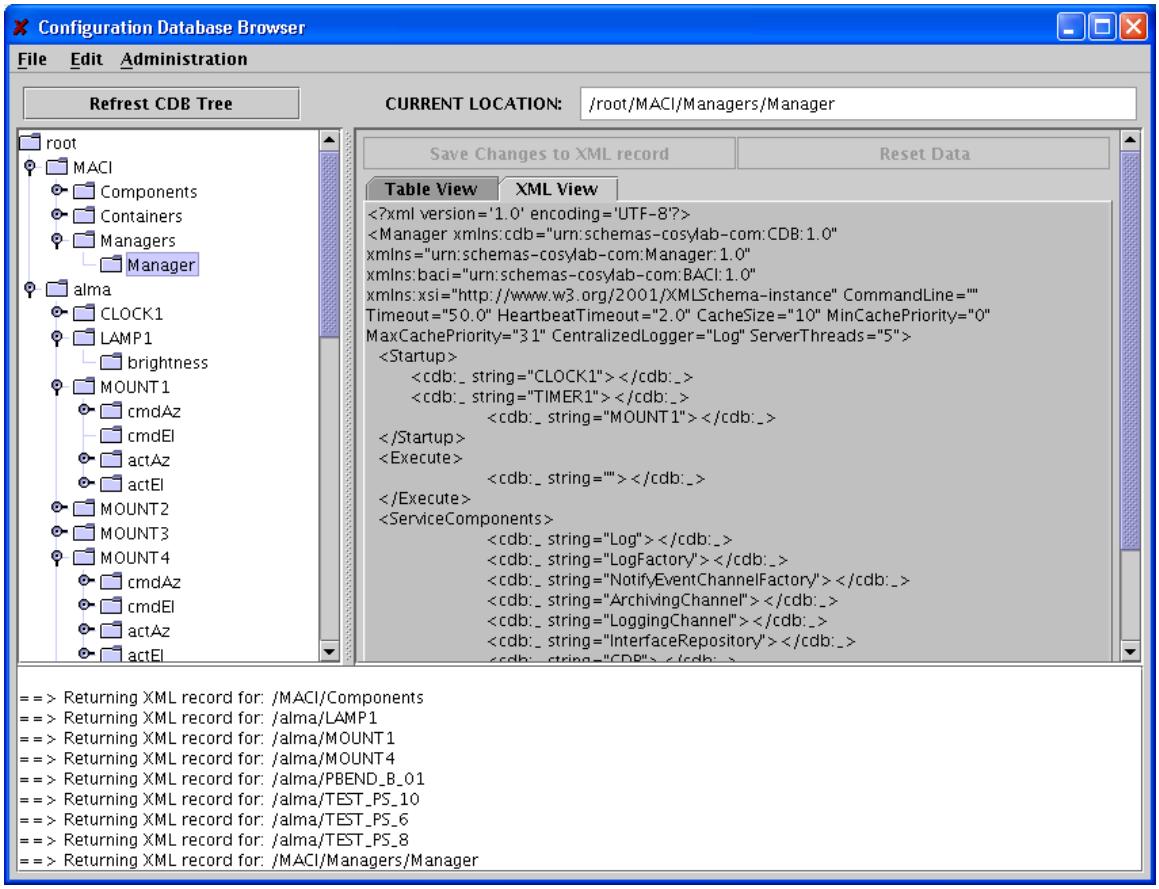


Figure 4 The XML View.

The *XML View* tab shows the XML string of the selected node. If a node has no XML record the XML tab will be disabled. The XML string is editable; during editing the CDB tree get disabled and the two buttons on top of the tabbed pane get enabled.
If the user decides to save the changes, the XML string will be send for validation to the DAL server. If validation succeeds the CDB tree will be refreshed.
By clicking the *Reset Data* button the XML string will be reset to its original value.
The user can save an XML string in a user defined directory by selecting the *Save XML View* option in the *File* menu.

4.4 Message Text Area

The message text area displays messages and shows the history of the users actions. Error messages also appear in the message text area.

5 The Dialogs

The program uses dialogs to warn the user that he is trying to do some action that is not allowed in that moment.

Note: *All dialogs are modal, so that no other action on the Browser is possible before the user responds to the dialog.*

Figure 5 Warning Dialog: Refresh CDB Tree.

Once the user starts editing a table he is not allowed to do any other action than saving the changes, reset the table values or keep editing. If the user tries for example to refresh the CDB tree, a dialog as shown above appears.

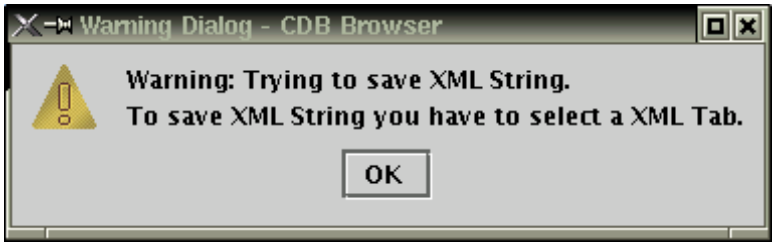


Figure 6 Warning Dialog: Save XML String.

This dialog appears when the user selects the *File->Save XML String* menu item without first selecting an XML tab.

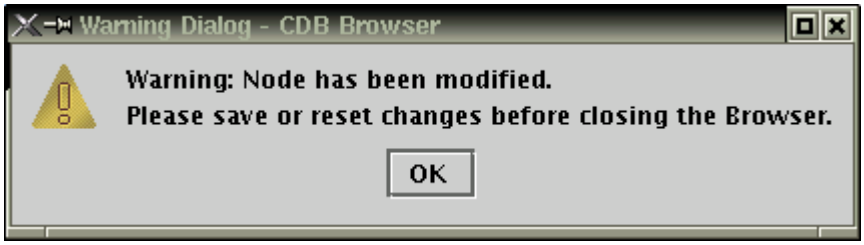


Figure 7 Warning Dialog: Closing Browser.

This dialog appears when the user edits the attributes of a specific node and tries to close (*File->Exit*) the Browser.

Figure 8 Warning Dialog: Save XML string.

This dialog appears when the user tries to save a XML String in its local file system without having saved the changes made.

6 The CDB Browser

6.1 Starting the CDB Browser

In order to run the CDB Browser the services provided by the Database Access Layer (DAL) must be activated. The command for starting the jDAL default implementation coming with ACS is:

```
$ cdbjDAL
```

See the corresponding ACS documentation for details.
After DAL activation (terminated with message: JDAL is ready and waiting…) the command to start the CDB Browser can be given (normally in a different window):

```
$ cdbBrowser
```