System User Manual

Open Simulation Framework

Parameter Editor

Javier Martin Ávila – Technical Responsible
Mercedes Pavía – Project Manager
Document Code: OPENSF-DMS-PE-SUM
Version: 1.14
Date: 27/11/2025
Confidentiality Level: Unclassified

Indra Deimos

indracompany.com

This page intentionally left blank

Document Status Log

Issue	Section	Change Description	Date
1.0	All	First issue of this document. Content extracted from [RD-3] and updated with HMI revamping for Eclipse RCP.	15/12/2017
1.1	All	Second issue of the document, released with the PE	15/06/2018
1.2	3 and 4	Removed Rules Files section and added one on XSD Replaced inconsistent images	14/12/2018
1.3	4.4.3 4.4.4.1	Added section on parameter groups edition Added information on the inline parameter edition	17/07/2019
1.4	1, 2, 3.2 4.2, 4.4 4.4.4.1	Updated references, acronyms and Eclipse platform requirements Updated description of the menus and the Insert Parameter dialog Added description of the inline-editable parameter value representation.	12/03/2020
1.5	4.1 4.4.2 4.4.4.1 4.6 4.7	New main window snapshot New parameter names and groups restrictions, merged with 4.4.3 Added Inline Edition condition New log panel described; snapshots substituted Added dimensions trimming to the validation process	03/07/2020
1.6	4.4.1 4.4.3 4.4.3.1	Added description about acceptance of empty string elements Clarification about empty string elements on array parameters Updated explanation about string parsing	26/10/2021
1.7	3.2.2 4.2 4.4.4 4.4.5 4.6	Updated target environments Modified the descriptions of the commands in the edit menu Removal of multiple parameters no longer requires individual confirmation New section about copy/paste of parameters Description of the context menu in the log panel	12/05/2022
1.8	All 4.2, 4.3	Updated screenshots of the tool Merged into new Sec. 4.1.1, updated menu and toolbar descriptions	15/12/2022
1.9	4.2	Updated description of find functionality and parameter validity	23/05/2023
1.10	4.4	Add mention of the new warning for empty groups.	01/12/2023
1.11	3.2.2 4.2.3 4.2.5	Updated target environments after base Eclipse RCP upgrade. Rewrite for the new tree-based parameter values edition UI. Explain the new functionality to paste parameters into a target group.	27/05/2024
1.12	3.2.2	Added support for Apple Silicon Mac	19/11/2024
1.13	(all)	Port to new template and styles	28/05/2025
1.14	(all) 3.2.2 4.2	Refreshed screenshots Updated target environments Add section explaining the parameters consolidation and sorting functionality, which was introduced earlier but was not described in the manual.	27/11/2025

Table of Contents

List of Figures

List of Tables

1. INTRODUCTION

During the concept and feasibility studies for the ESA Earth Observation activities, the mission performance up to the final data products needs to be predicted by means of end-to-end (E2E) simulators; later on this becomes a coherent test bed for L1PP and L2PP and to support the verification of space segment performance and associated sensitivity analysis.

A mission E2E simulator is able to reproduce all significant processes, design and steps that impact the mission performance as well as output simulated data products.

OpenSF is a software framework to support standardised end-to-end simulation capabilities allowing the assessment of the science and engineering goals with respect to the mission requirements. Scientific models and product exploitation tools can be plugged in the system platform with ease using a well-defined integration process

The openSF ParameterEditor (PE) is a software application that adds extra functionalities to the simulation framework. This software has been developed in the framework of the Sentinel 3 Optical System Performance Simulator contract (Thales Alenia Space France), it is an on-going activity so the application could be subject to updates, bug correction and minor changes in short term. This software is included in openSF default installation since version 2.0.

1.1. Purpose

This document has been produced by DEIMOS within the frame of the openSF project and represents the Software User Manual for the openSF ParameterEditor.

The objective of this document is to provide a clear description of the functionalities of the ParameterEditor.

The intended readerships for this document are openSF users.

1.2. Scope

This document applies to PE v2.9 and openSF v4.5 and contains:

- Section 1 talks about the document, giving a description and settling the basis to understand it.
- Section 2 links this document with information from other sources.
- Section 3 explains briefly the functionalities of ParameterEditor and how to start it.
- Section 4 describes one by one all the different functionalities of the ParameterEditor.

Reading the chapters in this order will help users to fully understand the use of the system

1.3. Acronyms and Abbreviations

The acronyms and abbreviations used in this document are the following ones:

- **AD:** Applicable Document
- **API:** Application Programming Interface
- **E2E:** End to end simulation
- **GUI:** Graphical User Interface
- **HMI:** Human-Machine Interface
- **ICD:** Interface Control Document
- **RD:** Reference Document
- **SUM:** System User Manual

1.4. Definitions

The definitions of the specific terms used in this document are the following ones:

- **Framework**: Software infrastructures designed to support and control the simulation definition and execution. It includes the GUI, domain and database capabilities that enable to perform all the functionality of the simulator.
- **Configuration File**: A XML file that contains parameters necessary to execute a module. A configuration file instance must comply with the corresponding XML schema defined at module creation time. A special case is the global configuration file that defines the configuration parameters that are common to all modules.
- **Parameter:** A constant whose value characterizes a given particularity of a module. Parameters are user-configurable, they are fixed before launching a module and, for practical reasons, and not all of them shall be accessible from the HMI.

2. RELATED DOCUMENTS

2.1. Applicable Documents

The following table specifies the applicable documents that shall be complied with during project development.

Table 1: Applicable documents

Reference	Code	Title	Issue
[AD 1]	openSF-DMS-ICD-001	openSF Interface Control Document	3.0.1
[AD 2]	openSF-DMS-ADD-001	openSF Architecture Design Document	2.2
[AD-3]	PE-ID-ESA-GS-464	ESA generic E2E simulator Interface Control Document	1.4.2

2.2. Reference Documents

The following table specifies the reference documents that shall be taken into account during project development.

Table 2: Reference documents

Reference	Code	Title	Issue
[RD 1]	OSFI-DMS-TEC-DM	openSF Integration Libraries Developers Manual	1.24
[RD 2]	OSFEG-DMS-TEC-DM	openSF Error Generation Libraries Developers Manual	1.8
[RD-3]	OPENSF-DMS-SUM01	openSF System User's Manual	4.8
[RD-4]	RCP-target-envs-def	Eclipse Target Environments	4.31

2.3. Standards

The following table specifies the standards that shall be complied with during project development.

Table 3: Standards

Reference	Code	Title	Issue
[STD 1]	ECSS-E-40C	Software development Standard	-
[STD 2]	www.w3.org/TR/xml11	Extensible Markup Language (XML) 1.1	2nd ed.
[STD-3]	www.w3.org/TR/REC-xml-names	Namespaces in XML 1.0	3rd ed.

3. GETTING STARTED

3.1. Introduction

The ParameterEditor is not only a graphical front-end for creating and editing the parameters involved in a simulation but also an interface to introduce constraints between parameters defined in different files.

The new functionalities provided by this tool are the following:

- User-friendly parameters interface allowing the creation, edition and deletion of them avoiding the XML text editing.
- Enhanced consistency checking of parameters. It includes range, type and dimension check.
- Enhanced editing with excel-like interface for vectors/matrix parameters and plot capabilities.
- Interface for rules and constraint definition connecting parameters that can be located in different configuration files.

3.2. System Requirements

3.2.1. Hardware requirements

No specific hardware requirements over those defined in [RD-4].

3.2.2. Operating system requirements

Not all the platforms targeted by the Eclipse platform are officially supported by openSF. Binary distributions are currently provided for the following platforms:

- Linux x86-64: in particular, PE has been tested with Ubuntu 24.04.2. Older versions may require an update of the Gtk3 libraries as indicated in the Eclipse requirements.
- macOS: version 13 or higher, either Intel or Apple Silicon.
- Windows 10 or 11, x86-64

3.3. How to Start the Application

The ParameterEditor can be launched from openSF clicking in the icon shown in the screenshot below:

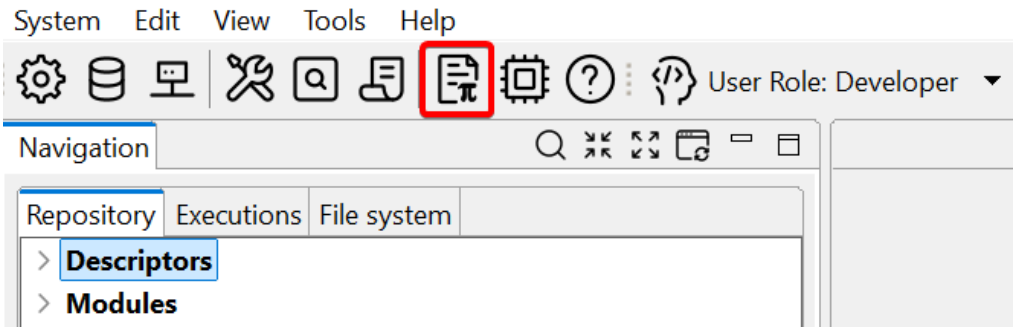

Figure 3-1: ParameterEditor Shortcut

In addition, it can be launched opening a terminal, going to openSF home folder and launching the start script inside the ParameterEditor subfolder, or just double-clicking in the executable.

4. PARAMETER EDITOR

4.1. Main Frame

Here is presented the look-and-feel, operational behaviour and design features of the ParameterEditor application. The HMI accepts input via devices such as the computer keyboard and mouse and provides articulated graphical output on the display. Thus, certain aspects of the HMI implement also the **Object Oriented User Interface (OOUI)** paradigm because it is built from different pieces, or objects with several properties and operations.

The main window is based in three split panels allowing users to resize the component they want to focus on. The three graphic components of the ParameterEditor main frame are:

- Parameter File View: Tabbed panel showing the parameters stored in a set of files. An asterisk will appear before the parameters file name anytime a change is performed but it is not saved into the correspondent XML file.
- Log Console view: This panel shows information about the operations performed by the application data layer, exceptions, work flows, etc.
- Side panel containing the function buttons and showing the status of the application, files already opened, etc.

Figure 4-1 shows the appearance of ParameterEditor's main window.

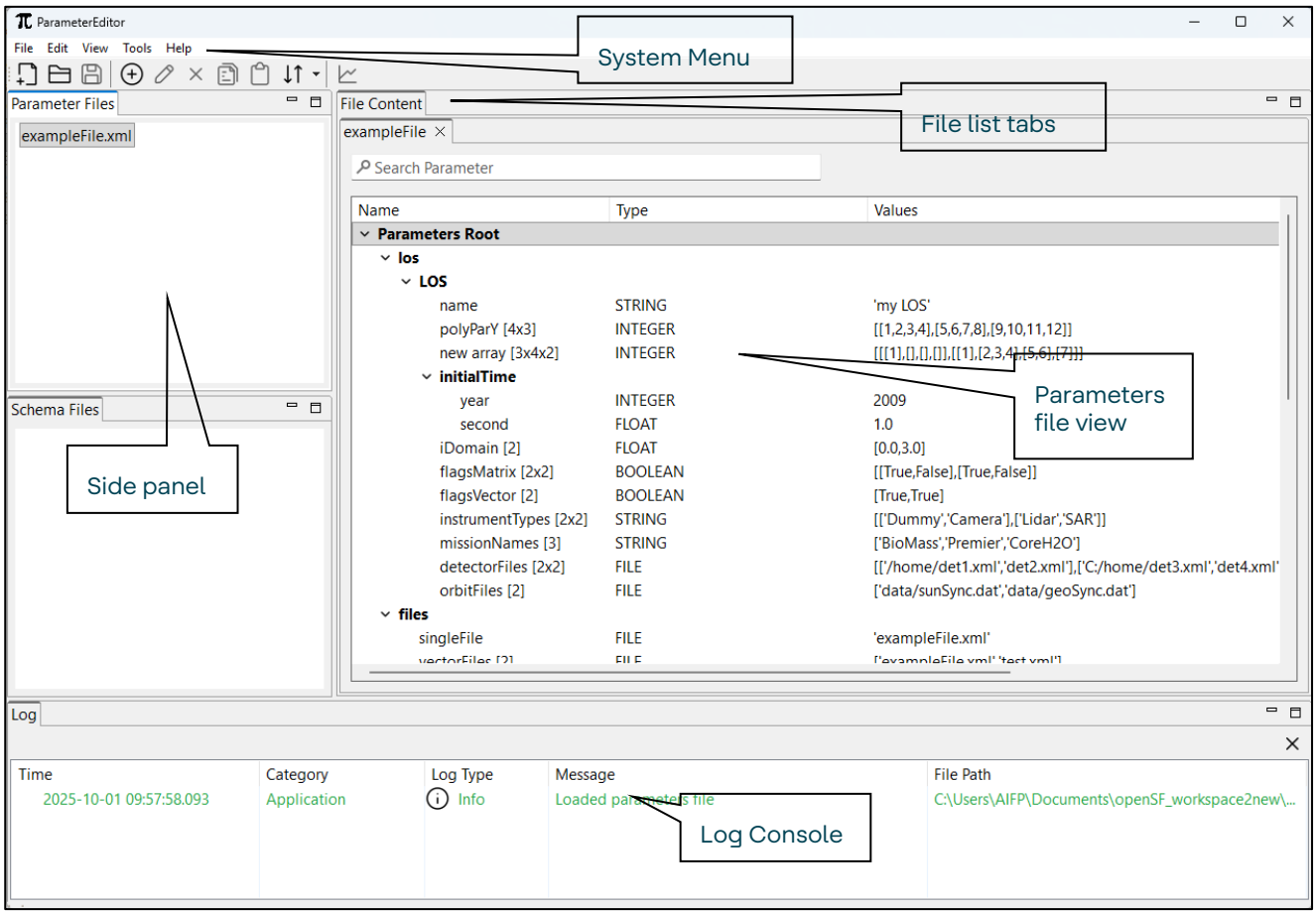

Figure 4-1: ParameterEditor Main Window Appearance

4.1.1. Main Menu and Toolbar

The ParameterEditor menu provides the following actions, grouped by menu item:

File Menu

This menu gives access to ParameterEditor main functionalities such as:

- Open File: load a new parameters file
- Save: save current parameters file
- Save All: save all parameters files
- Open Schema File: load a schema file
- Exit: close the application

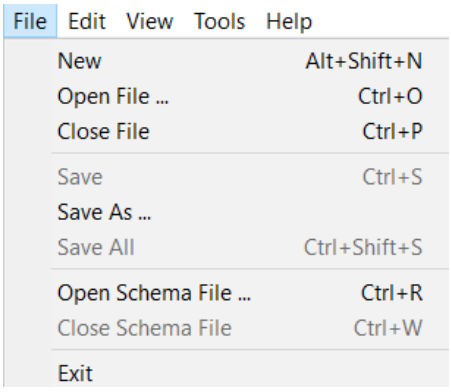

Figure 4-2: ParameterEditor File Menu

Edit Menu

- Copy Parameter: see Section 4.2.5.
- Paste Parameter: see Section 4.2.5.
- Delete Parameter: Delete the selected parameters.
- Edit Parameter: Edit the selected parameters. It opens a new window, as one can see in Section4.2.3.
- Insert Parameter: It inserts a new parameter after the index of the parameter selected by the user. It opens a new window, as one can see in Section 4.2.1.
- Plot Parameter: Plots a given layer of the selected parameter if that parameter is of type INTEGER or FLOAT (see Section 4.2.6).

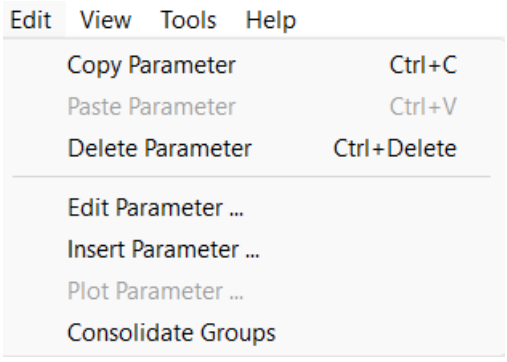

Figure 4-3: ParameterEditor Edit Menu

View Menu

- Clear Log: Clears the Log Console - see Section 4.4.
- Reset Views: If the user wants to reset the perspective of the several panels to the default one.

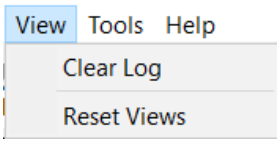

Figure 4-4: ParameterEditor View Menu

Tools Menu

- XML Text Editor: Opens the selected configuration file in the default OS tool to open .xml files.
- Plot: Plots a given layer of the selected parameter if that parameter is of type INTEGER or FLOAT (see Section 4.2.6).
- Validate (see Section 4.5).
- Apply Schema File: Applies the selected Schema File to the selected Parameters File (see Section 4.6).

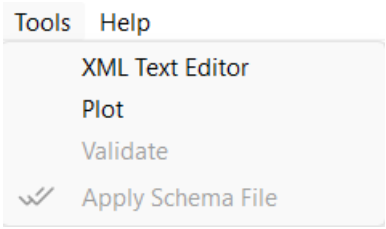

Figure 4-5: ParameterEditor Tools Menu

Help Menu

- About ParameterEditor
- E2E-ICD: ESA generic E2E simulator Interface Control Document
- ParameterEditor SUM: opens this document in the default PDF viewer selected in the system.
- Note: Keep in mind that the “Help Menu” looks different in the MacOS since MacOS automatically inserts “About ParameterEditor” under the application name’s category.

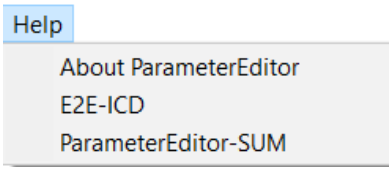

Figure 4-6: ParameterEditor Help Menu

Shortcuts Toolbar

From left to right the meaning/function of each shortcut is explained hereafter. Most of them have already been described previously in Section 4.1.1, and thus they are only mentioned and not explained in detail.

- First category – Configuration file shortcuts
  - New (Ctrl+ N in Linux/Windows; ⌘ + N in macOS)
  - Open File (Ctrl/⌘ + O)
  - Save File (Ctrl/⌘ + S)
- Second category – Individual Parameter shortcuts
  - Insert Parameter
  - Edit Parameter (Ctrl/⌘ + ENTER)
  - Delete Parameter (Ctrl/⌘ + DELETE)
  - Copy Parameter (Ctrl/⌘ + C)
  - Paste Parameter (Ctrl/⌘ + V)

- Consolidate groups / sort parameters

- Plot Parameter

Figure 4-7: Top toolbar giving shortcuts to several most used functions of the ParameterEditor.

4.2. Parameter Management

This section details the use of the "Parameter File View". The functions available from this panel are:

- Navigate through parameters contained in a parameters file
- Find a parameter in a configuration file (shortcut Ctrl/⌘ +F)
- Double click on a parameter, opening it up for edition – see Section 4.2.3.
- Act on one or more parameters through the context menu

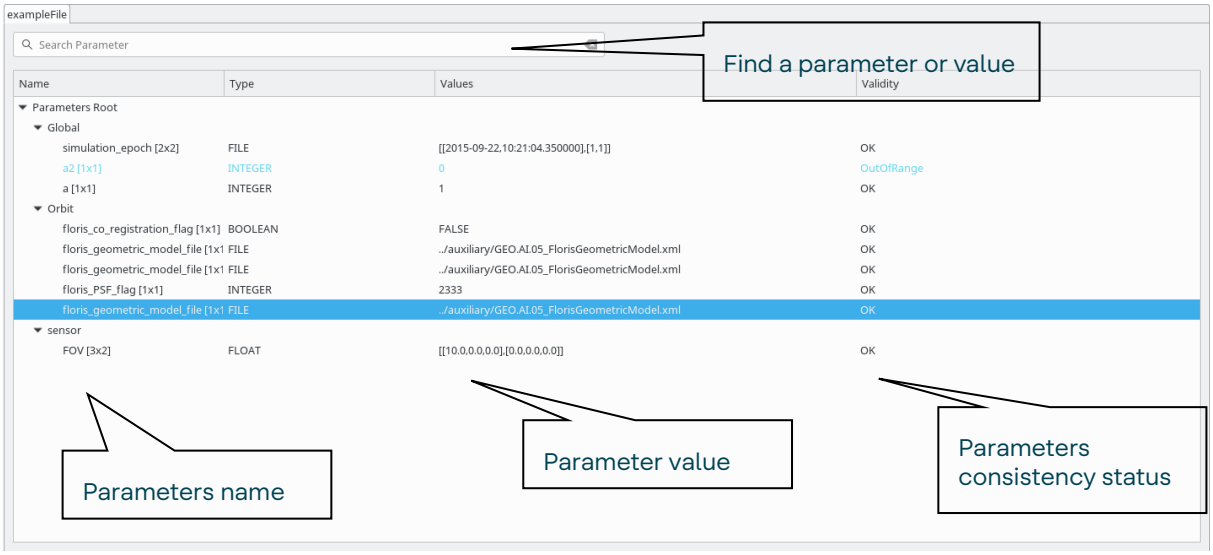

Figure 4-8: Parameters file, Parameters Tree View

Figure 4-8 shows the parameter file view and the functionalities it provides access to:

- Find: assisted finding of a parameter name or a value. Matching parameters are highlighted. If the user hits ENTER while in the search box, the view cycles between all matches. If there are none, or if the view cycles back to the first match, a beep signals this to the user.
- Tree showing parameters names and values, grouped according to the location in the file.
- Validity: for each parameter, the result of the consistency checks
  - Valid: all consistency tests passed.
  - Invalid value format: bad syntax of the value for the type specified in the parameter e.g. the value "7q2@" in an INTEGER parameter.
  - Dimensions inconsistent with value: when the number of values within a parameter does not match with the specified dimensions.
  - Disallowed missing values: MATRIX must have all entries defined and ARRAYS can only have empties in the end of rows.
  - Value outside allowed range: parameter values are not within minimum and maximum specified range. Only available for FLOAT and INTEGER parameters.

- Non-existing file/folder: a FILE/FOLDER parameter points to a path that does not exist.

4.2.1. Insert Parameter

From the Parameter File View shown in Figure 4-8 or from clicking in the respective toolbar icon (Figure 4-7) users can access the "Insert parameter" window. Within this view users can create a new parameter in the configuration file that is under editing. The upper half of the dialog configures the parameter (see below for details about each element), while the values are inserted in the lower half with a spreadsheet-like table interface.

Note that the allowable values depend on the type and structured type of the parameter. For example, INTEGER parameters cannot store characters, and non-ARRAY parameters cannot have missing elements (created when a cell is left empty and represented by a greyed-out NE). On the other hand, the empty string is considered a valid value. This means that a string parameter can contain empty elements. If it is an ARRAY type parameter, the empty elements that are placed at the end of a row will be trimmed (treated similarly to missing elements), but the empty elements placed at the first or intermediate positions will be left as the empty string. Attempting to write an invalid value may be prevented by the editor, or raise an error dialog when the parameter is saved.

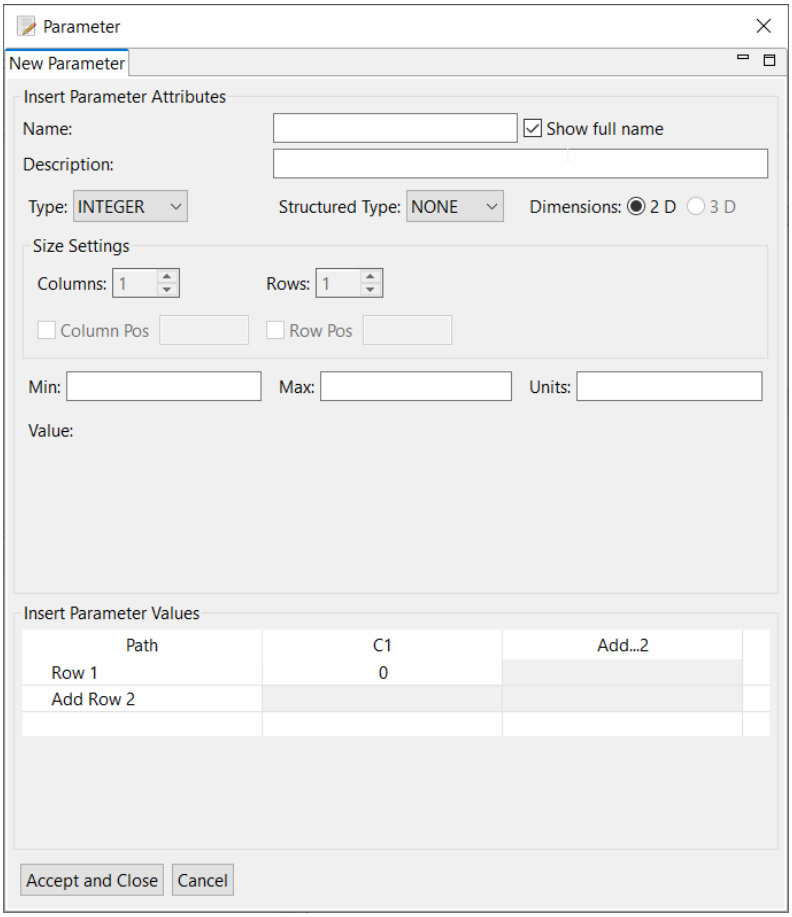

Figure 4-9: Insert Parameter Window

The attributes that can be configured for a parameter are the following:

- Name: parameter name, in full or just the short name. See section 4.2.2 for details.
- Description: optional, free text description of the parameter functionality
- Type: list of the available parameter types, as defined by [AD-3].
- Structured Type: list of the available structured parameter types, as defined by [AD-3].
- Dimensions: the default dimensions are 2D, but if the ARRAY structured type is selected, the user can choose to have a 3D parameter, adding new layers as the third dimension.
- Size Settings (when any field of this area is changed the matrix view is automatically updated):

- Columns and Rows are integer fields specifying the number of each parameter dimension (when a 3D parameter is chosen, there is also a Layers field – see Figure 4-11 (b)).
- Column Pos and Row Pos are integer fields specifying the parameter dimension where the new columns/rows shall be inserted (when a 3D parameter is chosen, there is also a Layer Pos field – see Figure 4-11 (b)).
- Min/Max: optional maximum and minimum values allowed for this parameter. Only available for FLOAT, INTEGER and TIME types.
- Value: string representation of the parameter values, in the same format used in the main table.
- Matrix View: table allowing the modification of a value in a determined matrix position. Row numbers are shown in vertical and column in horizontal. When a value is edited the field "Value" of the window automatically reflects the changes.

The "Accept and Close" button creates the parameter in the currently selected configuration document after basic integrity verifications, while "Cancel" rejects the changes and closes the window without saving the new parameter. Note that the XML file on disk is not modified until explicitly saved.

4.2.2. Parameter Names

A *full parameter name* is composed by a hierarchical set of zero or more group names, and a parameter name, concatenated by dots e.g. "*sensor.band.wavelength*", where "*sensor*" and "*band*" are group names and "*wavelength*" is the parameter name. Each name must be unique within a configuration file, and reflects the structure of the XML grammar that is prescribed in [AD-3]. For example, the mentioned parameter "*sensor.band.wavelength*" has the following XML representation:

```
<sensor>
    <band>
        <parameter name="wavelength" ….>
    </band>
</sensor>
```

When creating a parameter, if the user enters the full name with dot separators in the Parameter Edition window of Figure 4-10, ParameterEditor automatically creates the respective parent nodes corresponding to the group names, as specified. The checkbox "Show full name" next to the name text box is used to alter between showing the full name of the parameter or just the short name.

Specific restrictions are applied to the name of parameters and groups. These are derived from the rules in section 2.3 of [STD 2]. **Each** group or parameter name is defined as a sequence of characters that:

- starts with a letter or an underscore, followed by 0 or more characters – some characters are invalid[1] and cannot be used in names, such as "&", ">" or "<"
- contains at most one colon[2], but cannot start or end with one
- (groups only) cannot be named "parameter"

[1] For the exact definition of the character ranges that may be used in a name, see [STD 2]

[2] The colon rule enables the use by module developers of XML namespaces in configuration files. Namespaces, described in [STD-3][STD-3] allow data of different "domains" to be mixed in a single document. A typical example is the "xsi:schemaLocation" attribute used to mark a XML file for validation with a certain XSD schema.

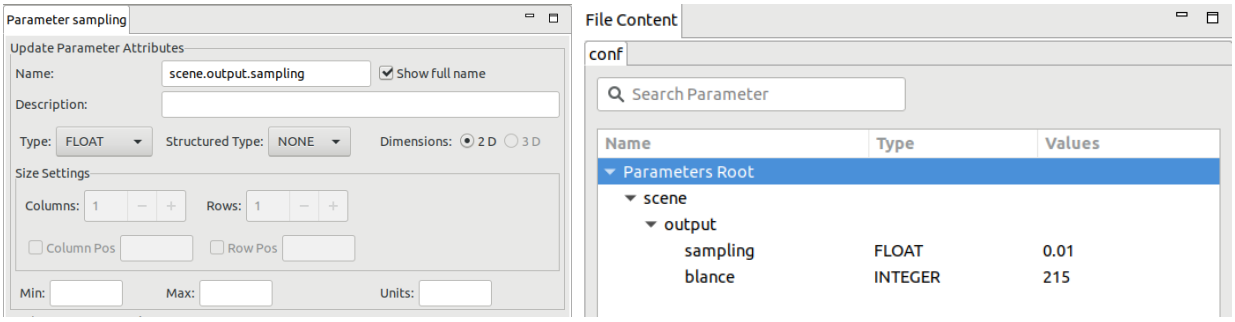

Figure 4-10: An example of a parameter and groups, in the parameter edition (left) and file (right) views

4.2.3. Edit Parameter

The window for edition of an existing parameter is the same as the one used for the creation of a new one, see section 4.2.1. Therefore, the rules to correctly edit a parameter are the same as the ones explained on this section. However, empty string elements at the row ends of array parameters will be removed, if edited on the edition window, or left as empty, if "inline" edited. The window for edition will appear when the user double-clicks on a parameter node in the Configuration File Panel or when he presses the respective toolbar icon/ Shortcut.

Figure 4-11 (a) shows the edition window for a FLOAT MATRIX parameter while Figure 4-11 (b) depicts the edition for a parameter of type INTEGER ARRAY with 3 dimensions.

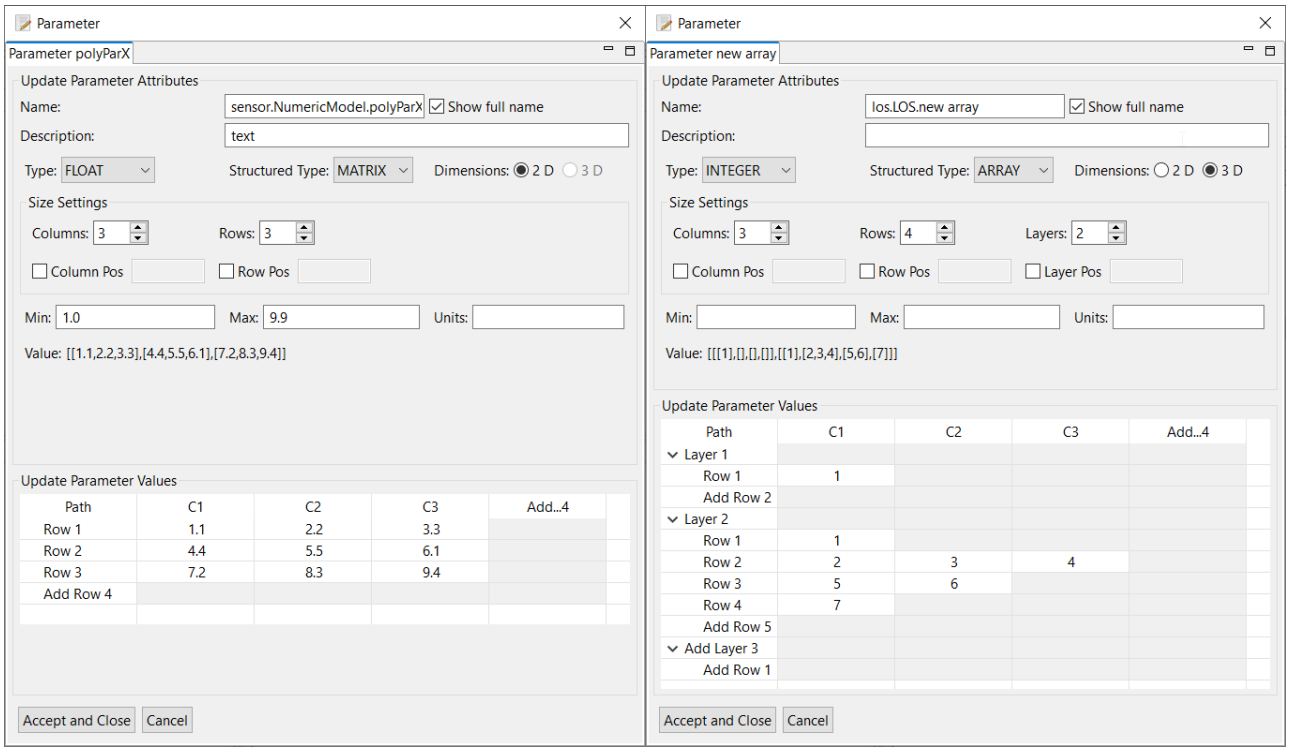

Figure 4-11: Edition of (a) 2D Parameter; (b) 3D Parameter

The following interactions with table are possible:

- **Click to edit an existing value.** In the context of an ARRAY parameter, if a cell is intentionally left empty and it represents the final column of the array, the corresponding row reduces by one column. This action effectively shortens the array by eliminating the trailing column. Also, if an entire layer, layer or column at the end is left empty, the parameter shrinks to fit the new data.

  If a cell is left empty unintentionally or contains an invalid value according to the data type constraints (e.g., non-numeric characters in a FLOAT), the user receives error feedback.

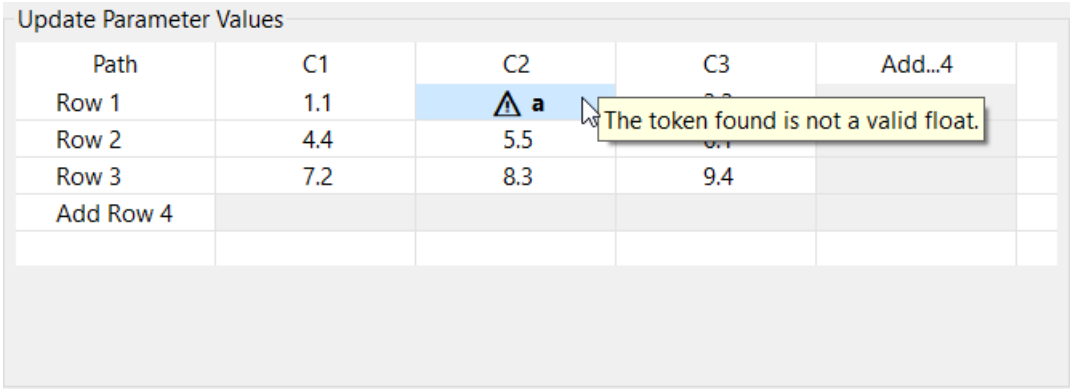

Figure 4-12: Example of marking of invalid value.

- **Click to write a new value.** In scenarios where the user clicks on empty cells adjacent to cells where a value already exists, new values can be entered. Upon confirmation and validation of the input, the corresponding row or layer expands by one column or row of zeros, respectively, accommodating the new data. Note that it is necessary just one click to start writing or enter if the tab is used. Also, a cell can only be edited if it is the first missing value in a row.

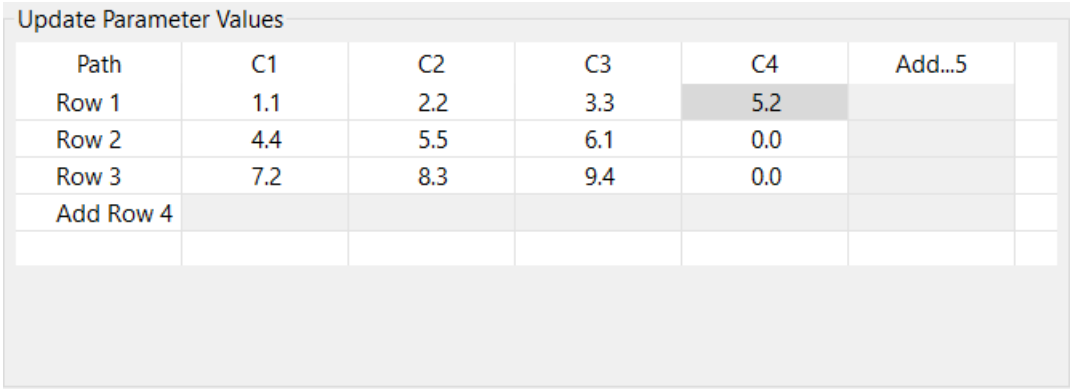

Figure 4-13: Writing a new value past the end expands the parameter.

- **Contextual menu for deleting.** Right-clicking on the header shows a menu that allows to delete the selected column, right-clicking on the “Path” column allows to delete the selected row or, in case of a 3D array, the selected layer, and finally, right-clicking on any other cell shows a menu that allows to delete the corresponding column, row or layer.

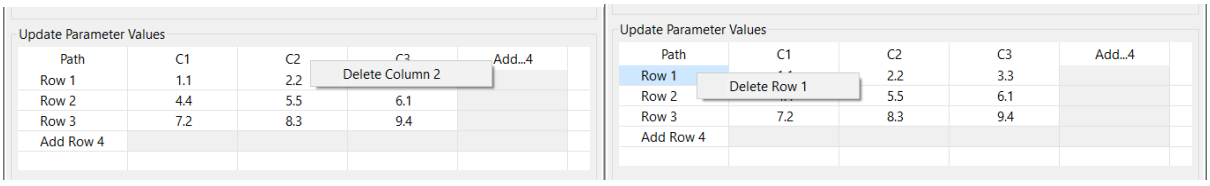

Figure 4-14: Context menu when clicking on the values editor header or on the "Path" column.

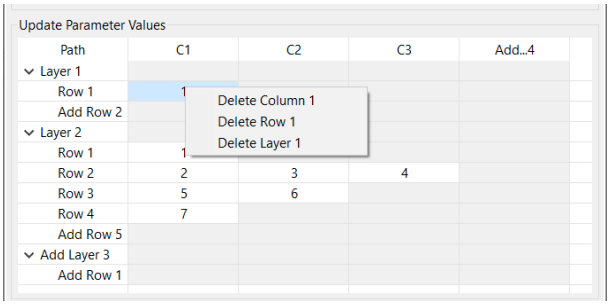

Figure 4-15: Context menu right-clicking on a cell in the values editor.

4.2.3.1. Inline Parameter Edition

If the user prefers, the parameter can be edited using the inline capability, which is accessed via the "F2" key when a given Parameter is selected or by Right-Clicking over the Parameter, opening a popup menu, as depicted in Figure 4-16, and selecting the "Edit inline" option.

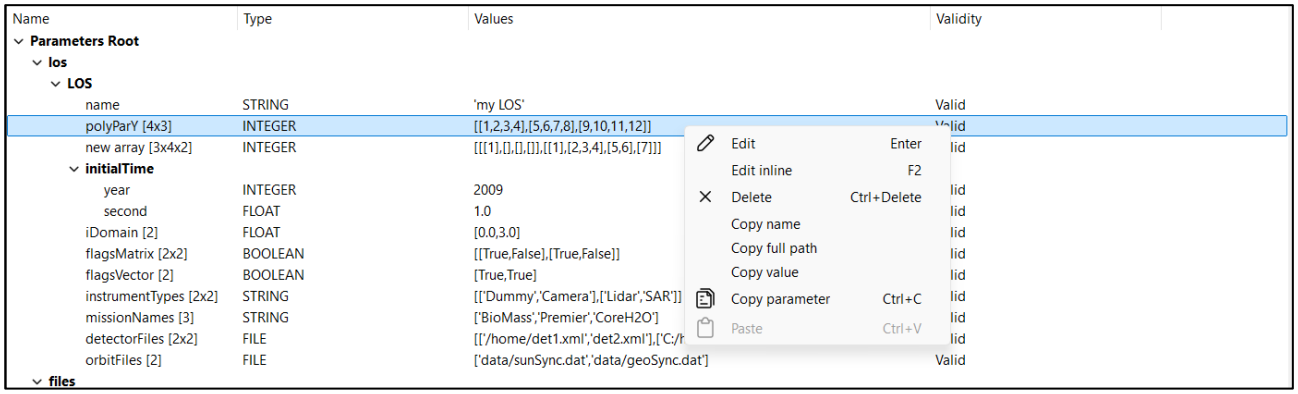

Figure 4-16: Popup menu accessible using a mouse right-click over a parameter.

The format used is the same as in openSF. The current representation (for non-scalars) is designed to be copied into Python. The same operation can occur in the other direction, from Python to the application. Valid Python syntax is accepted with the following additional restrictions:

- Elements must be literals. For example, `[1,2,3]` works but `[40+2,sum(range(1,10))]` does not
- All elements must have the same type. Thus, `["a",2,True]` is not valid input for a parameter.
- Only ARRAY parameters may have staggered dimensions
  - empty rows and layers are supported
  - empty columns are discarded
  - completely empty ARRAYs are not supported

In addition, for scalar parameters that can be represented as python string, the explicit string format is not needed, and a fully literal string is accepted. In other words, the string does not have to be between quotes, if it does not begin with a quote or a square bracket.

Note that this representation may change in future versions, and interoperability with Python (or any specific version or library) is not guaranteed in general.

4.2.4. Delete Parameter

From either the shortcut or by pressing the "Deletion" icon in the top toolbar, users are able to delete a selected parameter.

Note that the parameters tree allows the selection of more than one parameter[3] and delete action will erase from the system all the selected parameters.

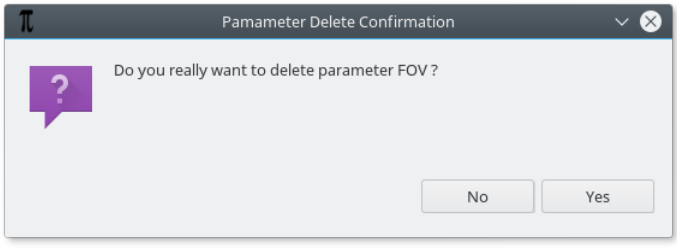

Figure 4-17: Parameter Deletion, Confirmation Message

4.2.5. Copy/Paste Parameter

By selecting on or more parameters and right-clicking on one of them it is possible to access a context menu that, among other things, allows the user to:

[3] To select more than one parameter, use Shift key for consecutive parameters and Ctrl key for non-consecutive selection.

- Copy a parameter/group: It copies one or more selected parameters to the system clipboard.
- Copy a parameter name: copies just the names of the selected parameters
- Copy a parameter full path: copies one or more parameter names, each preceded by every group or subgroup they are part of separated by a dot
- Copy a parameter value: copies the values of the selected parameters

When pasting one or more parameter or groups, there are two behaviours possible, depending on what is selected when the paste command is activated:

- If a single group is selected, parameters are pasted into that group, by replacing their common parent group with the selected group. For example, if parameters `a.b.cx` and `a.b.c.p` are copied, the common part is `a.b.`, and this prefix is replaced with the full path of the target group. It is possible for the common part to be empty.
- Otherwise, the previously copied parameters are pasted in the currently selected file. If another parameter with the same full path already exists in the given configuration file, a "_copy" string is appended to the name of the newly pasted parameter.

4.2.6. Consolidate groups and sort parameters

Once a parameter file is loaded into the ParameterEditor, the user can reorganize its contents in two ways: (1) by "consolidating" parameters that belong to the same group, and (2) by sorting parameters alphabetically. These actions are available through the dropdown menu attached to the icon showing interconnected nodes in the toolbar. Users are able to perform group operations on the loaded parameter files.

- **Consolidate Groups:** This option merges parameters that belong to the same group into a single group entry in the parameter tree. Once consolidated, redundant group entries are removed, and all parameters are shown under a unified group node.
- **Sort Parameters:** This option sorts the parameters alphabetically within each group, and also at the root level for any ungrouped parameters. It improves readability and allows users to more easily locate specific parameters.

The current state of the parameter tree is updated immediately upon selection of any of these actions.

4.2.7. Plot Parameter

The application also allows the user to plot parameter values in a 3D plot frame. This functionality uses a third-party library to handle 3D plots in Java (JMathPlot).

This functionality is only available for numeric parameters (FLOAT and INTEGER). If the user selects for plotting a 3D parameter, i.e. with several layers of values, normally just the first layer of the parameter is plotted. However, if the parameter is open for edition and a specific layer is selected, that layer is selected to be plotted instead.

Figure 4-18 shows the plot of an integer matrix parameter.

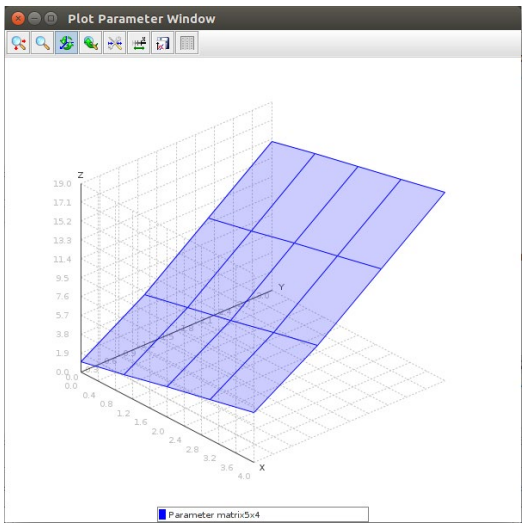

Figure 4-18: Parameter Plot Visualization

4.3. Side Panel

The ParameterEditor side panel can be found in the left side of the main frame and comprises the global status of the system showing:

- List of loaded configuration files
- Schema files loaded in the system

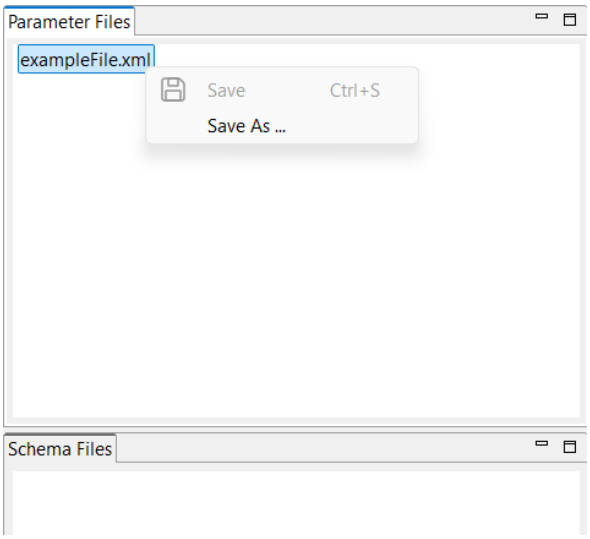

Figure 4-19: Side Panel Contextual Menu

This panel has also associated a contextual menu that allows to access extra functionality:

- Close configuration file
- Save only the selected file
- Save a file with a different filename
- When selecting a parameter's filename from the list, the tab corresponding for this file is focused.

The two groups, Parameter Files and Schema File, contained within the ParameterEditor side panel are expandable/collapsible through clicking on the arrow at the left-top corner of each block.

4.4. Log Console

The log console is a graphic panel that shows the system messages produced by the data/file management layer and the validation logic. Warnings and errors that arise while a file is loaded or while parameters are edited end up here. For example, if a group does not contain any parameters under it, then a warning is raised from OSFI, which will automatically go to the PE log.

This panel is resizable and can be found at the bottom of the ParameterEditor main window. Figure 4-20 shows the interface of the Log Console.

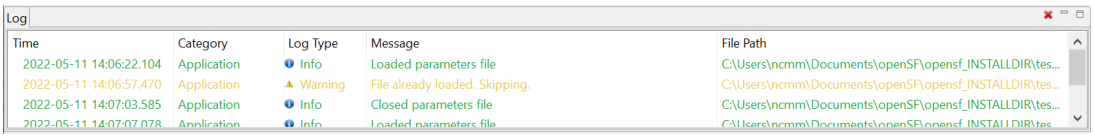

Figure 4-20: Log Console

A ParameterEditor log message has the following attributes:

- *Time*: system time when log was produced
- *Category*: software source of the log. Some of those are:

- *Application*: ParameterEditor status log
- *Parameter Parsing*: logs from the consistency check of the configuration files
- *Schema Validation*: logs corresponding to the schema validation process

- *Log Type*: depending on the log message impact: Info, Error, Exception, Warning, Debug
- *Message*: text describing the system event
- *File Path*: path of the related file

By right-clicking on the table it is possible to access a context menu that allows to:

- Toggle the visibility of some table columns
- Toggle the visibility of log messages relative to specific files
- Erase the log messages (this operation can also be performed with the red X in the top right corner of this tab)
- Copy a number of log messages. The log messages that are to be copied should be selected before opening the context menu.

4.5. Validation Process

By pressing the "Validate Parameters File" icon in the top toolbar, the user can now validate the list of parameters of the selected configuration file against the following conditions:

- **Type consistency.** If a parameter's value type is not consistent with the type reported for that parameter a *Bad syntax for the type* error is reported.
- **Dimension consistency.** If the number of values of a parameter is different than its specified dimensions, a *Dimensions inconsistent with value* error is reported.
- **Minimum/Maximum consistency**. If the parameters have minimum and maximum values, all the values are analysed and compared to them. If a value falls outside the [Minimum,Maximum] interval, a warning *Outside [min,max] range* is reported.

The result of this validation is shown in the Log Console, specifically in tab "ParameterParsing". Note that in Figure 4-21, since the *Outside [min,max] range* is a warning and not an error, the result of the validation performed for "exampleFile.xml" is still reported as successful with a warning. The same does not happen in Figure 4-22, where a *Bad syntax for the type* error occurs, resulting in error "VALIDATION FINISHED STATUS :: ERROR".

As one can see from the figures below, the messages reporting errors or warnings contain the name of the problematic parameter and the name of the parent configuration file. This way, if the user has multiple opened configuration files and/or a long list of Console messages, it will be easier to distinguish the log messages between one another.

Log

Time	Category	Log Type	Message	File Path
2020-05-29 14:45:34.313	Application	Info	Loaded parameters file	C:\Users\ncmm\Documents\openSF\test_files\sceneConf.xml
2020-05-29 14:45:45.851	Parameter Parsing	Info	STARTING VALIDATION OF FILE	C:\Users\ncmm\Documents\openSF\test_files\sceneConf.xml
2020-05-29 14:45:45.851	Parameter Parsing	Warning	Parameter Nbands => Max. range violation; Max=10.0 Value[...	sceneConf.xml
2020-05-29 14:45:45.992	Parameter Parsing	Info	VALIDATION FINISHED STATUS :: WARNING	C:\Users\ncmm\Documents\openSF\test_files\sceneConf.xml

Figure 4-21: Validation result in the log console (1)

Log

Time	Category	Log Type	Message	File Path
2020-05-29 15:39:04.320	Application	Info	Loaded parameters file	C:\Users\ncmm\Documents\openSF\test_files\sceneConf.xml
2020-05-29 15:39:12.300	Parameter Parsing	Info	STARTING VALIDATION OF FILE	C:\Users\ncmm\Documents\openSF\test_files\sceneConf.xml
2020-05-29 15:39:12.300	Parameter Parsing	Error	Error retrieving numeric value. Parameter: Nbands	sceneConf.xml
2020-05-29 15:39:12.400	Parameter Parsing	Error	VALIDATION FINISHED STATUS :: ERROR	C:\Users\ncmm\Documents\openSF\test_files\sceneConf.xml

Figure 4-22: Validation result in the log console (2)

Similarly, the Edit Parameter window performs as well a validation on the fly every time the user presses "Accept and Close" button, not allowing the user to save the parameter file while there are reported errors - Figure 4-23 show examples of two cases of wrong parameter values being inserted.

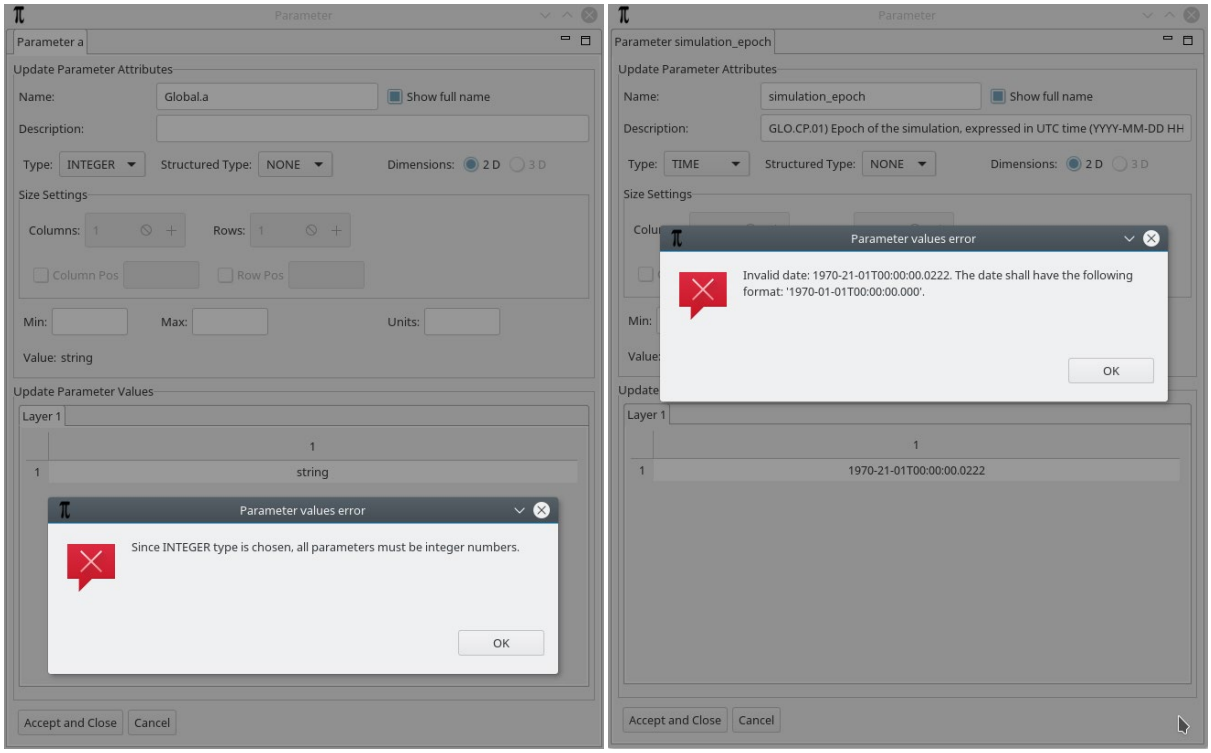

Figure 4-23: Errors when editing parameter: (a) the input value type is not in agreement with the selected parameter TYPE; (b) the input date is not valid (note the invalid month 21).

The Parameter Editor performs dimensional checking at this stage. The envelope dimensions are calculated and, if they are smaller than the declared dimensions, the user is prompted to trim the dimensions down. Because empty rows and layers are allowed, in practice only columns are discarded.

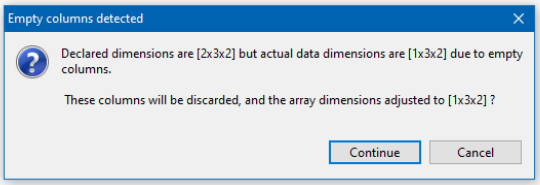

Figure 4-24: Trim columns confirmation dialog.

Because completely empty arrays are not supported, the user will be prompted to always insert some valid value before committing the changes.

4.6. Schema Validation

If the user wants to validate a given Parameters file against a Schema file, he can now do it by choosing both the Parameters and Schema files and then pressing the “Apply Schema File” button in the Shortcuts Toolbar – section 4.1.1.

The user will be prompted with a new log tab, “Schema Validation”, containing messages reporting possible errors in the Parameters file when validating against that given Schema file or even syntax errors in the used Schema validation file, if that’s the case.

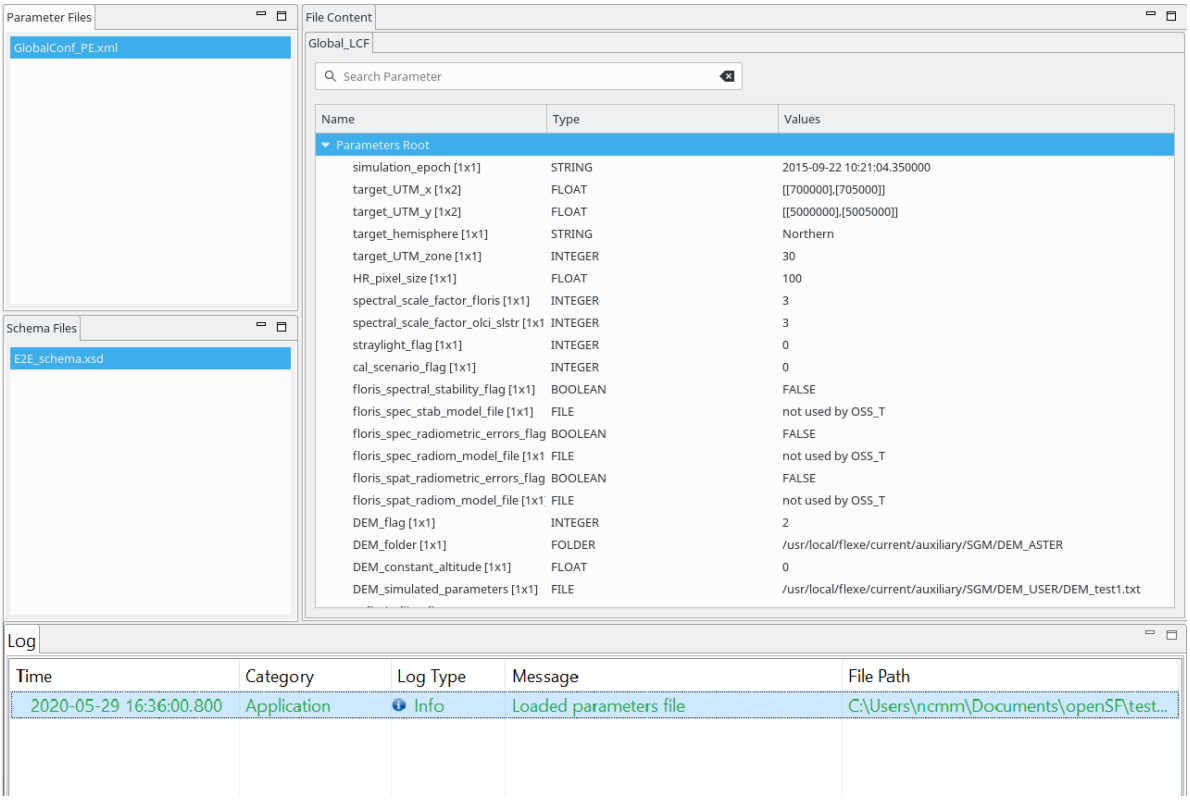

Figure 4-25: Schema Validation result in the log console

Note that the only Schema file format accepted at the moment is the XML Schema Definition (XSD)[4] language.

[4] https://en.wikipedia.org/wiki/XML_Schema_(W3C)

END OF DOCUMENT