User Interface for the Knowledge Engineer

Standard view

In the standard view for a Knowledge Engineer (see Quaestor Interface) the multiple document interface is shown, consisting of four windows (figure 1).
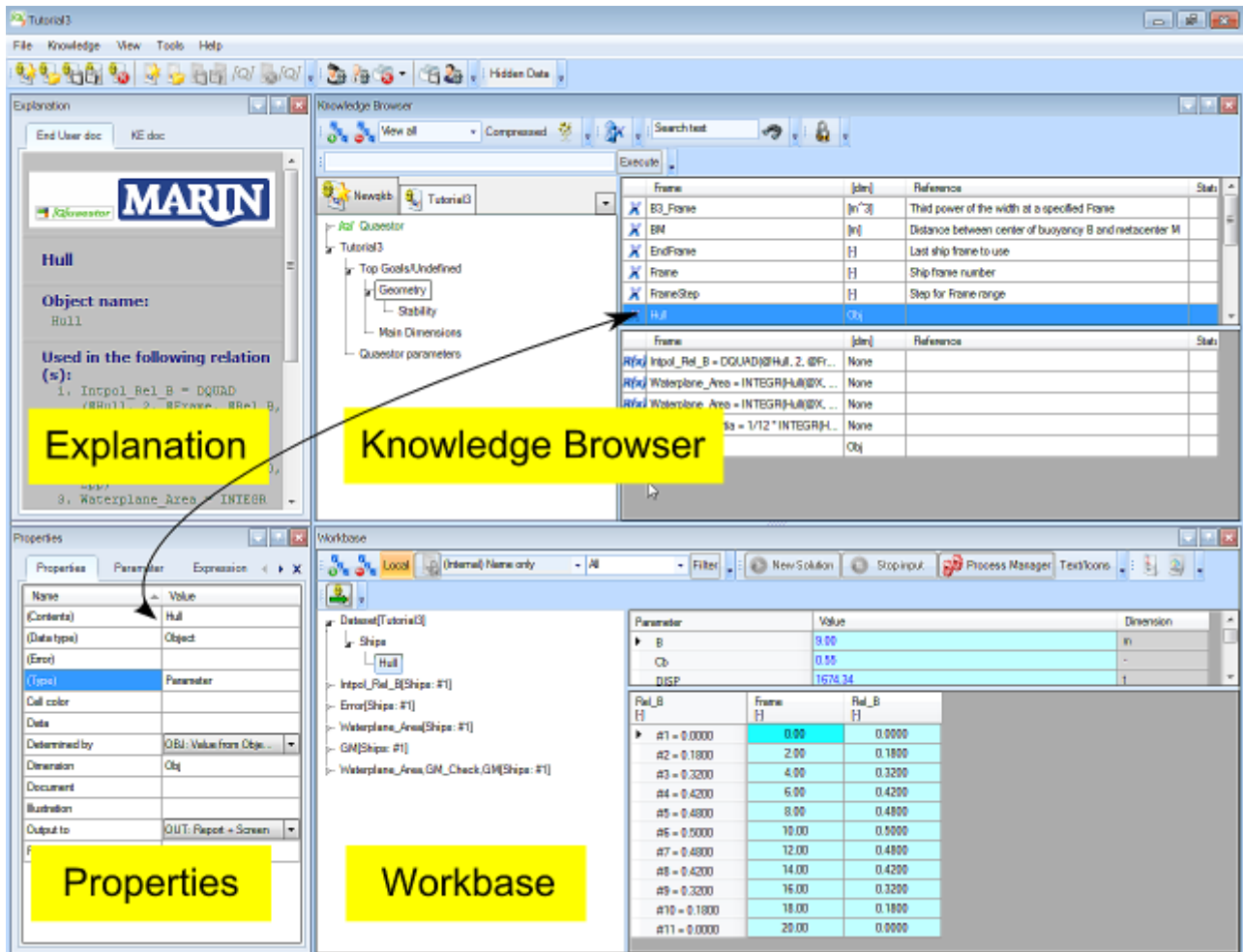

Figure 1   Basic Quaestor interface overview

These are the basic windows to work with Quaestor as a knowledge engineer with a corresponding function.

1. The **Knowledge Browser** provides acces to the classes, parameters, relations and constraints in the knowledgebase. It's the central place for adding and editing knowledge.
2. The **Explanation** window provides access to detailed information about a frame (a parameter, a relation or a constraint) in the knowledge base.
3. The **Properties** window provides even more details on the knowledge. It provides information on the way of use by the modeller.
4. The **Workbase** facilitates solutions and datasets, and provides the control buttons of the modeller.

This standard view can be modified in any desirable way by dragging and adjusting the size of the individual windows. For example, one might want to maximize the Workbase window when inspecting solutions. The standard view can be recovered by choosing *Window>Standard* in the main menu.

Besides the windows a **main menu** is shown. Items in the main menu are enabled when it is allowed to use the selected item. This will depend on a lot of things such as user level, position in the Knowledge Browser or Workbase, etc. The menu provides access to the standard and Quaestor-specific file handling and edit functionality. Furthermore, the basic handling with respect to the Knowledge Browser and Workbase is facilitated. However, realise that most of the functionality is enabled by **right clicking** () in either the Knowledge Browser or Workbase. The menu item *Tools* provides access to advanced options related to repair and debugging of knowledge bases and Quaestor settings.

Interconnection Knowledge Browser, Workbase, Frame Viewer and Slots & Properties

Information shown in the four basic windows is connected to each other, this enables quick detailed overviews while browsing in the Knowledge Browser or Workbase, and presentation of the relation between results and their origin.

Please note that the option "Connect Workbase to Knowledge Browser" under the Appearance tab of the Tools>Options window we create an additional connection between the Workbase and knowledge browser.

How this interconnection is shown depends a little on the type of application. Below several demonstrations of this interconnection:

- Flash demo of a simple knowledge base with Workbase-Knowledge Browser connection: click here....
- Flash demo showing the difference between Workbase-Knowledge Browser connection switched on and off: click here....
- Flash demo of a Taxonomy knowledge base with Workbase-Knowledge Browser connection: click here....

The Knowledge browser

In Quaestor, the Knowledge Browser provides access to the knowledge assembled in the knowledge base and the functionality of Quaestor. Together with the Workbase the Knowledge Browser is the most significant component of the user interface and offers all the necessary possibilities to adapt search and even combine knowledge databases. Basically, it consists of seven parts (figure 2).
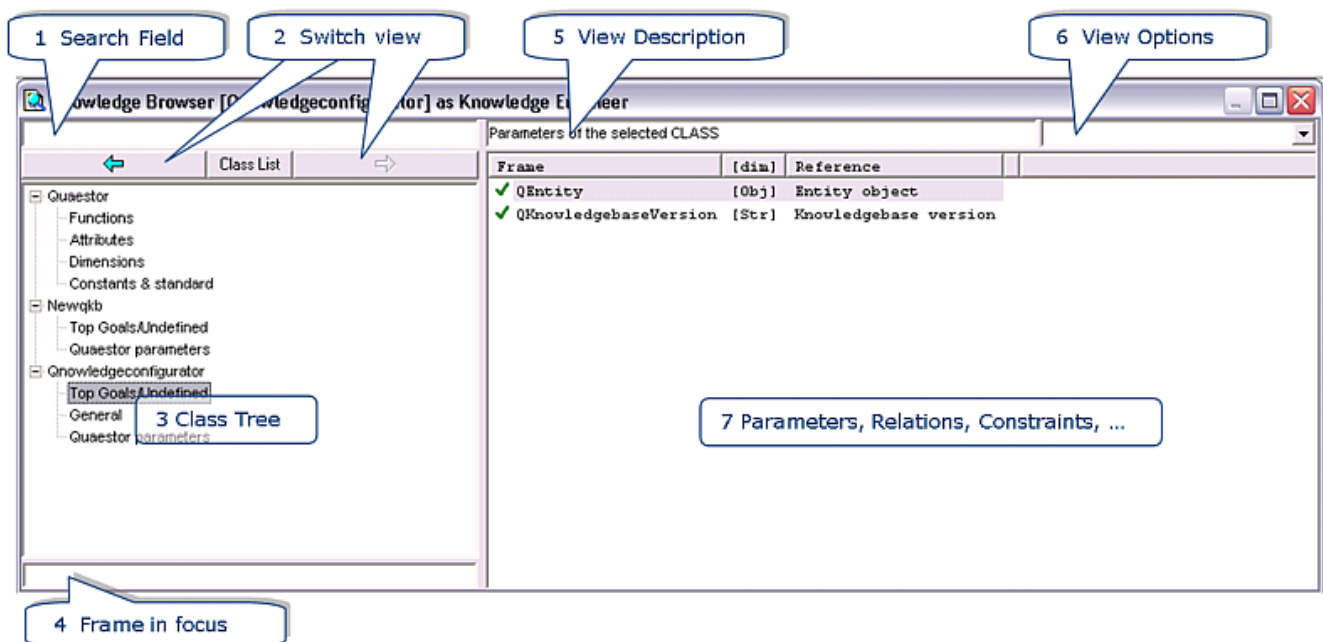

Figure 2   The Knowledge browser window

1. The **Search Field** is a dynamic and powerful way of searching the knowledgebase. While typing here, field 7 shows all frames (partially) corresponding to your search string.
2. The **Switch View Buttons** are the back and forward buttons of the Knowledge Browser, comparable to those in standard HTML-browsers. By pushing the button between the back and forward buttons ('Class List'), field 3 can be switched between a tree view mode , list view mode (default) and domain view mode (see below).
3. The **Class Tree** shows the knowledgebases currently opened, and the classes they contain. Furthermore, it provides access to the integrated support system by means of the *Quaestor node*. Show flash demo...
4. The **Frame in focus/Browse field** window shows the frame (parameter, relation, constraint,...) currently selected in the knowledge browser. Clicking once on a parameter, makes field 7 show the *Expressions in which the current parameter is present*, double clicking makes sure it shows the *relations that can be used to determine current Parameter*.
5. The **View description** describes what is shown in field 7.
6. The **View Options** pulldown menu provides extended options to the data shown in field 7. For example, one might only want to view parameters in relations, containing attributes or subgoals etc.
7. The **Parameter, Relations and Constraints** window shows the contents of the knowledgebase, depending on the view options selected in field 4 and 6. It's the central place to add and edit knowledge.

By means of these fields you are able to browse through all type of knowledge in the knowledge base. Please, go to the topic on the Knowledge Browser for more details about searching and browsing knowledge.

The Frame viewer

The frame viewer provides access and edit possibilities to detailed information of a frame. A frame is either a parameter or a relation in the knowledge base. Basically, it consists of 7 segments (figure 3).
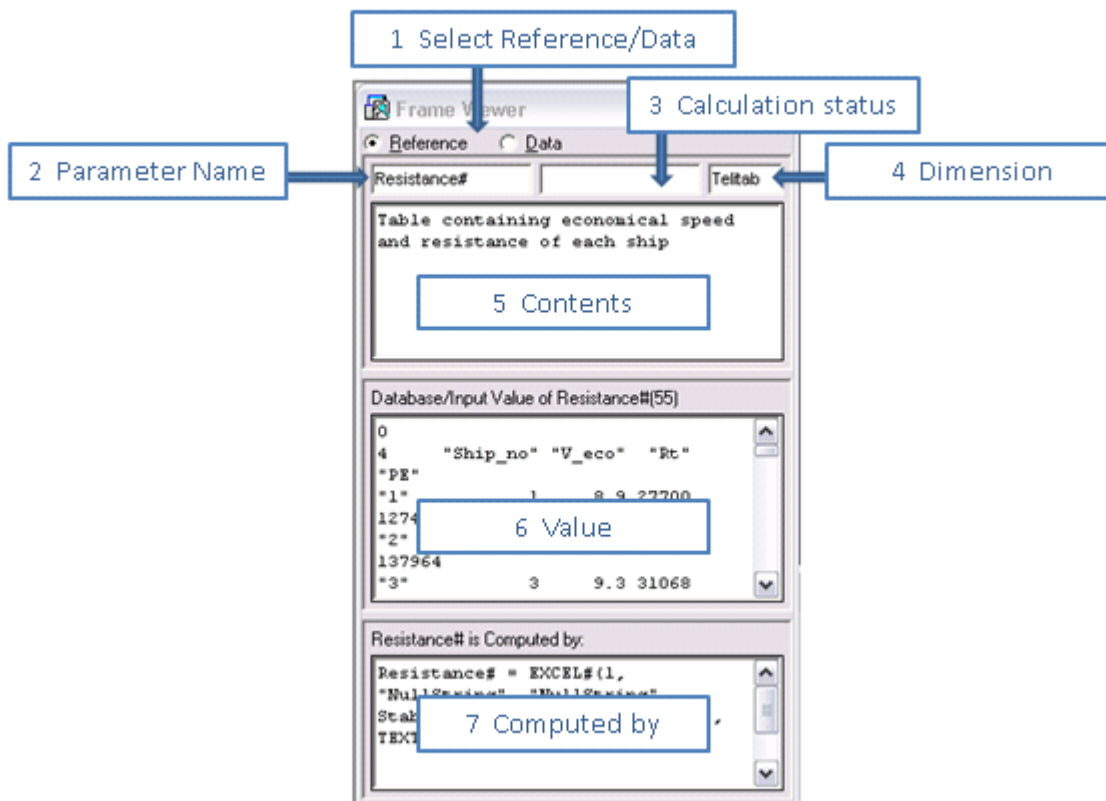
Figure 3   The frame viewer window

1. The **Select Reference/Data** radio buttons make it possible to switch between a reference mode and data mode. This will change the functionality of field 5.
2. The **Parameter Name** field holds the name of the parameter currently selected
3. The **Calculation Status** field shows the calculation status when selected in the Workbase (e.g. pending, determined etc.)
4. The **Dimension** field shows the dimension of the parameter, which can be changed.
5. The **Contents** field holds the reference or data, depending on the selection in field 1. In reference mode, a reference for the parameter can be assigned or changed. In data mode, it is the location to assign attributes to the parameter.
6. The **Value** field provides information on the value of the parameter if the parameter is used in the selected solution in the Workbase.
7. The **Computed by** field shows the relation by which the parameter was calculated.

The specific information shown in the Frame viewer will depend on the properties of the frame selected in the Knowledge Browser and the solution selected in the Workbase.

The Slots & Properties window

The Slots & Properties window (figure 4) provides even further details on the knowledge than the Frame viewer and makes it possible to provide information on the way of use by the modeller.
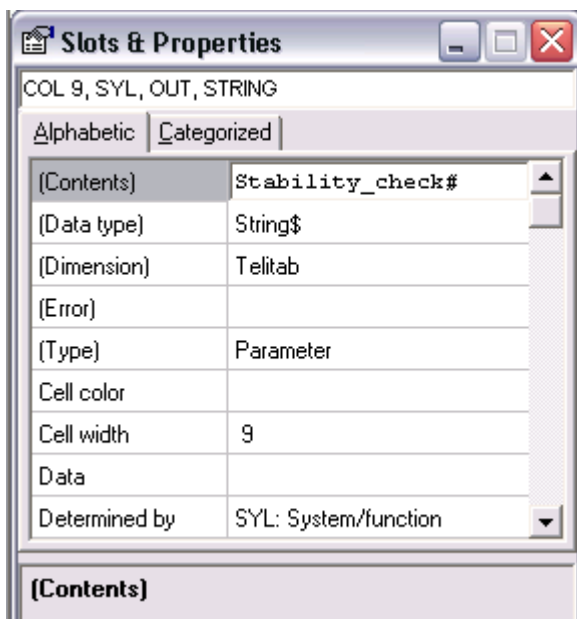
Figure 4   The slots & Properties window

The properties shown in this window depend on the type of knowledge (parameter, relation or constraint) selected. Moreover, some of the fields can also be edited in the Frame viewer. For a description of all fields please refer to the Extended KE Interface.

The Workbase

In Quaestor the Workbase (figure 5) facilitates the dialog with the user when a **solution** is modeled. Furthermore, it provides access to the present dataset, the Taxonomies in this dataset and solutions in a selected knowledge base.

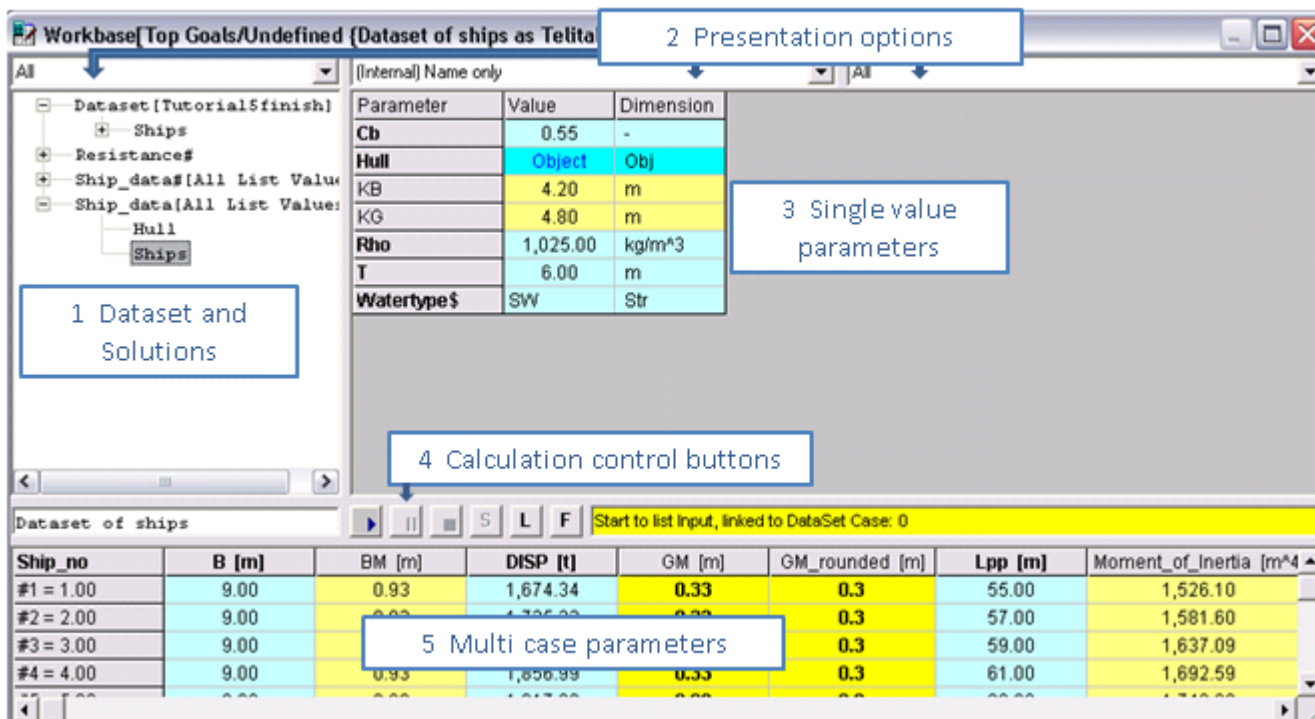*Figure 5   The Workbase window*

Please note that Figure 5 shows the window with the classic buttons. In addition to this buttons the Workbase buttons are introduced.

1. The **Datasets and Solutions** field shows data objects used as input for calculations (dataset node) and all solutions that are performed. Solutions are generally presented as [SolutionName(Dataset)]. Solution options (e.g. delete, recalculate etc.) are performed by right clicking a solution node.
2. The **Presentation Options** fields direct the output and presentation of the workbase.
3. In the **Single value parameters** field, parameters are shown that are independant of the case.
4. The **Calculation control buttons** can start, stop or pause solutions. Furthermore they provide acces to the process manager and filter options.
5. The **Multi Case parameters** field shows a table of parameters that are case-dependant, each row representing a different case.

For more details about the [Workbase](), please visit the [Workbase]() topic.