A Guide to Aquator

1 Application

Version 4.2

Oxford Scientific Software Ltd.

DISCLAIMER

Contents

Aquator

Introduction

Aquator is one of the most advanced solutions available for modelling water resource systems. It provides the professional hydrologist and water resource planner with a state-of-the-art system that allows any real-world water resource system to be modelled.

The Aquator solution is:

- component-based: models are constructed by joining together components that represent hydrological entities
- customizable: the rules that govern component operation can be modified
- extensible: new components can be added
- open-architecture: third-party software developers as well as the user can customize and extend Aquator
- co-operative: usable from other packages such as *Microsoft® Excel®*
- standards-based: uses an industry-standard component architecture and the industry's premier customization environment
- globally-optimised: with Aquator version 4, the new facility *Aqua Solver* looks for the 'best' solution for water movements needed to meet demand.

Component-based

Aquator is a component-based software package. An Aquator model is constructed from components which are placed on a schematic representation of the system and then joined together, using an intuitive point-and-click user interface. These components encapsulate the operating rules that govern their use. For example, a reservoir component will typically have control curves which the reservoir operating rules will use to determine releases from the reservoir.

Customizable

Aquator is customizable. It incorporates the world's most advanced and easy-to-use customization tool: *Microsoft® Visual Basic® for Applications* (*VBA*), the same technology used in the Microsoft Office suite of applications. With this tool the user can optionally customize the operating rules of any component. *VBA* would typically be used to modify a component whose basic operation was correct but which required relatively small enhancements. The skills and knowledge required to use *VBA* can be mastered by the hydrologist and professional programming expertise, while useful, is not a pre-requisite.

Extensible

The Aquator component system is extensible. A software development kit (SDK) can be made available for third-party developers to add additional components. The programming effort required is significantly greater than that of VBA customization, and would typically be done only when a different type of component than those installed with Aquator was required.

Open architecture

The design and implementation of Aquator components and Aquator itself is based on an industry-standard open architecture called COM (Component Object Model) defined by Microsoft and also used as the architecture of VBA. This means that literally millions of software developers throughout the world are familiar with COM and VBA, and would have the necessary skills to undertake either the customization of an Aquator component or the development of a completely new one.

Co-operative

The Aquator application itself can be used as a type of component. For example, water resource models can be run and results retrieved from Aquator while working inside *Microsoft® Excel®*. A GIS (Geographic Information System) application could load Aquator and create a model automatically from the information held in the GIS database.

Globally optimised

With Aquator Version 4, *Aqua Solver* looks for the ‘best’ solution, where ‘best’ means ‘cheapest’ or ‘best resource state’. A global analysis is performed where all water movements needed to meet demand are simultaneously evaluated. The advantage is that this global analysis completely prevents false failures without any additional effort on the part of the modeller (although it is still necessary to assign costs and define the resource state of each supply in an appropriate way). Note especially that VBA customisation can continue to be used.

Aqua Solver is implemented by default for new models but for compatibility with models developed under previous versions of Aquator, it is possible to disable *Aqua Solver* and run models in the traditional way.

Getting Started

Aquator is a software application for simulating complex water resource systems where the user constructs a representation of a system by 'drag and drop' of components from the Aquator Toolbox on to the screen.

Once the components have been linked together and assigned appropriate hydrological parameters, the model may be run over any defined period by a simple click on the Aquator Toolbar.

By setting appropriate attributes for the model run, full reporting of all or selected variables on a daily basis may by displayed in graphical and tabular form.

As Aquator is a richly-featured software application and it is difficult to absorb all of its capabilities the first time you use it, we strongly recommend that you read this section and then carry out Tutorial 1, Tutorial 2 and Tutorial 3 in the following sections.

The tutorials model very simple water resource systems but in the process you will learn the fundamentals of how to use Aquator. From there, using the on-line *Context-Sensitive Help*, you will be able to construct your own, more complex models and, in the process, explore the features of Aquator.

System requirements

The following versions of Windows are supported:

- Windows XP
- Windows 7

Aquator models require a powerful system in order to minimize execution time. Doubling the processor speed will typically halve the execution time.

- CPU speed: 1 GHz minimum, 2 to 3 GHz for larger models
- Memory: 256 MB minimum, 1 GB for large models and long simulation periods
- Disk space: 25MB for the program plus 10-200 MB per database dependent on project size

- Monitor: minimum 1024 x 768 resolution, 1280 x 1024 preferred (e.g. 17" LCD or 19" CRT)
- Pointing device e.g. mouse (models are constructed using a drag-and-drop method)

Installing Aquator

The install package for Aquator is downloaded from the Oxford Scientific Software website – please contact:

support@oxscisoft.com

for further instructions on how to access the download and the procedure for licencing the software

Follow the on-screen instructions presented in a series of dialogs. These will ensure that any previous version of Aquator is uninstalled before the installation proceeds – **it is not possible to run two different versions of Aquator on the same PC**. Choosing the default options where they appear will result in the Aquator executable and all associated files being installed in *\Program files\Oxford Scientific Software\Aquator*.

Uninstalling Aquator

To uninstall Aquator, first ensure you are running with administrator privileges.

From the **Start** menu go to **Settings...Control Panel...Add/Remove Programs**. Find and select the entry for Aquator and click on the button to uninstall.

This will remove every file that was installed but will leave any folders where Aquator databases are stored. The installation folder (normally *\Program files\Oxford Scientific Software\Aquator*) will not be removed if any additional file has been manually added to that folder after installation.

Running Aquator For The First Time

Before starting Aquator, we will briefly review how data is organised and stored, and define some terminology.

An Aquator **project** defines a **model** of a water resource system plus all the information needed to use the model. Often we use the words **project** and **model** interchangeably.

A project is stored in a single **database**. The version of Aquator current at the time of writing uses a *Microsoft Access* database (MDB) file for this purpose but future versions may use other database systems. You do not need to have *Access* installed since Aquator contains all the functionality required for reading and writing *Access* databases.

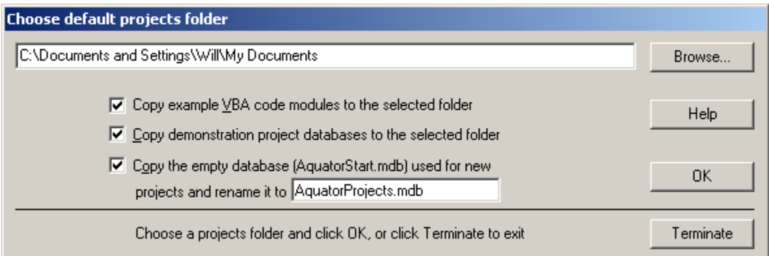
An Aquator **database** can contain

- any number of projects
- time series and profile data, shareable between projects
- pictures used to annotate model schematics, shareable between projects.
- the results from any number of model runs, shareable between projects.

When you start Aquator, you must specify the database you are using, and then which project in that database. Within the Aquator application itself you may have any number of project windows open i.e. as many projects simultaneously loaded as you wish.

First Time Use Dialog

The very first time Aquator is run, depending on your installation, you may see a splash screen appear briefly but the first window to appear which requires any user interaction is the Default Projects folder dialog:



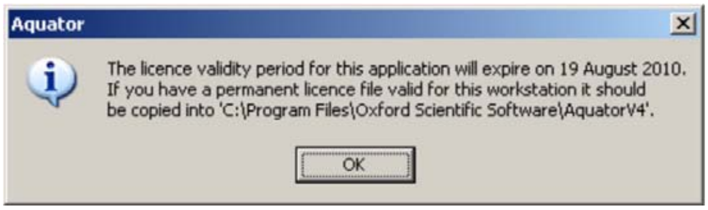
Use this to specify the default location for your database and projects. It is recommended that you click on *Browse…,* and make a new folder under **My Documents***.* For more information on this see *Default Projects Folder Dialog* *p. 151*.

Tip: if you select *Make New Folder* immediately after you have clicked *Browse…*, you may miss that you have created a new folder with name **New Folder** within **My Documents** because the standard Windows behaviour is not very clear when it does this. To rename it to something more useful at this point simply locate the folder, right click with the mouse and select **Rename**.

Leave the three check boxes on this dialog ticked. This will result in several databases being copied to your chosen folder. One of these will be a new empty database for your projects (you can choose a name for this database in the above dialog). The remainder will be databases containing demonstration projects.

Licence Validity

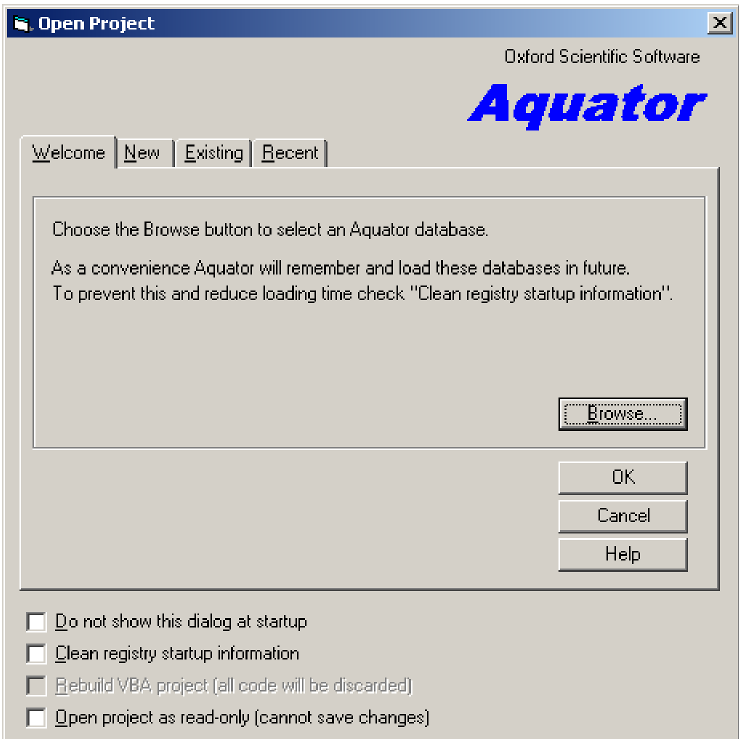
When you have setup your default projects folder a licence information dialog will appear.



If you have not received information on how to license Aquator from Oxford Scientific Software you may click OK. For further information see Licencing Aquator.

Normal Startup Dialog

When you have setup your default projects folder, the Startup dialog appears:

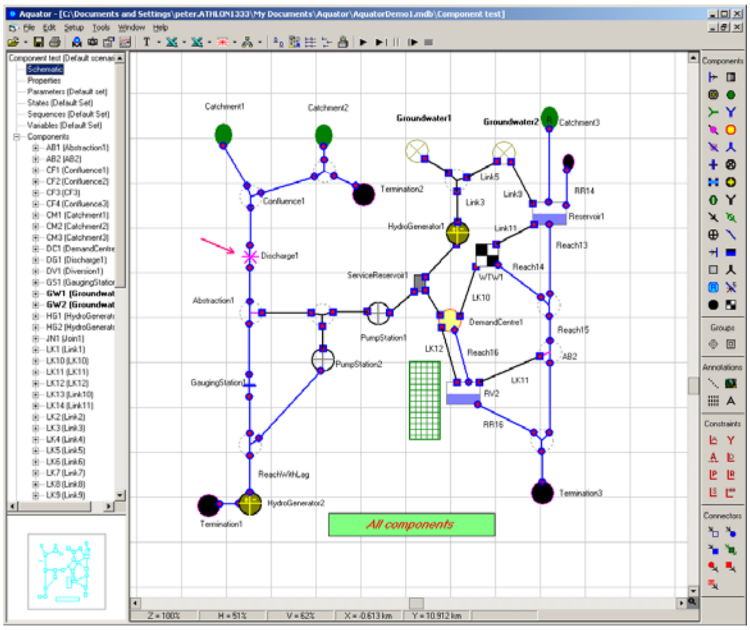


This window allows you to specify the database and project you wish to use at startup. We will look at how to use this in the tutorials. For details on the different options see *Startup Dialog* *p. 93*.

For now, click Browse and choose AquatorDemo1.mdb (this will have been copied to your default projects folder). Then choose the *Component test* project on the *Existing* tab and click OK.

The Main Aquator Window

Once you have specified the database and project, the main Aquator window will be displayed:



Within this window you use following items:

1. The **Aquator Menu Bar** at the top for accessing menu commands
2. The **Aquator Toolbar**, just below the menu bar, provides shortcut buttons for commonly used actions
3. The **Aquator Toolbox** adjacent to the right-hand edge provides access to the objects used to construct a model

The window contains three panes:

1. The **Aquator Tree View Pane** along the upper-left edge presents all the objects and properties of the project in a hierarchical *Microsoft® Windows Explorer* format.
2. The **Aquator Information Pane** is the large central area which displays information for whichever node is selected in the Tree View Pane; in the above example the model schematic has been selected
3. The **Aquator Thumbnail Pane** at lower-left displays a miniature image of the entire model schematic plus a rectangle showing the portion of the schematic visible in

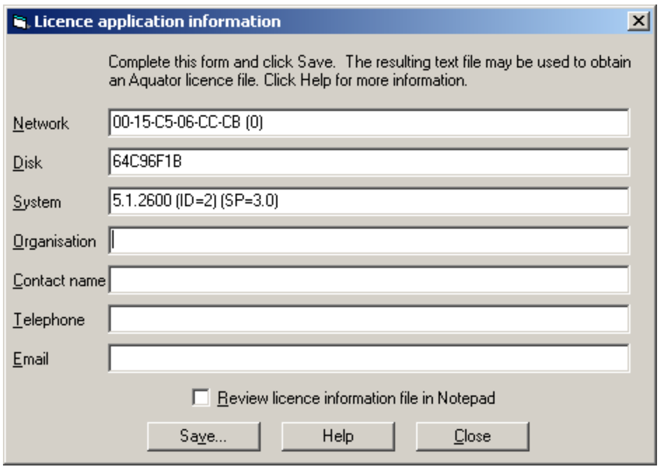
the information pane; dragging the rectangle with the mouse allows easy navigation around a large schematic

Licencing Aquator

As installed from download, Aquator can be used for evaluation purposes for a period of 30 days. Further use of Aquator is enabled by copying a small licence (.LIC) file into the Aquator installation folder.

This licence file is specific to each machine. To obtain a licence file follow this procedure.

- Run Aquator but do not load a project (click Cancel on the Startup dialog as necessary)
- Click on the **File...Licence info** menu command, which will display the form shown below
- Fill in the form and click Save to save the required information in a small text file, named like ATHLON1333.lin (Here ATHLON1333 is the name of the machine but any valid filename will do)
- Logon to the Aquator user are on the oxscisoft.com website: http://oxscisoft.com/aquator/userlogin. (You will need to have received a user name with instructions from Oxford Scientific Software to do this. Contact support@oxscisoft.com for further information).
- Follow the instructions you have received from Oxford Scientific Software which describes how you can upload the .LIN file to the website and then receive an email from the website with a .LIC file attached which should be copied into the Aquator installation folder:
- Test by running Aquator to verify that the Licence Information dialog no longer appears



Tutorial 1: Parameters and Variables

In this tutorial we will set-up one of the simplest water systems possible: a **Groundwater** source linked to a **Demand Centre**. The Demand Centre will have a demand that varies by month throughout the year. When we run the model we will see how that demand is satisfied by the Groundwater source.

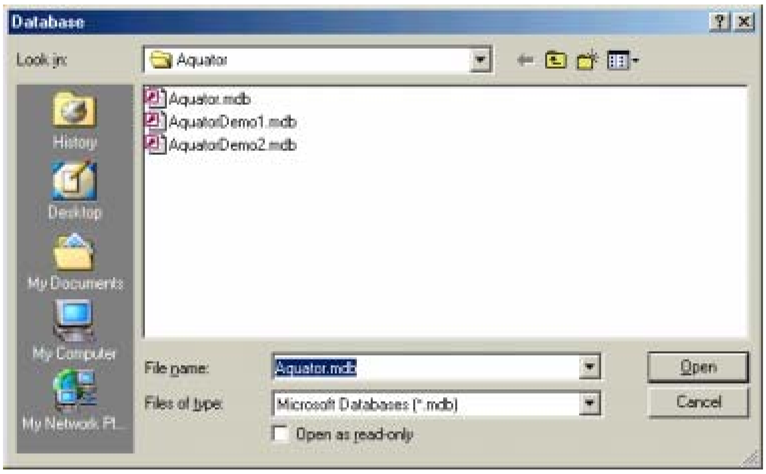
In the process of the tutorial we will learn how to:

1. Open Aquator and set-up a new database and project.
2. Add a Groundwater component to the water system schematic.
3. Add a Demand Centre component to the water system schematic.
4. Set Component Properties for the Demand Centre and Groundwater.
5. Join the Groundwater to the Demand Centre with a Link.
6. Set Component Parameters for the Demand Centre and Groundwater.
7. Run the model.
8. Change what output data is displayed and run the model again.

Open Aquator and set-up a new Project

Run Aquator and when the *Startup Dialog*(*Startup Dialog p. 93)* is displayed, select the **New** tab and click on the button labelled "...". This will display the Open File dialog.

At this stage we need to choose an Aquator database in which to store the project. Aquator provides an empty database for this purpose which was copied to your default projects folder when Aquator was first executed. Select this database file, bearing in mind that you may have changed the MyAquatorProjects.mdb default name of this file to e.g. Aquator.mdb, as shown in the following dialog.

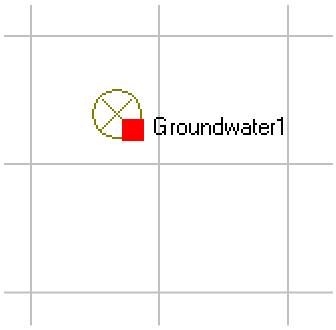


Click on the *Open* button. Back on the **New** tab of the startup dialog, replace the default name of the new project with a name of your choosing e.g. "Tutorial 1". Click *OK* and the Aquator main window will appear.

Add Groundwater Component

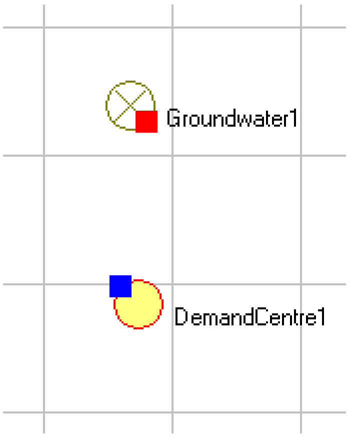
You should see a model schematic, empty apart from a grid, displayed in the **Aquator Information Pane**. We will now add a Groundwater component to the schematic as follows:

1. Left click and hold on the Groundwater component symbol in the **Aquator Toolbox.**
2. With the mouse button held down, drag this component on to the schematic in the Information Pane.
3. Release the mouse button leaving an instance of the Groundwater on the schematic:



Add Demand Centre Component

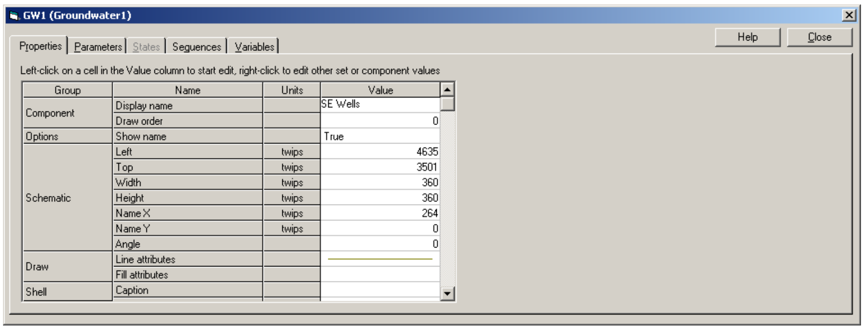
Repeat the process above with the Demand Centre component and drop it onto the schematic below Groundwater1. Your schematic should look like this:



Set Component Properties

Now that we have instances of these two components on the schematic, we can give them particular names by changing a **Component Property**. Firstly we will change the Groundwater name as follows:

1. Select Groundwater1 by right-clicking on it (notice that a grey selection box appears around the component).
2. From the resulting pop-up menu choose *Properties* by either (a) clicking directly on the *Properties* item, or (b) by using the up and down cursor keys to select the *Properties* item and then pressing the *Return (Enter)* key
3. Select the cell in the *Value* column, *Display name* row for editing by either (a) clicking in that cell, or (b) moving the focus rectangle to that cell using the cursor keys and then pressing *F2*. This is the general way to initiate an edit.
4. Change the text in that cell from *Groundwater1* to *SE Wells*. The result should appear as below:



5. Click on the *Close* button, or press the *Return (Enter)* key twice, and notice that the name of this component has now changed on the schematic.

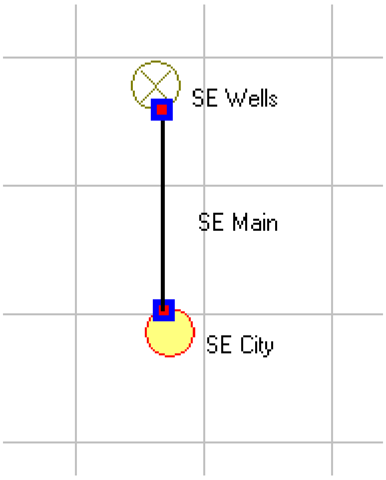
6. Repeat the same procedure and change the name of *DemandCentre1* to *SE City*.

There are many Properties associated with a Component and they vary for different types of component. However they can all be changed by this method. For details see: *Properties Node* *p.109*.

Join Groundwater to Demand Centre with Link

We shall now join *SE City* Demand Centre to *SE Wells* Groundwater source with a Link as follows:

1. Left click on the Link component symbol in the Aquator Toolbox.
2. With the mouse button held down, drag this component on to the schematic in the Information Pane.
3. Release the mouse button leaving an instance of the Link on the schematic
4. Notice that the Link component has a blue square at one end and a Red Square at the other. The blue square represents a **man-made (or Supply type) Input** and the Red Square represents a **man-made (or Supply type) Output**. The rule in Aquator is that an output from one component can **only** be joined to the input of another component. Thus to attach the Link to the Groundwater, left click on its blue square and, with the mouse button held down drag it towards the red square of the Groundwater component.
5. Notice that as you get close to the Groundwater Red Square, the size of the Link Square doubles. At this point you may release the mouse button and the end of the Link will drop onto the Groundwater. You can be sure that the connection has been made if you see a blue outline to the Red Square on the Ground water component. Notice also that if you left click on the Groundwater component and, keeping the mouse button held down, drag it over the schematic, the Link remains attached.
6. Repeat this procedure to join the Red Square of the Link (**output**) to the blue square of the Demand Centre (**input**).
7. By default, Links are not normally labelled on the schematic, however if you set the Link Component Property *Display Name* to *SE Main* and set the Property *Show name* to *True*, your schematic should appear as below:



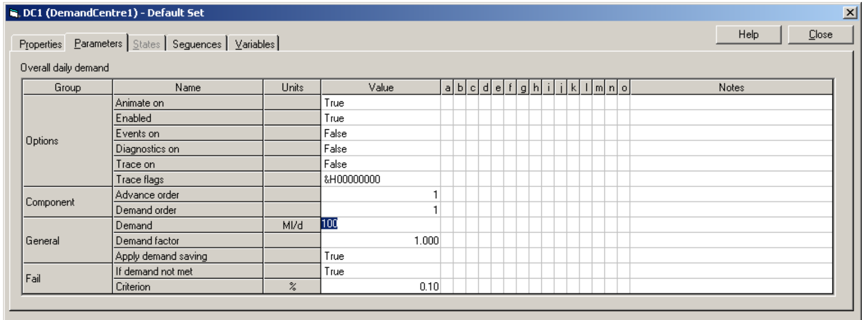
In general, joining components in Aquator is simply a question of dragging and dropping inputs onto outputs. However there are strict rules about how components can be joined, for details see: *Rules for Linking Components* *on p. 100.*

Set Component Parameters

Now that we have set-up the scheme for this simple model, we need to specify at least one hydrological/water resource **Parameter** for each component before we can run the model in a meaningful way.

Firstly we will specify a demand for *SE City*. At this stage we will keep it simple and specify a constant 100 Ml/day throughout the year. To do this carry out the following:

1. Select *SE City* by right clicking on it with the mouse.
2. From the resulting pop-up menu choose *Parameters* by either (a) clicking directly on the *Parameters* item, or (b) by using the up and down cursor keys to select the *Parameters* item and then pressing the *Return (Enter)* key.
3. In the *DC1 Parameters* window select the cell in the *Value* column, *Demand* row for editing by either (a) clicking in that cell, or (b) moving the focus rectangle to that cell using the cursor keys and then pressing *F2.*
4. Change the value to 100. The result should appear as below:

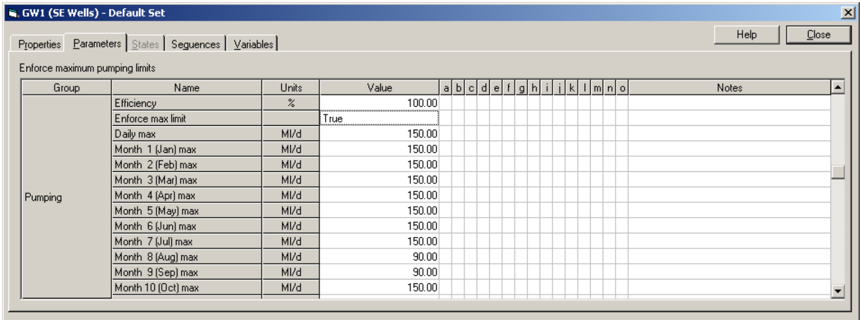


Group	Name	Units	Value	a b c d e f g h i j k l m n o	Notes
Options	Animate on		True		
	Enabled		True		
	Events on		False		
	Diagnostics on		False		
	Trace on		False		
	Trace flags		&H00000000		
Component	Advance order		1		
	Demand order		1		
General	Demand	Ml/d	100		
	Demand factor		1.000		
	Apply demand saving		True		
Fail	If demand not met		True		
	Criterion	%	0.10		

5. Click the *Close* button, or press the *Return (Enter)* key twice, to return to the schematic.

Now we wish to set the maximum pumping output from *SE Wells* for each month during the year. In order to see how the model handles system failures, we will set the maximum pumping output to less than the demand during the summer months. To do this carry out the following:

1. Select *SE Wells Parameters* in the same way as above i.e. right-click on the groundwater symbol on the schematic and choose *Parameters*. Notice that the details of *GW1 Parameters* window are different from *DC1 Parameters*.
2. Click in the cell in the *Value* column, *Pumping.Enforce max limit* row, or move the focus rectangle to that cell using the cursor keys and then press *F2*. A small toggle button will appear, which is used to change the value of boolean (True/False) parameters.
3. To toggle the value of the *Pumping.Enforce max limit* parameter from *False* to *True* press the *Space* key or alternatively click the small button using the mouse.
4. To move to the next cell below press the **down-arrow** cursor key, or, if using the mouse, click in the next cell
5. In the *Value* column next to the *Daily max* row, change the value to 150 and press **down-arrow**. This should take you to the next row: *Month 1 (Jan) max*.
6. Now repeat this for all the months in the year, giving values of: Jan: 150, Feb: 150 etc. except for Aug and Sep where the values should be 90. Notice that you use the **down-arrow** key for moving to the next field in a table. The result should appear as below:



7. After entering the last value of 150 click the *Close* button, or press the *Return (Enter)* key twice, to return to the schematic.

There are many Parameters associated with a Component and they vary for different types of component. However they can all be changed by this method. For details see *Parameters* and *General User Interface Rules.*

Model Run 1

We are now ready to run this simple model. From the Parameters we have specified, we would expect the water system to fail in the months of August and September, when the maximum pumping capacity of *SE Wells* is less than the Demand from *SE City*.

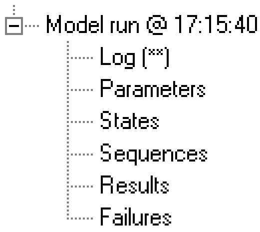
Before we see how this result can be displayed, we need to review the set of buttons on the **Aquator Toolbar,** which control how the model is run.

There are 5 buttons which, from right to left,

1. **Run** the model from start date to finish date,
2. **Run To** a specified date,
3. **Pause** the model run,
4. **Step** the model run one day,
5. **Stop** the model run.

Firstly click on **Run**. By default this will run the model for one year starting on the first day of the current calendar year. Notice that over a period of a couple of seconds (or less on fast computers), numbers are displayed under each component name, the colour of *SE City* briefly flashes Red and when all this has finished, an extra node *Model Run* appears in the **Aquator Tree View Pane**.

Results for Model Run 1



At this stage it is interesting to click on the following nodes of the Model Run.

<u>Model Run</u>

This displays the overall attributes of the run. Change the name to *Model Run 1* by clicking the button to the right of the *Name* field.

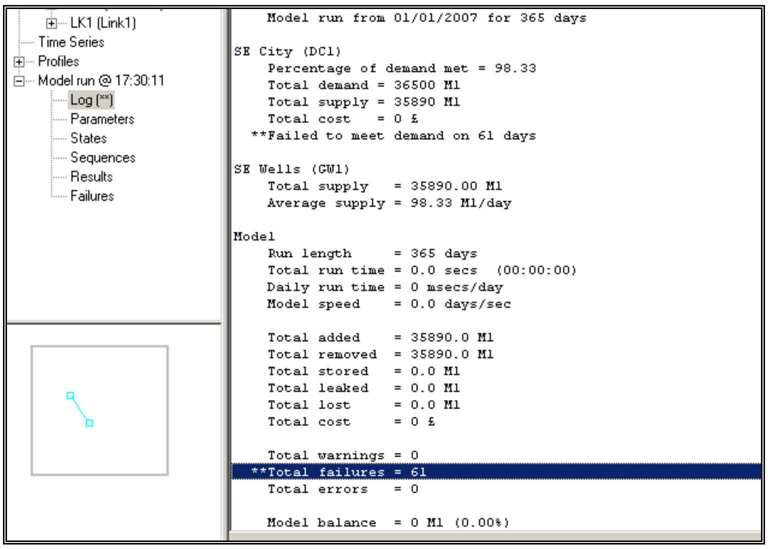
<u>Log (**)</u>

This displays a log of the run. The (**) symbol indicates failures, in this case failures to meet demand. Warnings will display as (*) while errors will display as (***). The additional symbol (!) is shown after the word "Log" if the model run terminates early for any reason.

Amongst the information shown for the Log node you will find a line:

Total failures=61

This is an indication of the failure to meet demand during the 61 days of August and September.



Parameters

This will display the values of the Parameters for all of the components in the system during the run. This provides an 'audit trail' for parameter values used to generate these results.

States

This will display the values of the States for all of the components in the system during the run. This provides an 'audit trail' for state values (starting values) used to generate these results.

Sequences

This will display the names of the links to time series and profile data stored in the database and used to provide data for all of the components in the system during the run. This provides an 'audit trail' for which time series and profiles were used to generate these results.

Results

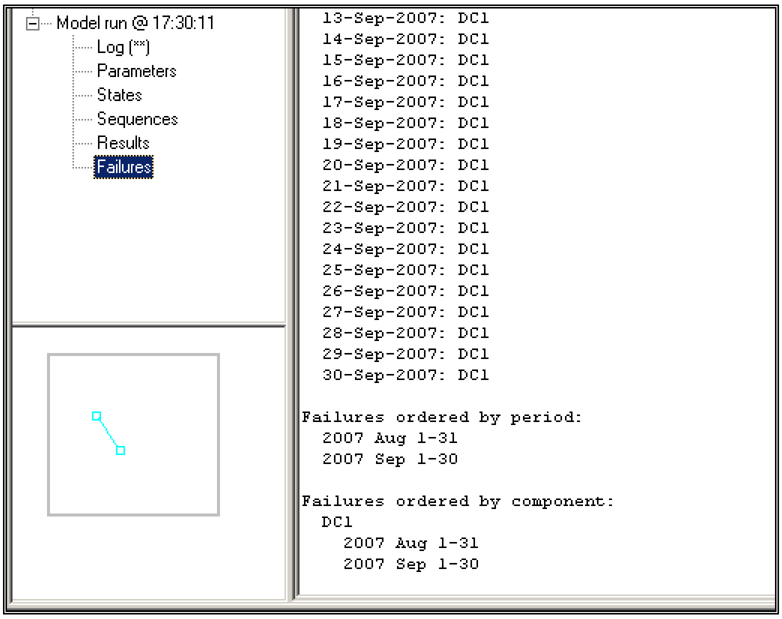
This gives results for each day of the run for the project and each of the components. Notice that by clicking the navigation buttons at the top of the form, you can display the results for one day during the model run. Also the calendar control allows a specific day to be selected.


01 January 2000 |<< << < 31 December 2000 > >> >>| 31 December 2000

Object	VBA Key	Group	Name	Units	Value
Tutorial 1 05/Jun/2001 14:30:36	Model		Stored	Ml	0
			Leaked	Ml	0
			Added	Ml	100
			Removed	Ml	100
			Lost	Ml	0
			Failures		0
SE City	DC1		Demand	Ml/d	100.000
			Supply	Ml/d	100.000
SE Main	LK1		Outflow	Ml/d	100.000
SE Wells	GW1		Flow	Ml/d	100.000

Failures

This node provides an analysis of all failures by date, period, and component.



Using Step and Run To Buttons

Return to the schematic display by either clicking on the *Schematic* node in the Aquator Tree View Pane or by pressing F10 or by clicking in the thumbnail view at bottom-left or by clicking on the Schematic button  on the Aquator Toolbar.

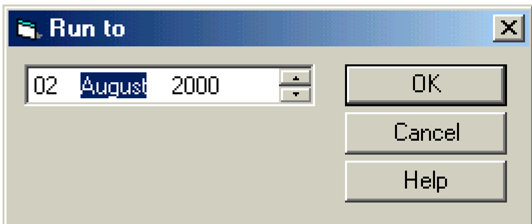
Now click on the **Step** button (if you have forgotten which this is, use **Tool Tips** *p. 102,* to find the right button). Notice that another Model Run is started and under each component name the schematic displays a numerical value for 01 January, i.e.

for *SE Wells* it displays the **Flow**,

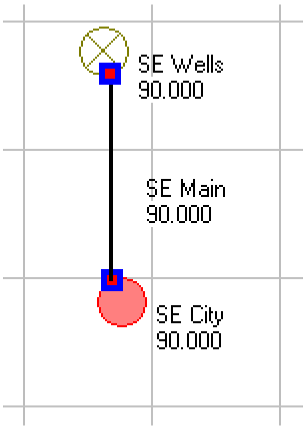
for *SE Main* it displays the **Outflow,**

for *SE City* it displays the **Demand**.

You may step through the whole year in this way, however for our simple model nothing changes until August. To get there quickly, click on the **Run To** button. A dialog appears and if you click on the month and then the down the month will change. Use this to select *02 August* as shown below:



Click on *OK* button and the schematic should now display the values for 2 August, as shown below:



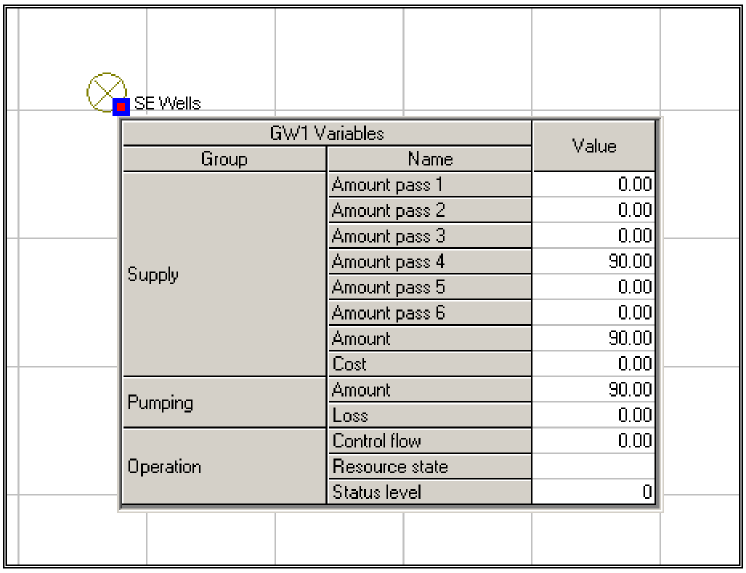
Notice that *SE City* now appears in Red, which indicates a failure to meet demand. To stop the model run, click on the **Stop** button.

Notice that each time you activate a run with one of these buttons, a new Run node is created. If you wish to delete any of these, simply click on the Model Run node and press the **Delete** key.

Viewing variables while paused

When paused during a model run you can view the current value of all variables for any component. First, scroll so that the desired component is in view. This is most easily done by dragging in the thumbnail window with the left mouse button held down.

Then click and hold down the left mouse button while hovering over any component on the schematic. A popup grid appears whilst the mouse button is held down showing the variable values for that component.



Set the Variables to be displayed on the schematic

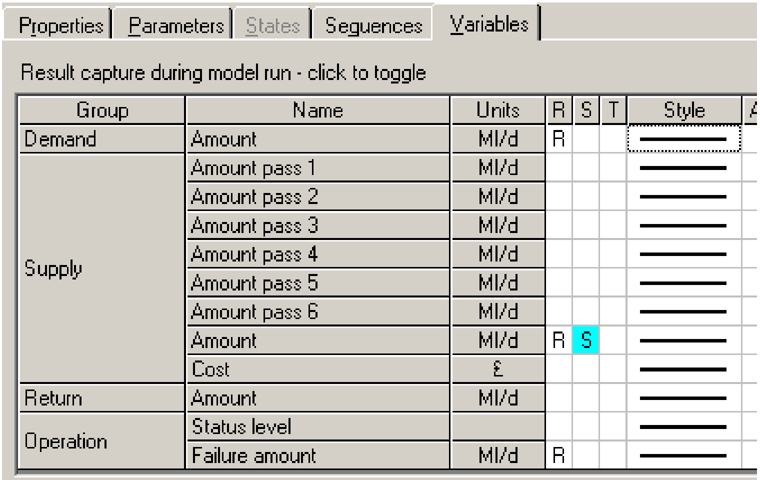
The output data from the model run are specified by the **Variables** of each Component or the Schematic. So far our model run has displayed the default variables and only in text form.

We will now set the following:

- Include the *Status Level* variable for *SE City* in the results,
- Chart the *Demand, Supply* and *Status Level* over the year for *SE City*.

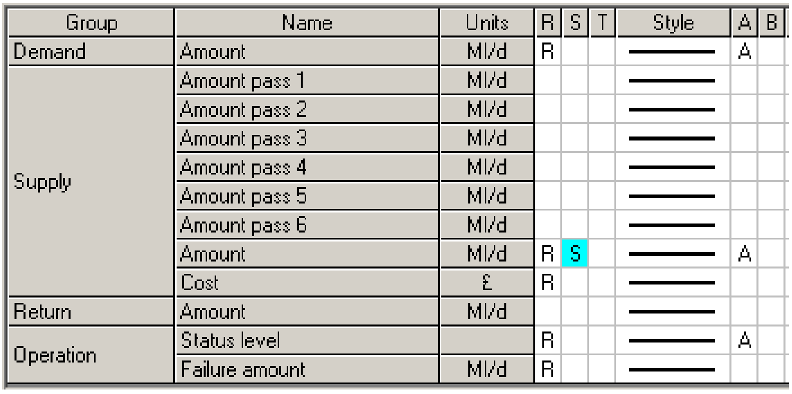
This is achieved by the Variables for *SE City* as follows:

1. Select *SE City* by right clicking on it with the mouse.
2. From the resulting drop-down menu, select *Variables*.
3. The *DC1 (SE City) Variables dialog* will appear, from which all variables for a component can be set. Click on the grid cell under the *R* column, next to the *Status Level* row. (Notice that the Tool tip text in the left top corner of this dialog displays *Result capture during model run – click to toggle*.)



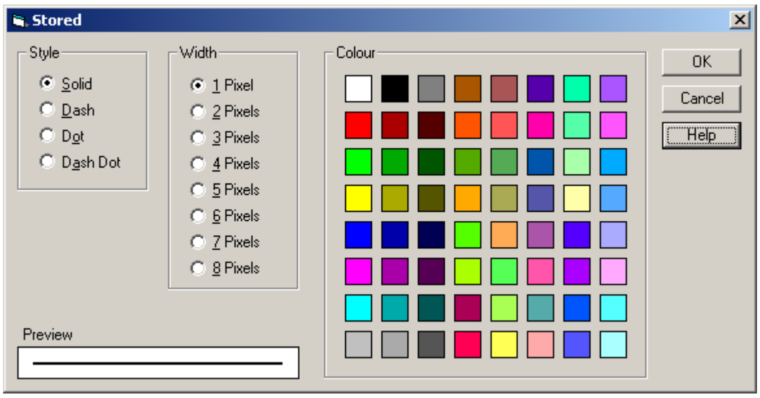
Group	Name	Units	R	S	T	Style	A
Demand	Amount	Ml/d	R				
Supply	Amount pass 1	Ml/d					
	Amount pass 2	Ml/d					
	Amount pass 3	Ml/d					
	Amount pass 4	Ml/d					
	Amount pass 5	Ml/d					
	Amount pass 6	Ml/d					
	Amount	Ml/d	R	S			
	Cost	£					
Return	Amount	Ml/d					
Operation	Status level						
	Failure amount	Ml/d	R				

4. Click on the grid cell under the *A* column, next to the *Demand* row. (Notice that the Tool tip text in the left top corner of this dialog displays *Plot on Chart – click to toggle*.)
5. Repeat this procedure for column *A* and the *Supply* and *Status Level* rows. We have now set *Status Level* to be reported in the model run and set Chart A to display *Demand, Supply* and *Status Level*:



Group	Name	Units	R	S	T	Style	A	B
Demand	Amount	Ml/d	R				A	
Supply	Amount pass 1	Ml/d						
	Amount pass 2	Ml/d						
	Amount pass 3	Ml/d						
	Amount pass 4	Ml/d						
	Amount pass 5	Ml/d						
	Amount pass 6	Ml/d						
	Amount	Ml/d	R	S			A	
	Cost	£	R					
Return	Amount	Ml/d						
Operation	Status level		R				A	
	Failure amount	Ml/d	R					

6. Finally, in order to change the colour in which the *Demand* curve is displayed, under the *Style* column, next to the *Demand* row, click on the grid cell. On the resulting *Line Attributes Setup Window*, click on one of the Blue colour squares so that a Blue Line of thickness 1 pixel is displayed in the *Preview* box as shown below:



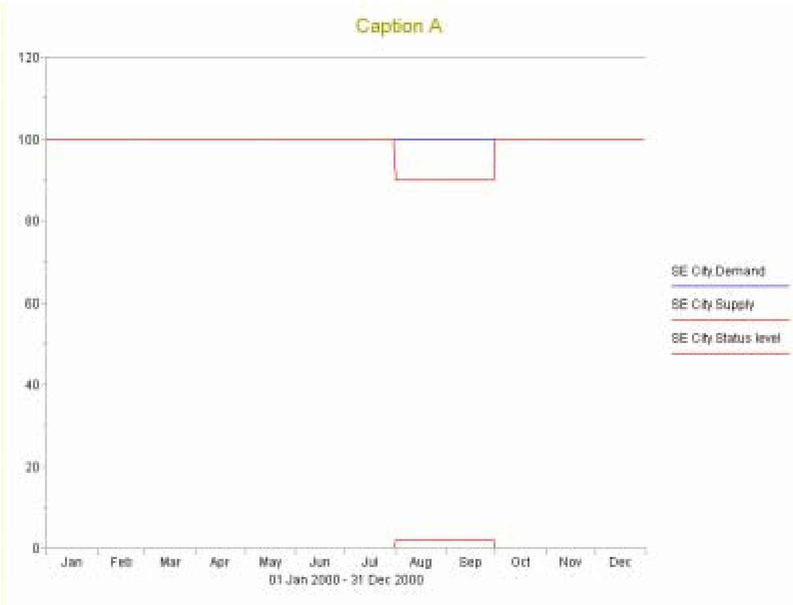
7. Click on *OK*. Now all the Variables we need have been set. Notice the Cyan shading of the shading of the *S* grid cell for *Supply*. This indicates that of the three variables set to be part of the Model Run results, only *Supply* values will be displayed on the Schematic. Click *Close* to return to the schematic.

Model Run 2

Now **Run** the model as before. Notice that an extra node: *Chart A* appears in the *Model Run* node in the **Aquator Tree View Pane**.

If you click on *Results* node you will see that *Status level* has been added to the results for the component *DC1*. If you look at the results for any day in August or September, you will see that the value for *Status level* has changed from 0 (OK) to 2 (Failure).

If you click on *Chart A* node, you will see a chart for *Demand, Supply* and *Status Level* over the year as shown below:



It is possible to customize the appearance of this chart by changing the caption, colours, annotation etc. For details see *Charts* *p. 191*.

Where To Go Next

Although a very simple model, this tutorial has demonstrated the way in which you set-up a water resource system model and run a simulation to produce Results in both tabular and graphical formats.

Before you move on to construct more complex models, we strongly recommend that you work through Tutorial 2 and Tutorial 3. In Tutorial 2 you will learn how to use **Sequences** and work with one of the more complicated components that Aquator uses: the **Reservoir**. In Tutorial 3 you will see how *Aqua Solver* can automatically prevent false failures in demand.

Tutorial 2: Sequences and States

In this tutorial we will set-up a simple water system consisting of a **Reservoir** storage between a **Catchment** and a **Termination**, connected by river **Reach** components. The flow from the Catchment will be determined by *Time Series* data stored in the database which will cause the Reservoir to fill up and spill into the lower Reach which will transport it to the Termination.

As the name implies, time series data is data which is time stamped. Aquator can use daily and/or monthly time series data. Aquator can also utilise profile data where a profile is either 366 daily values or 12 monthly values, repeated for each model run year.

Time series and profile data are held in the database independently of any model. Then a link is made between any component's requirement for time series or profile data and the actual time series or profile to be used.

To avoid having to repeat the phrase "time series or profile" everywhere the word *Sequence* is used. Thus a component's requirement for *Sequence* data is satisfied by linking each sequence to a time series or a profile or both. In the latter case the profile data fills in any gaps in the time series data.

In the process of the tutorial we will learn how to:

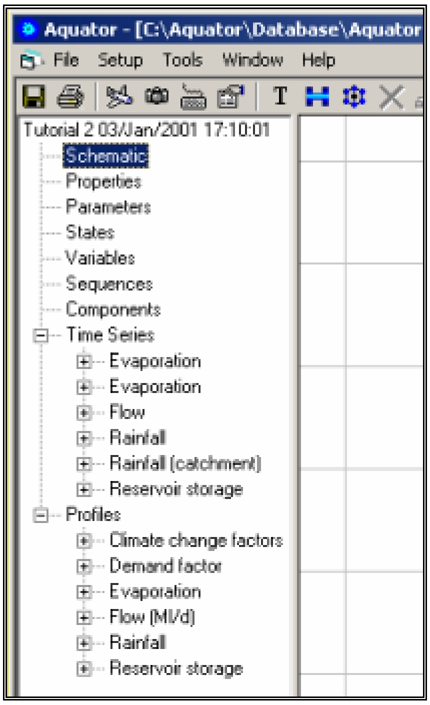
1. Open Aquator and set-up a new project with an existing database.
2. Add a new connection to a Component.
3. Assign *Time Series* flow data to a Component.
4. Change the *State* of the project for the start and finish dates.
5. Run the model and chart the results graphically.

Open Aquator and set-up a new Project with an Existing Database

Run Aquator and when the *Startup Dialog* is displayed, select the **New** tab and click on the browse button labelled "**...**". This will display the Open File dialog.

1. Click on the file *AquatorDemo1.mdb* and click on *Open*.
2. On return to the *Startup Dialog*, click on the *Project name* field and edit the text to be: *Tutorial 2 [date and time]* and click on *OK* button. The Aquator Main Window will appear.

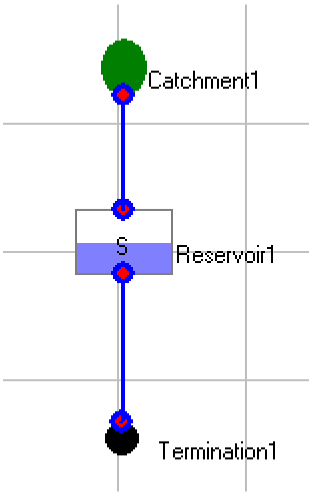
Notice that in the *Aquator Tree View Pane*, the nodes *Time Series* and *Profiles* have a **+** symbol beside them. This is because the database *AquatorDemo1.mdb* contains demonstration time series and profile data. If you click on both of these **+** symbols, you should see something like the figure below:



Setup Catchment, Reservoir and Reaches

Following the techniques of Tutorial 1:

1. Place a Catchment, Reservoir and Termination on the schematic.
2. Join the Termination to the Reservoir using a Reach component. Notice that it is not possible to use a **Link** for this purpose because now the connection points are circles *(see Rules for Linking Components* *on p. 100*.
3. Drag and drop another Reach onto the schematic for linking the Reservoir with the Catchment but notice that the Reservoir does not have an input to match the Reach's output.
4. To remedy this, right click on the Reservoir and select the *Add river inflow* command from the drop down menu.
5. Now join the Catchment to the Reservoir using the reach in the usual way. Your schematic should appear as below:



Assign Time Series data to Catchment

We will select the time series *Flow D* to prescribe the flow from the Catchment into the upper Reach. To do this, carry out the following:

1. Review the time series data. In the *Aquator Tree View Pane*, right click on the **+** symbol by the *Flow node* under the *Time Series Node.* It will display the available flow time series. Click on *Daily Flow* which should display a table of data in the *Aquator Information Pane*.
2. Using the date picker and button controls above the table verify that the time series spans the years from 1987 to 1996 but with some missing data. We will use the data for 1994 which should be complete.

30 April 1987 | I<< | << | 01 January 1994 | >> | >>I | 19 February 1996

Day	Jan	Feb	Mar	Apr	May	Jun
1	203.73	83.03	141.78	463.71	33.87	17.63
2	491.96	67.74	129.77	195.18	32.14	33.26
3	439.17	264.30	112.15	160.01	33.18	43.11
4	612.06	220.06	96.08	463.10	38.28	32.14
5	723.08	107.31	91.76	337.31	39.23	29.72
6	556.76	118.89	79.57	185.41	36.72	19.35
7	238.38	147.83	76.64	167.10	42.16	31.02
8	176.08	101.43	66.70	258.85	30.24	28.43
9	268.44	85.45	95.21	370.83	25.32	18.06
10	449.54	73.87	90.20	297.73	25.92	22.55
11	268.53	350.09	80.70	152.84	25.23	18.23
12	392.08	179.45	73.70	120.18	22.90	17.97
13	613.87	113.62	71.37	111.02	27.22	18.75
14	319.33	95.82	56.94	84.41	29.29	17.19
15	200.88	89.77	153.19	77.67	61.86	16.16
16	164.16	87.35	105.15	75.43	49.08	15.90
17	130.90	101.61	80.18	64.28	36.63	16.07
18	121.22	92.53	102.47	57.89	26.27	15.38
19	120.87	78.62	204.68	55.56	25.83	14.77

3. Return to the schematic display, right click on the Catchment and select *Sequences* command from the drop down menu.
4. Under the *Assigned Time Series* column, next to the *Flow* row, click on the field. On the resulting pop-up dialog, click on the check box: *Daily Flow* followed by a click on *OK*. The resulting dialog should look like that below:

Required sequence	Type	Assigned Time Series
Flow	Flow	Flow D
Climate change	Climate change factors	(none available in database)

We have now assigned the *Daily Flow* Time Series data to the Catchment. It is possible to assign Time Series and Profile data to different types of Components, for details see *Sequences Node*, *p. 113*.

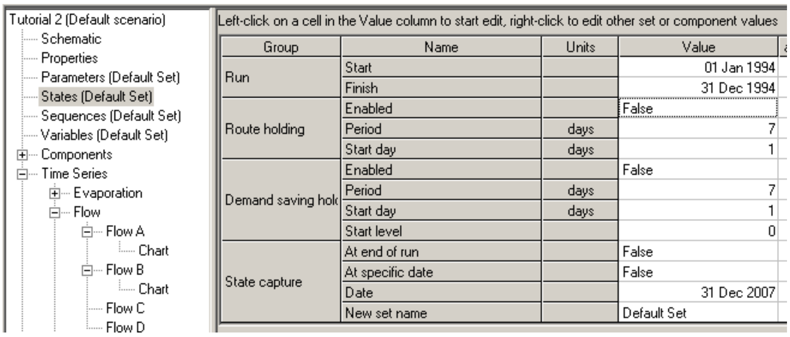
Set New Start Date Using Project State

Because we are now using "real world" data which varies over a number of years, when we run the model, we may wish to specify a particular *Start* and *Finish Date*.

The *Daily Flow* data is complete for the year *1994*, and we can make the model run from *01 January 1994* to *31 December 1994* by changing two of the Project *State* values the as follows:

1. Click on the *States Node* for the *Project* in the *Aquator Tree View Pane*. (This is the 4th node down from the top).

2. Change the *Run Start* date to 1 Jan 1994. As always you begin editing a value by either clicking in the cell or by moving the focus rectangle to the required cell and pressing the F2 key.
3. Change the *Run.Finish* date to 31 Dec 1994. The result should be as shown below:



Group	Name	Units	Value
Run	Start		01 Jan 1994
	Finish		31 Dec 1994
Route holding	Enabled		False
	Period	days	7
	Start day	days	1
Demand saving hol	Enabled		False
	Period	days	7
	Start day	days	1
	Start level		0
State capture	At end of run		False
	At specific date		False
	Date		31 Dec 2007
	New set name		Default Set

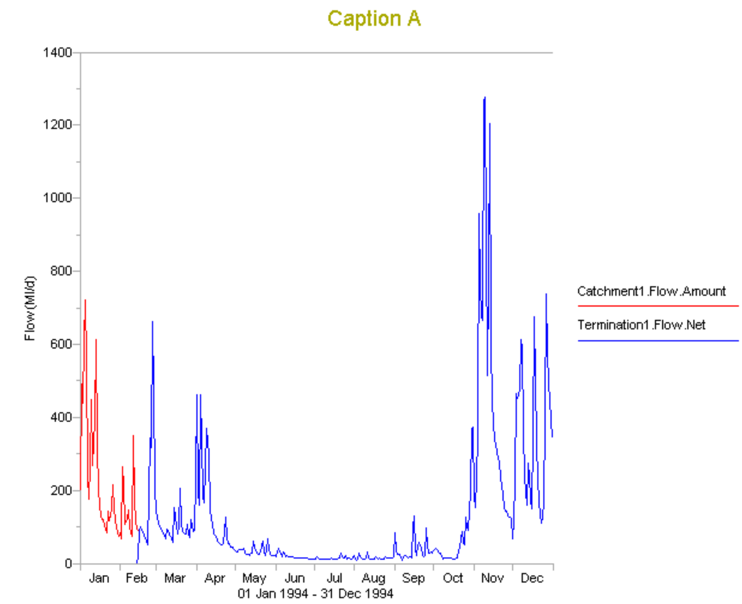
Set Variables to be Charted and Run the Model

Following the techniques of Tutorial 1 we will set the model run to chart **Catchment Flow** and **Termination Inflow** on **Chart A** and **Reservoir Calculated Storage** on **Chart B**.

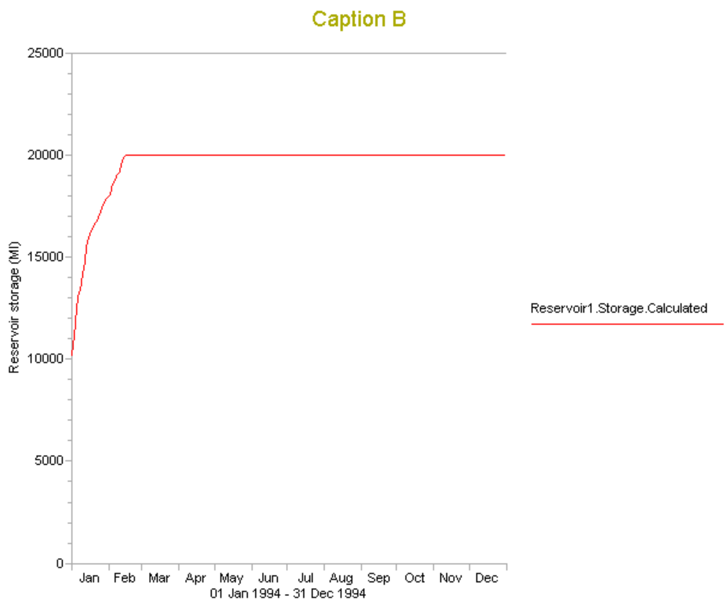
1. For the Catchment, set the *Variable: Flow.Amount* to be captured and plotted on Chart A with colour Red say.
2. For the Termination, set the *Variable: Flow.Net* to be captured and plotted on Chart A with colour Blue say.
3. For the Reservoir, set the *Variable: Storage.Calculated* to be captured and plotted on Chart B with colour Black say.
4. By default the Termination component's name is not shown and hence no Termination variable's value is shown on the schematic at runtime. Right-click on the Termination and toggle the *Options.Show name* property (not parameter) so that the name is shown. By default the net flow at the termination will now be shown at runtime.

Now run the model following the techniques of Tutorial 1. Notice that if you *Step* through the model that the Reservoir fills up completely by *20 march 1991* and begins spilling into the lower Reach (i.e. the Termination Inflow is no longer zero).

If you *Run* the model to the *Finish* date, the charted flows from the Catchment and to the Termination will appear as shown below:



And the charted calculated storage will appear as:

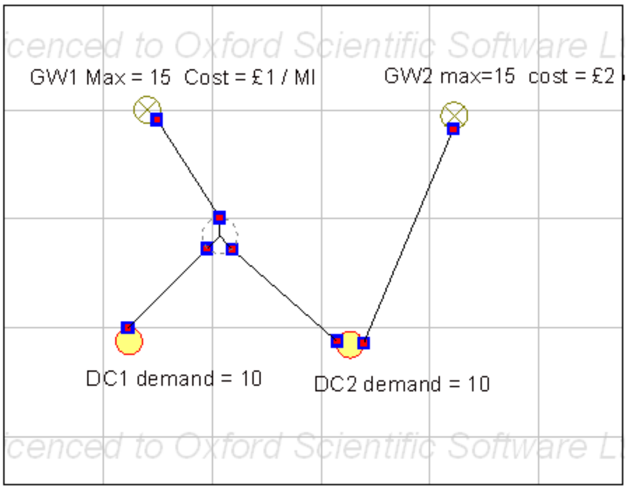


Tutorial 3: Using *Aqua Solver*

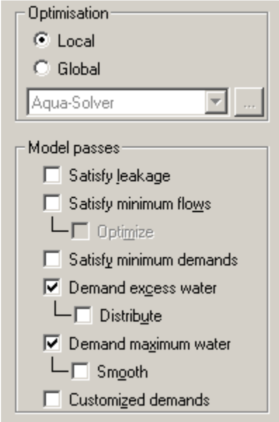
In this tutorial a simple model is constructed that initially does not meet demand but which meets demand once the *Aqua Solver* global optimiser is enabled.

Set up this model as follows (see schematic below):

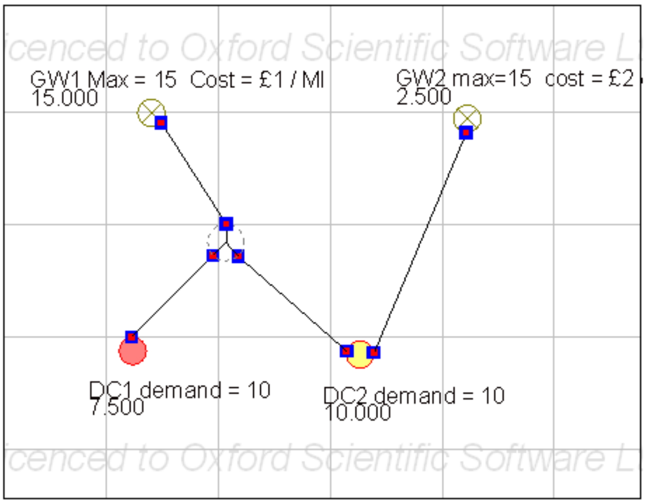
1. Each demand centre (DC1 and DC2) has a fixed demand of 10 Ml / day
2. Each groundwater (GW1 and GW2) has a maximum flow of 15 Ml / d
3. GW1 supplies cheaper water at £1 / Ml
4. GW2 supplies expensive water at £2 / Ml



Go to the model setup dialog ( button on the toolbar) and on the *Parameters* tab set the *Optimisation* option to *Local* with the following checkboxes checked:



Single-step the model one day and you should find that DC1 suffers a failure.

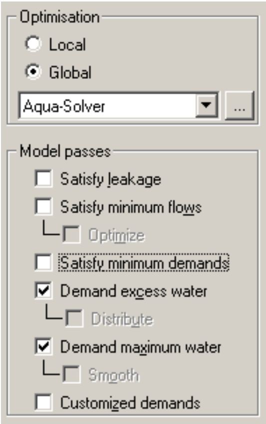


This is caused by Pass 4 (use excess water in order of cost) sharing the 15 Ml available from GW1 between DC1 and DC2. This leaves a shortfall of 2.5 Ml at both DC1 and DC2, which only DC2 can get from GW2.

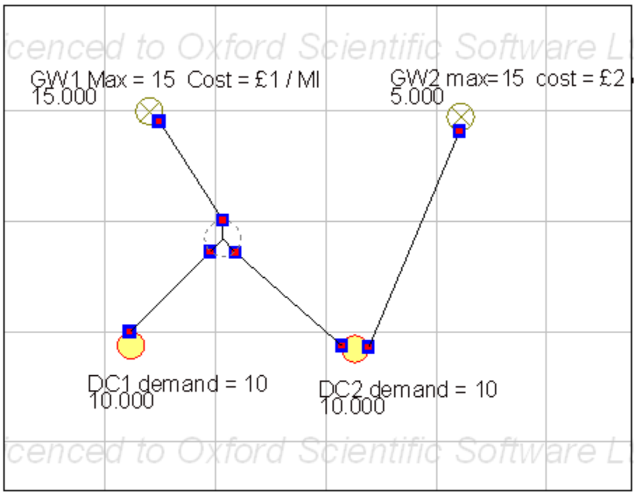
In this simple tutorial it would be straightforward to use any one of a number of ways of fixing this problem: alter cost weighting; alter the Advance Order of each DC; alter route priority; use VBA; etc.

In a more complex model it can be time consuming to find and fix a failure to meet demand when water is apparently available. The *Aqua Solver* global optimiser will do this automatically.

Now, in the model setup dialog set the *Optimisation* option to *Global*:



Now demand is met. The global analysis performed by *Aqua Solver* still preferentially uses the cheaper water from GW1 but ensures that DC1 does not fail.



Where To Go Next

Tutorial 2 uses one of the most complicated components: the **Reservoir** in the simplest of ways. However more complex uses of this and other components are achieved using the same techniques practised in both tutorials.

Now that you know how to manipulate components within a project, the best way of exploring further is to build your own water resource model.

To help you in this process, Aquator provides a comprehensive *On-Line Help* which is context–sensitive and can be accessed interactively while building and testing your model.

The *Aquator Reference* section can be used to find out further details on the techniques practised in the tutorials.

The *How It All Works* section explains the underlying methods Aquator uses to run the model.

The *How to Use VBA with Aquator* section explains how to customize the operating rules of any component using *Microsoft Visual Basic for Applications*.

The Aquator installation provides a number of examples for different components which are listed in *Further Aquator Tutorials*.

Finally, a section on *Troubleshooting and FAQ* is provided to resolve problems when Aquator does not appear to behave in the expected way.

Further Aquator Tutorials

The installation includes tutorial projects in the database: *AquatorTutorials.mdb*. A short explanation for each project can be found in the *Description* box of the *Project Node*.

Test #1 River reach losses + prescribed flow

Test #2 Sharing

Test #3 Reservoir operation

Test #4 Maintained flow and river regulation

Test #5 Annual licence

Test #6 Control flow

Test #7 Minimum demands

Test #8 Bidirectional flow

Test #9 Reach losses with time delays

Test #10 Diversion

Test #11. Multiple abstractions

Test #12 QueryRelease + GS maintained flows

Test #13 Abstraction releases

Test #14 Pump station supporting river

Test #15 Discharge + Hydrogenerator + WTW

Test #16 Blender

Test #17 Abstraction groups

Test #18 Reservoir groups

Test #19 New licences for V2

Test #20 Forecasting

Test #21 Multiple abstractions with time delay

Test #22 Max flows

Test #23 No other sources allocator

Test #24 Circular routes

Test #25 New licences for V2 + min demands

Text #26 Bulk Supply

Test #27 Hunting

Test #28 East Midlands GW Hunting

Test #29 Leakage/min flow reservations

Test #30 Optimize minimum flows

Test #31 Route priority

Test #32 Min flow u/s of regulator

How It All Works

In this section we give a description of how Aquator works for the default case where the *Aqua Solver* global optimiser is enabled. See *the Appendix* for a similar description of the legacy case where the *Aqua Solver* global optimiser is disabled.

Algorithms, rules and optimisation

At the highest level, and ignoring a lot of detail to be described later, an Aquator water resource system model is designed to:

- Meet demand
- Enforce all rules, constraints and licences
- Minimise cost of operation when water is plentiful
- Minimise impact on resources when water is scarce

Aquator is the only software package of its type to incorporate Microsoft® Visual Basic for Applications® (VBA) which the modeller can use to craft bespoke algorithms. VBA is the software industry's premier solution for end-user customisation, providing:

- A visually rich development environment with syntax colour coding, automatic code generation, *Microsoft intellisense* code writing assistance, and hundreds of pages of online help
- Complete compatibility with VBA in Microsoft Office
- Wide range of courses offered by many training companies
- Large pool of external consultants and companies available that provide programming development services using VBA

There are a very few restrictions on the algorithms that can modelled with VBA when the *Aqua Solver* global optimiser is enabled. See the section *Aqua Solver* for a detailed description and a summary of these restrictions.

Resource state

In order to provide a description of how model calculations are performed we need to define the concept of *Resource State.*

Every supply has a resource state which is a numerical value that can vary from zero to infinity. This value is automatically calculated at the start of each days calculations.

A Resource State of 1.0 means that the resource (supply) is in its ‘normal’ state for the current day. Some examples are

- A reservoir control curve defines the daily storage corresponding to a resource state of 1.0 – the resource state of the reservoir is the ratio of actual storage to the control curve
- The ratio of the maximum water available to the so-called *Control Flow* of a groundwater determines the resource state of the groundwater component
- Licences affect resource state. The resource state of a licence is the amount left divided by the amount that would be left if the licence was being used evenly throughout the year
- The resource state of a supply with one or more licences is the minimum of the resource states of the supply itself and all its licences.

Model Calculations

Aquator models a water resource system by combining a daily multi-pass calculation of how water is to be distributed together with the operating rules built into each component in the model.

Aquator calculates water movements on a daily basis but sub-daily calculations are done by some components, for example for minimising pumping costs by taking advantage of different electricity tariffs in each 24 hour period.

The *Aqua Solver* global optimiser built into Aquator provides a mathematically optimal solution. At the heart of this algorithm is a classic linear programming algorithm which has the advantage of a known solution technique which is numerically stable and can be completed in a finite number of steps.

However the problem to be solved is not a single linear programming problem. The Aquator philosophy, based on the real-world operation of UK water companies, is to

- Minimise cost of operation for those supplies whose *Resource State* is greater than one.
- Minimise the impact on *Resource State* from those supplies whose *Resource State* is less than one
- Perform both these optimisations in a single day, as required e.g. the *Resource State* of some supplies may be greater than one, others less that one, or the *Resource State* of some supplies may transition from greater than one to less than one in a single day as the result of water movements
- When minimising cost ensure that licences will not run out later in the model run

- When minimising the impact on resources ensure demand is met (if at all possible) even if licences might run out later in the model run.

The *Aqua Solver* global optimiser joins together mathematically optimal solutions that meet these goals and constraints simultaneously.

Overview of the Model Run

During each day of a model run water is moved according to the input data, the rules built into each component, and the connectivity (the *network*) of the model. There are five phases to such water movements.

1. Any catchments in the model add to river flows at the start of the day. Such flows may be controlled by a simple time series of values or possibly by a full catchment model running at the same time.
2. Any river regulators in the system may then augment river flows in order to satisfy river flow constraints and today's expected abstractions. If the forecasting option is turned on then releases may also be made to satisfy flow constraints and predicted abstractions on future days.
3. Demand centres then try to satisfy their demands by drawing water from any or all available supplies, such as river abstractions, groundwater abstractions, reservoirs, etc. Demand is typically specified by a profile of daily demand values repeated each year, but can be defined in other ways.
4. Reservoirs refill as necessary from their available supplies according to built-in rules governing refilling
5. At the end of the day any reservoir which has had excess water pushed into it will spill into its attached river spillway, possibly leading to further spills from any other downstream connected reservoirs.

In phases 3 and 4 water is *pulled* from supplies to demands through what is called the *supply* part of the network. These movements of water are termed *supply-type*. Components lying between the demand and the supply can control, disallow, decrease, or even increase, the amount requested by the demand.

In the other phases water is *pushed* into the *river* part of the network. These movements of water are termed *flow-type*. Downstream components have no control once water has been pushed into the river.

The *river* and *supply* paths in the network may partially overlap. For example demands for water in phases 2 or 3 may be routed upstream along a river, entering at an abstraction and exiting at a regulator, finally terminating at a supply connected to the regulator. In other words Aquator allows the use of river reaches to transport water to satisfy demands.

This very high level overview omits much detail. The next sections give more comprehensive descriptions. First we describe how the model attempts to satisfy demand (phases 3 and 4). Then we describe river regulation (phase 2). This is followed by a discussion of the finer detail omitted for clarity in the previous sections.

Demanding water

There are two types of component that demand water in phases 3 and 4. Demand Centres represent the demands that drive the system e.g. towns, industry, etc. These demands are made during phase 3 and will typically reduce reservoir storage levels. Reservoirs act as demands in phase 4 when they attempt to refill. Because Demand Centres and Reservoirs act as demands in near-identical fashion the description of this section applies to both except where noted.

When a model run is initiated, Aquator determines the connectivity (the network) of the components that make up the system. Demands for water will be passed up the network from component to component, starting with a demand-type component and terminating on a supply-type component. During this initiation phase the order in which these demands will be made each day is determined.

When the *Aqua Solver* global optimiser is enabled demand-type components are ordered by the **Advance Order.** This is usually determined automatically by Aquator and is so named because the model calculations are advanced in this order. The first components to demand water are Demand Centres and these all have an advance order of 1 by default. Reservoirs immediately up-stream of demand centres will have an advance order of 2. Reservoirs upstream of reservoirs with an advance order of **N** will have an advance order of **N+1**. Components that do not demand water have an advance order of 0 (zero).

In a multiply-connected network it is possible to get a situation where for example reservoir **A** is upstream of both reservoir **B**, whose advance order is **M**, and also of reservoir **C**, whose advance order is **N**. In such a case reservoir **A** is assigned an advance order which is the larger of **M+1** and **N+1**.

If the *Aqua Solver* global optimiser were disabled then within each **Advance Order**, demand-type components can be ordered by **Demand Order**. But when the global optimiser is enabled Demand Order has no effect.

The advance order of individual components can be manually set if desired by ticking the "Allow Advance Order (AO) to be set manually" check box on the *Model Setup* dialog. But with *Aqua Solver* enabled it is strongly recommended to allow Advance Order to be determined automatically.

Each day all the calculations for one advance order are completed before the calculations for the next advance order are started. Each day water is moved in response to demands as follows:

- Each advance order is taken in turn, starting with advance order 1 (demand centres), finishing when all demand-type components have been done
- For each advance order: conceptually take water in six **passes,** described next
- For each pass: each demand-type component in this advance order first reserves and then takes water, potentially using each available **route**

The term **Pass** is used to denote water moved to satisfy a particular requirement. The six passes currently supported by Aquator are:

- **Pass 1**. Leakage. Water is taken to satisfy any leakage along each route. On this pass demand-type components actually request a zero quantity water but this request is increased to a non-zero value by any component on the route that has a non-zero leakage that has not yet been satisfied.
- **Pass 2**. Route minimum. Water is taken to satisfy any minimum flow requirement along the route. The minimum may not be satisfied e.g. if the maximum flow has been set lower than the minimum. When the *Aqua Solver* global optimiser is enabled this pass is actually merged with passes 4 and 5.
- **Pass 3**. Minimum demand. Demand centres can optionally have more than one input and each input can optionally specify that a certain part of the total demand can only be met by water entering on the specified input. This models situations where water cannot be freely transported across a city and pass 3 attempts to satisfy these requirements. This pass does not apply to reservoirs. When the *Aqua Solver* global optimiser is enabled this pass is actually merged with passes 4 and 5.
- **Pass 4**. Excess demand. Ideally the bulk of the required water is moved in this pass. Supply type components are allowed to release water that is in excess of their normal operating rules and constraints. Excess water is water that can be distributed while staying within normal limits. Supplies used on pass 4 have a *Resource State* greater

than one. The *Aqua Solver* global optimiser minimises cost of operation on this pass.

- **Pass 5**. Maximum demand. If the previous passes fail to satisfy demand then on this pass, demand-type components request supply-type components to release additional water, up to the maximum available. This is typically more than their excess water i.e. it may break normal operating rules but it will still obey safety and any other overriding rule. On this pass supplies with a *Resource State* less than one may be used. The *Aqua Solver* global optimiser minimises the impact on resources on this pass.
- **Pass 6**. User Defined. On this pass water is taken as determined by a user-defined algorithm coded using *VBA*. with the *Aqua Solver* global optimiser enabled this pass should never be used. It is left as an available option for special cases.

As described earlier and repeated here: the amount of excess water that a source has available for Pass 4 depends on its **Resource State**. Sources and constraints report their Resource State to Aquator at the beginning of the day as a number that must be zero or greater. A Resource State of 1.0 means that the source is in its normal state for the current day. A Resource State of less than or equal to 1.0 indicates that there is no excess water available from this source for pass 4. A resource state of greater than 1.0 indicates that there is excess water available for pass 4.

The allocation of resources to meet demand is therefore driven by Aquator in a way that tends to equalise Resource State of all the sources. It is only when water is plentiful (i.e. Resource State>1.0) that cheapest water is used first.

Optimisation and Smoothing

The previous section described the optimisation algorithms employed by Aquator:

- Minimise cost of operation when water is plentiful
- Otherwise minimise impact on resources

and the multi-pass calculations performed each day:

- Pass 1. Satisfy leakage
- Pass 2. Meet minimum flows
- Pass 3. Meet minimum demands
- Pass 4. Meet demand while minimising cost
- Pass 5. Meet any remaining demand while minimising impact on resources

Pass 1 is done independently and then the *Aqua Solver* global optimiser provides an optimal solution to passes 2-5. Pass 4 minimises cost and pass 5 maximises resource state. Passes 2 and 3 provide constraints that are merged into passes 4 and 5. The result is that the optimiser computes the precise amounts of water to be moved on passes 4 and 5 in one solution.

The solution can be further constrained to behave smoothly i.e. not to fluctuate too much day-to-day. See section *Aqua Solver* for a description and how smoothing is enabled.

River Regulation

In phase 2 of the calculations water may be released from Regulator components into rivers in order to satisfy downstream flow constraints and/or downstream abstractions that will take place later, in phases 3 and 4, either on the same day or optionally on subsequent days.

Regulators demand water from the supplies to which they are connected. Therefore much of the previous section which describes how *Aqua Solver* attempts to satisfy demands applies to regulators. For example, six passes are made, exactly as for demand centres and reservoirs.

But regulators also differ significantly in two ways, which is why we describe them separately and why they are allocated a separate phase in the calculations.

Firstly, the order in which regulators make releases, the so-called **Regulation Order**, can be arbitrarily changed, whereas the **Advance Order** of demand centres and reservoirs is usually automatically determined.

Secondly, their demands are determined by *other* components whereas demand centres and reservoirs determine their demands internally.

The simplest way to set the **Regulation Order** is to open (Setup...Model) the *Model Setup Dialog* and click the up and down arrows inside the *Regulation and Demand Order* box on the *Parameters* tab. Note the abbreviations RO for Regulation Order and DO for Demand Order on this dialog. Some experimentation may be needed to understand how one pair of buttons control two parameters.

Regulators with different **Regulation Order** release water completely independently, at different times, lowest regulation order (1) first. Regulators with the same **Regulation Order** all work at the same time. When the *Aqua Solver* global optimiser is enabled the **Demand Order** parameter of each regulator has no effect.

Regulators query downstream components in order to determine how much water to release. A component such as a Gauging

Station which has a flow constraint i.e. a constraint that dictates the minimum flow in the river, may request a release to augment the river flow. An Abstraction may also request a release in order to support any expected demands later in the day.

If forecasting is turned off then regulators only query downstream through river reaches with zero time delay. If forecasting is turned on this restriction does not apply and releases may be made in order to satisfy flow constraints and/or abstractions on days after the day on which the release was made.

Route ordering

When the *Aqua Solver* global optimiser is enabled the numerical optimisation algorithm completely determines the optimal amounts of water to be moved along each route from supply to demand.

Demand Saving

In times of water shortage Aquator can optionally implement a multi-layered demand saving algorithm. These calculations are controlled by several model sequences and model parameters.

Here we provide a step-by-step explanation of how demand saving is calculated.

1. First we distinguish between demand saving level and demand saving factor. Level can go from 0 (no saving) to 5 (max saving) while factor can be 0% (no saving) to 100%. The original idea was to first determine level, then translate to factor. This gives the freedom to decouple the algorithm that determines level from the actual percentage. Demand saving can be applied globally, as a single number across the whole model, or through demand saving group components that allow different groups of demands to have different demand saving factors at any one time.
2. Under *Demand Saving Source* there are three options on the Sequences tab of the model setup and demand saving group setup dialogs. The third option (*Sequence factor*) is the simplest while the first option (*Component level*) is the most complex, so taking them in reverse order:
3. The project or demand saving group has one sequence named *Demand saving factor*. If the *Sequence factor* option (third option) is chosen and a time series is linked to this sequence then the demand saving factor for every day is taken from this time series, unless the value for the day happens to be missing. No other calculations are done i.e. all other demand saving factors and levels (sequences and parameters) are ignored. But if this

sequence is not linked to a time series, or if the value is missing, then the demand saving is simply set to zero percent.

4. The project or demand saving group has a *Demand saving level* sequence. If the Sequence level option (second option) is chosen and a time series is linked to this sequence then the level for the day is taken from this series. If there is no time series, or the value for the day is missing, then the demand saving level is set to zero, which implies a factor of zero percent.
5. If the *Component level* option (first option) is chosen then the demand saving level is the maximum demand saving level returned by any component in the model or demand saving group.This value is returned in the call to InitializeDay for each component (this call takes place before any water has been moved in the model for that day). In practice only reservoirs set a non-zero demand saving level at present.
6. To prevent any possibility of the demand saving level changing every day there is a *demand saving hold* option on the *States* tab. If enabled then after a demand saving change, the demand saving level cannot change again until the specified number of days have passed. Additionally the demand saving level is constrained to change by plus 1 or minus one only.
7. To translate level to factor, the appropriate project or demand saving group sequence *Demand saving factor - level N* (*N*=1 to 5) is examined. If the sequence is linked to a profile of factors then the value from the profile is used. If not, or if the value was marked as missing (unlikely for a profile), then the value of the project parameter *Demand saving.Level N* is used instead. In the Model dialog the relevant parameters can be seen on the *Parameters* tab, the sequences on the *Sequences* tab.
8. Finally, Aquator passes the now-calculated demand saving factor to every component in the model or demand saving group, as appropriate in the call to *StartDay*.

Demand saving hold

To constrain the demand saving level from changing too rapidly demand saving hold may be enforced. Essentially this puts a value greater than one on the minimum number of days that must elapse after a requested change in demand saving level before the demand saving level that is enforced is changed.

This is controlled by the *Demand saving hold (DSH).Method* state value that can take one of three enumerated values.

None No demand saving hold and the demand saving level may change as frequently as every day.

Simple period A single period is specified in days, which applies to all levels and to both increases and decreases in demand saving. The period is specified by the *Simple DSH.Delay/hold period* state value and the number of days so far at the current demand saving level by the *Simple DSH.Day in period* state value.

Multi-level delays Before a higher demand saving level is reached that level must have been requested for at least *D* days, where *D* is given by one of five state values *Multi-level DSH: 1.Delay* to *Multi-level DSH: 5.Delay*, one for each level. Similarly the demand saving level cannot decrease until at least *H* days that this level have passed, where *H* is given by one of five state values *Multi-level DSH: 1.Hold* to *Multi-level DSH: 5.Hold*, one for each level. Finally, the five state values *Multi-level DSH: 1.Days* to *Multi-level DSH: 5.Days*, one for each level, record how many days have passed at each level respectively.

Forecasting

The forecasting facility in Aquator enables the water moved on any one day to allow for river flows and abstractions on *future* days and not just the current day. In particular it enables Regulators (and Pump Stations acting as regulators) to add water to rivers to meet anticipated flow constraints and demands in the future.

Forecasting is done on a best-effort basis, it is not possible for the *Aqua Solver* global optimiser to guarantee a solution for the whole model run because the problem to be solved would become unbounded. The crucial thing to remember is that it is the regulator components that are responsible for attempting to add water to rivers that have time delays in order for water to be available for abstraction on future days.

By default the model parameter *Options.Forecasting on* is set to *False* and no forecasting is performed. This means each daily time step is executed without regard for the state of the system in the future.

At the time of writing the only (non 3rd party) component with a time delay state is the Reach component, whose time-of-travel may be specified in days. If all reaches are set to have zero time-of-travel then forecasting will have no effect and should be left off.

Consider a Gauging Station with a maintained flow constraint, and a Regulator, on the same river but separated by one or more reaches with a total time-of-travel of T days where T > 0. To meet the maintained flow constraint on any day N, the regulator may need to add water to the river on day N – T.

To turn forecasting on take the following steps:

- set the model parameter *Options.Forecasting on* to *True*
- for a Regulator component, set its *Release.Support river* parameter to *True*; for a Pump Station to act as a regulator set its *Pumping.Amount* enumerated parameter to *Release (RO)* which is shorthand for "make a release in river regulation order"
- on the *Model Setup Dialog* (menu command *Setup...Model...*) set the order in which these regulator components are to add water to the river by using the buttons in the *Regulation and Demand Order* box on the *Parameters* tab
- for both Regulators and Pump Stations acting as regulators set the parameter *Release.Maximum delay* to the number of days ahead these components are to make forecasts
- also for both, set the parameter *Release.Maximum components* to an integer not smaller than the total number of downstream abstractions and gauging stations which are to be inquired as to whether they have forecast requirements for water; include abstractions and gauging stations that will respond negatively i.e. this parameter simply limits how many downstream components will be examined; setting this parameter and the one described in the previous bullet point to the smallest valid values may greatly improve execution speed
- for downstream Abstraction and Gauging Station components that are to participate in forecasting, set the parameters *Upstream releases.Minimum Delay* and *Upstream releases.Maximum delay* to values such that the time-of-travel from upstream regulators to these components lies between these limits; this enables you to fine-tune which regulators support which abstractions and gauging station flow constraints
- for downstream Abstractions whose predicted demands are to be supported by forecast releases by upstream regulators, set the remaining parameters in the *Upstream releases* group appropriately; the *Prediction method* enumerated parameter allows a choice of how to predict the demand in the future; you can use the *Prediction amount* parameter value, yesterdays abstraction, or yesterdays demand; finally the *Prediction scale factor* parameter is used to scale any prediction e.g. to allow for uncertainties, losses, etc

- both Regulator components and Pump Station components acting as regulators also have a parameter *Release.Scale factor* which can be used to scale up the sum of the releases requested by downstream components

This completes the actions needed to add forecasting to river releases. But this leaves open whether or not abstractions are to take forecasts into account.

Consider an *Abstraction* component and a *Gauging Station* component separated by reaches with a total time-of-travel of T days where T > 0. if the *Gauging Station* has a flow constraint then the *Abstraction* may optionally look at the forecast river flows T days ahead to decide if abstraction today will break a flow constraint in T days time.

There are two parameters on each *Abstraction* component which control this:

- the *Check downstream.Maximum flow constraints* limits how many downstream flow constraints will be taken into account; this can be used whether or not forecasting is on, and setting to the smallest valid value can greatly improve execution speed

the *Check downstream.Maximum delay* only applies when forecasting is on; downstream flow constraints whose time-of-travel from the abstraction is greater than this value are ignored when the abstraction is deciding how much water it may supply; again, setting to the smallest valid value can greatly improve execution speed.

Scenarios and Sets

Multiple data sets

A common modelling requirement is to run the same model with different input data, possibly capturing different output results. Aquator provides extensive facilities for storing multiple data and result sets with each project and an easy way to select amongst those available.

Project Sets

A project set comprises a set of data for the entire model and a name by which the set is identified. Naturally you can have parameter sets and state sets i.e. sets which supply, respectively, every parameter on every component with a value, and every state on every component with a value.

You can also have sequence sets. These provide links to time series and profiles for every sequence on every component.

You can also have variable sets. A set of variables comprises all the information entered for every variable on every component i.e. whether to capture the value during a model run, how to plot, etc.. Typically one variable set would be designed to plot certain variables on certain charts at the end of a model run, another set might plot completely different variables on the same charts or the same variables on different charts, or any combination that is useful.

Project scenarios

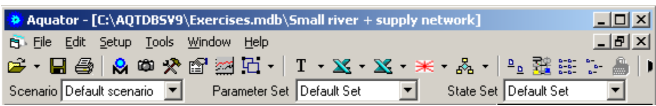
Having to individually select a particular parameter set, state set, sequence set, and variable set is still slightly onerous and increases the chance of selecting an unsuitable combination. Project scenarios overcome these problems.

A project scenario is a named collection of sets, with at least one set of each type. For each set type, one is considered **active** in each scenario. Selecting a scenario therefore implicitly selects a parameter set, a state set, a sequence set, and a variable set.

Within one scenario, if there is more than one parameter set to that scenario for example, then another parameter set can be made active.

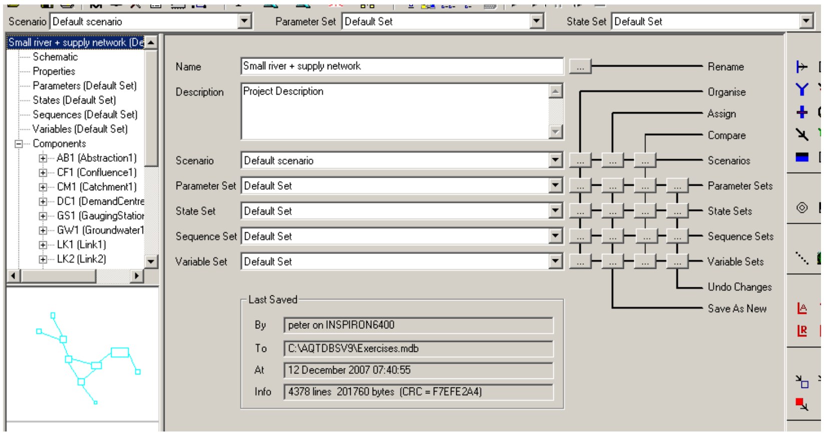
Thus scenarios link to sets but sets can exist independently of scenarios.

Selecting scenarios and sets



An optional additional toolbar, below the main Aquator toolbar, provides a way of selecting any scenario or set at any time. To set this toolbar up, or to hide it completely, go to the *Scenario toolbar* tab of the *Options* dialog (*Setup...Options...* menu item or double-click in an empty space on the toolbar itself).

Alternatively the topmost project node provides full control over scenarios and sets, as do the various menu items off the *Setup...Scenarios* and *Setup...Sets* menu commands.



The topmost project node in the project tree view displays dropdown lists of the available scenarios and sets in the information pane. Simply selecting an item in one of these lists makes that set or scenario active.

If a new scenario is selected then this implicitly selects a new set of each type. With one scenario selected, only those sets assigned to that scenario can be selected.

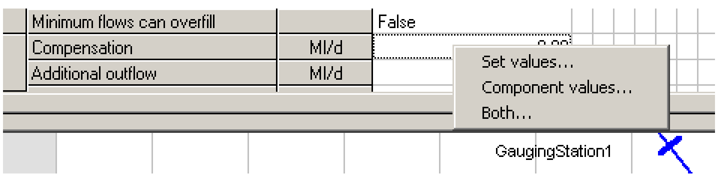
Altering data in a set

Unless you take special steps, to be described, any editing applies to the currently selected set, and all data in other sets is effectively hidden. Thus the normal way of working is to select a set and then alter the data in that set.

All the description in the tutorials, for example, of how to edit parameters, implicitly assumed that the required parameter set had first been selected. A new project implicitly creates a single parameter set, state set, etc. which are pre-selected (and until new sets are created, cannot be de-selected).

Data in another set can be changed, without selecting the other set, using the *Compare Sets Dialog* (click on the appropriate button in the above screen shot, or use the *Setup...Sets ..* menu commands). This allows data to be copied from one set to another using the small buttons marked with the '<' and '>' symbols.

In addition most edit dialogs accept a right-click as well as the usual left-click. The latter always initiates the process of editing the value in the selected cell i.e. the value in the current set.



Choosing one of the items from the right-click popup menu shows a dialog that allows editing of values across multiple sets and/or components. See the *Edit Set/Component Values* dialog.

Creating new sets and scenarios

New sets are always made by cloning existing sets and similarly new scenarios are made by making a copy of an existing scenario. It is advantageous therefore to set up data common to more than one set before making the other set(s), although the *Compare Sets Dialog* does allow data to be copied from one set to another and even provides a one-click way of synchronizing two sets.

Project sets and component sets

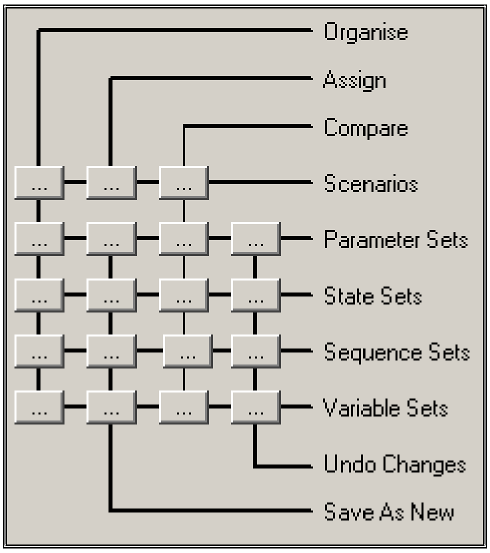
In earlier versions of Aquator it was possible to have some components with multiple sets of parameter values while other components only had a single set. Although this reduced memory usage it was complicated to set up.

In this release a simpler way of working has been implemented. When a parameter set is cloned a copy of every parameter is made, for the project and for every component in the project. The concept of individual component sets is no longer used and instead we just talk about 'project sets', or even more simply, 'sets'.

Opening an old project with this release of Aquator will silently and seemlessly convert the old way of working into the new.

Organising scenarios

The first row of three buttons in the topmost (project) node in the project treeview provide facilities for organising, assigning, and comparing scenarios.



Alternatively use the *Setup...Scenarios...* menu commands.

For organising (creating, deleting, and renaming) scenarios, see *Scenarios Dialog*.

For assigning sets to scenarios see *Scenario Sets Dialog.*

For comparing scenarios see *Compare Scenarios Dialog.*

Organising sets

The remaining four rows of buttons in the topmost (project) node in the project treeview provide facilities for organising and manipulatng sets. The rows correspond, respectively, to parameter, state, sequence, and scenario sets. Alternatively use the *Setup...Sets...* menu commands.

For organising (creating, deleting, and renaming) sets (first column of buttons) see the *Projects Sets Dialog*.

The second column of buttons (Save As New) allow the current set to be saved as a new set while optionally restoring the current set to its original state. Thus a new set with different values can be created by editing and then creating rather than creating and then editing. See *Create New Set Dialog*.

The third column of buttons is used to compare sets. See *Compare Sets Dialog*

The fourth column of buttons is used to undo changs to the current set.

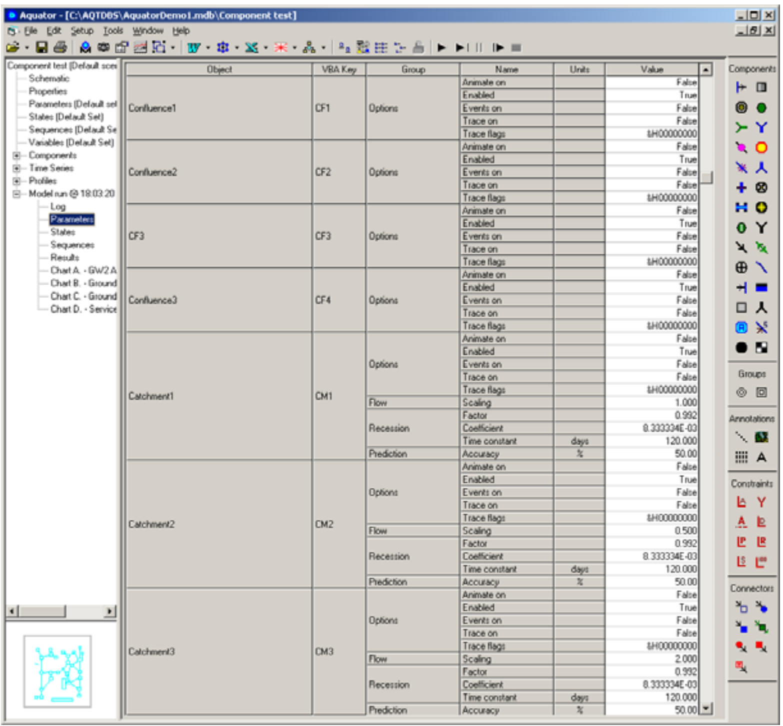
Advanced use of sets

When a model executes the parameter values, state values, and sequence links that comprise the input data to the model run can optionally be captured and displayed as nodes under the model run node.

If the model run is stored in the database (either right-click the model run node and select *Store to database* from the popup menu, or use the *File...Model Runs....Store To....Database* command) then all the parameter values, state values, and sequence links used in the model run are stored as well.

This model run and its associated data can be later retrieved from the database in another Aquator session through the menu command *File...Mode Runs...Retrieve From...Database*. The nodes of a model run that has been saved are shown in a different colour.

In this way parameter sets, state sets, and sequence sets are effectively stored independently in the database in a non-editable form. Aquator allows you to compare these values and links with set values and links, and even to recall the run data into a set.



In the above screen shot, right-clicking on the *Parameters* node (or *States* node or *Sequences* node) and selecting *Compare* you can compare the parameter values (or state values or sequence links) with those of any parameter set (or state set or sequence set).

The comparison is shown using the same dialog as is used to compare one set with another. This dialog also allows you to copy one, some, or all values (or links) from the model run into the selected set. See *Compare Sets Dialog*.

Exporting and importing data

Aquator provides facilities for exporting data to and importing data from

- Excel spreadsheets
- WISKI format files
- HYSIM format files
- WRA (Water Resource Associates) format files
- The clipboard can also be used to import data.

Exporting data

Use one of the following actions to begin exporting data

1. *Tools...Export* menu item
2. Right-click on a time series or profile node, or the parent node of any of these (e.g. Flow), or the grand-parent node (i.e. *Time series* or *Profiles*)
3. Click the down-pointing arrow immediately adjacent to the *Export* toolbar button
4. Click on the *Export* toolbar button itself.

In cases 1-3 a sub-menu appears where you choose one of

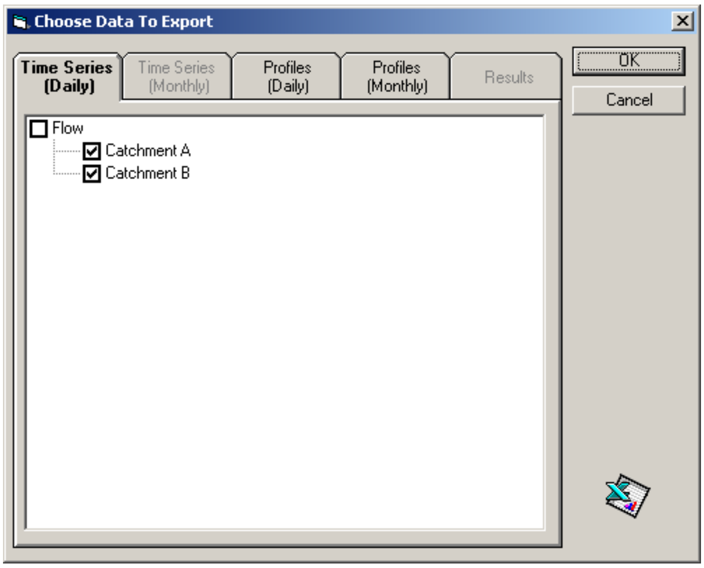
- Wizard
- Excel spreadsheet
- HYSIM format file
- WISKI format file
- WRA format file

In case 4 the last-used export tool, as shown by the button icon, is used. Because the icon changes it may be helpful at first to hover the mouse pointer over the toolbar buttons to locate the *Export* button.

If you choose *Wizard* then the *Export Wizard* dialog, described elsewhere, will appear that steps through a series of questions to assist the novice with exporting data.

Otherwise, if the source of data is unambiguous e.g. the currently displayed node shows data for a particular time series or profile, then the data is immediately exported into Excel or Notepad, according to the choice made, and in the chosen format.

The data source might be ambiguous for two reasons. either the current node is not a time series or profile node; or checkboxes are on to allow multiple node selections. In this case a dialog appears where the data set(s) to be exported are chosen.



Checking more than one item and clicking OK will then export multiple data sets into Excel or Notepad, according to the choice made, and in the chosen format.

Importing data from a file

Use one of the following actions to begin importing data from a file

1. *Tools...Import* menu item
2. Right-click on a time series or profile node c, or the parent node of any of these (e.g. *Flow*), or the grand-parent node (i.e. *Time series* or *Profiles*)
3. Click the down-pointing arrow immediately adjacent to the *Import* toolbar button
4. Click on the *Import* toolbar button itself.

In cases 5-7 a sub-menu appears where you choose one of

- Wizard
- Excel spreadsheet
- HYSIM format file
- WISKI format file
- WRA format file

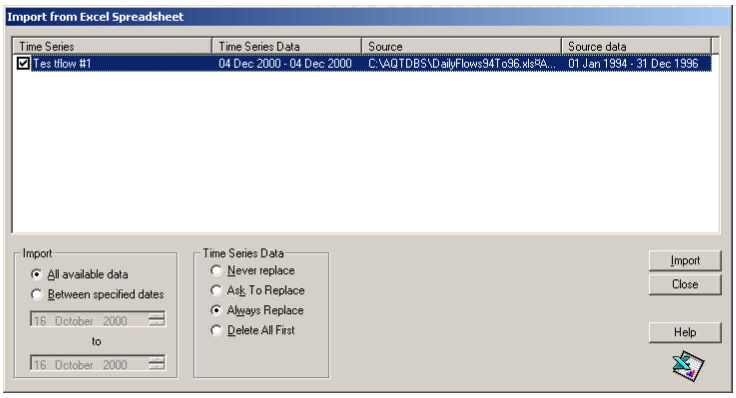
In case 8 the last-used import tool, as shown by the button icon, is used. Because the icon changes it may be helpful at first to hover the mouse pointer over the toolbar buttons to locate the *Import* button.

If you choose *Wizard* then the *Import Wizard* dialog, described elsewhere, will appear that steps through a series of questions to assist the novice with importing data.

Otherwise, if the import target is unambiguous e.g. the currently displayed node shows data for a particular time series or profile, then a dialog apears where you choose the file where the data is stored.

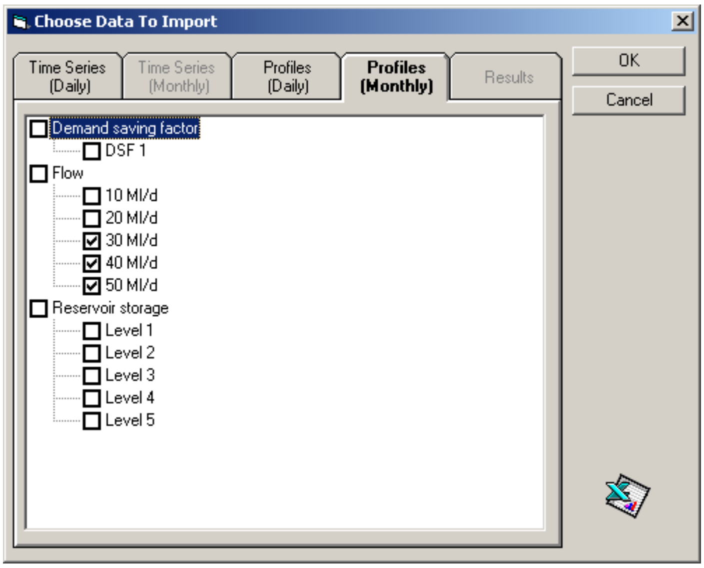
For each import tool supported - Excel, HYSIM, WISKI, WRA - a specific dialog will appear where you specify which data set in the file you want to import. Since these dialogs vary with the format, and more can be added, they are not shown here.

The final step is to confirm the import process on the following dialog where you can optionally choose the range of dates to import, and also whether to allow existing data to be over-written (time series only - profile data is always completely over-written).



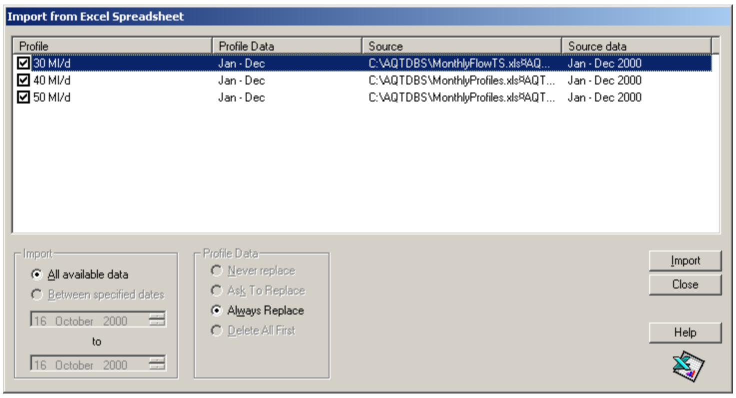
But the target of the import might be ambiguous for two reasons. either the current node is not a time series or profile node; or checkboxes are on to allow multiple node selections. In this case a dialog appears where you must specify if you are importing into an existing time series or profile, a new time series, or a new profile.

Choosing *Import into existing time series or profiles* then displays a dialog where you choose which time series or profiles.



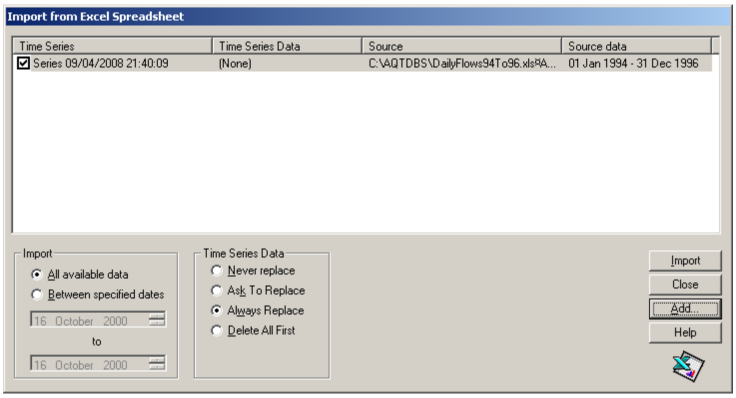
For each import tool supported - Excel, HYSIM, WISKI, WRA - a specific dialog will then appear, once for each selected time series or profile, where you specify which data set in the file you want to import. Since these dialogs vary with the format, and more can be added, they are not shown here.

The final step is to confirm the import process on the following dialog where you can optionally choose the range of dates to import, and also whether to allow existing data to be over-written (time series only - profile data is always completely over-written).



Choosing *Create and import into new time series* or *Create and import into new profile* is similar to the previous procedure except that the dialog where you choose which time series or profile(s) to import into is not shown. Instead the custom dialog for each type of import tool allows you to specify the name and data type of the new time series or profile.

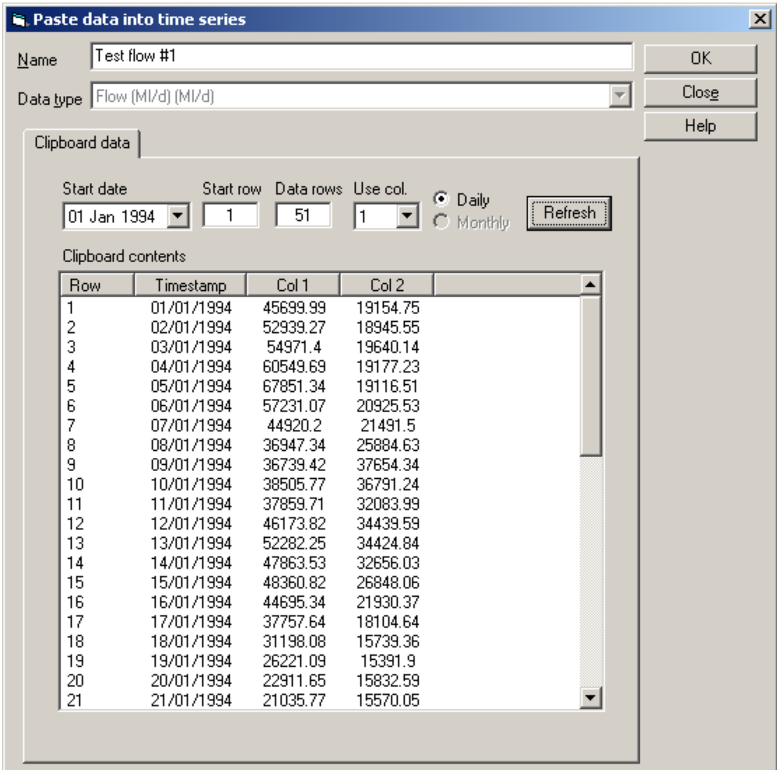
Then the familiar dialog where you confirm the import appears.



The only difference is that the *Add* button allows you to add additional new time series or profiles and import them in one go.

Importing data from the clipboard

When the currently selected node is displaying time series or profile data then clicking *Paste...* displays the following dialog



In this example three columns in Excel - one containing dates and two containing numeric data - have been copied into the clipboard. Clicking *Refresh* will re-scan the clipboard. Each time the search is for rows of data terminated by CR-LF, with commaor tab-separated data items on each row. The first column can optionally specify date, otherwise the *Start date* control in the dialog must specify the start date and the *Daily* or *Monthly* option buttons must specify the interval.

If more than one column of numeric data is present then specify which column to use. Finally click OK to import the clipboard data.

Right-clicking on the *Time series* or *Profile* node, or any node under these, and choosing *Add*, enables using the clipboard to create a new time series or profile in a similar manner.

Printing from Aquator

How to print

The following describes the many settings in Aquator which can be used to control printed output, and gives some hints and tips that may be useful.

The general procedure is

1. set up the printer
2. set up the print properties
3. choose which page(s) to print
4. preview how the pages will look
5. print

Set up the printer

Use the *File...Print Setup* command to select and configure a printer. Selecting a printer here changes the default printer used by Aquator. Clicking the Printer toolbar button always prints to the last-selected printer.

Set up the print properties

Several entries in the topmost Properties node of the project tree view can affect printing. Most layout problems can be solved by changing one or more of these settings. See *Print properties* below for more information.

Choose which page(s) to print

By default just the current page is printed i.e. the current node selected in the project tree view. But if check boxes are visible in the project tree view (*Options...Show checkboxes* on the project tree view) then those pages that correspond to checked nodes are printed. If none are checked then the current page is printed.

Preview how the pages will look

Use the *File...Print Preview* command to preview how the printed page will appear.

Print

Either click the Print toolbar button or use the *File...Print* command.

Print properties

Several entries in the topmost Properties node of the project tree view can affect printing. Most layout problems can be solved by changing one or more of these settings.

Note that most problems and most settings apply only when printing the schematic.

Print.Preview scale

This only affects print preview. The window where printed output is previewed will be reduced to the specified percentage of the actual paper size. All previewed pages use this setting.

Print.Scale

Normally set to 100%, this setting scales all printed output *except* the schematic which treated separately. Possible uses for this scaling factor are to squeeze a grid onto one page when it would otherwise print across two or more pages.

Schematic.Print orientation

Pages print in the default printer paper orientation *except* the schematic which can be forced to print in portrait or landscape by choosing the appropriate option for this property.

Schematic.Print/Report extent

This option allows you to choose whether to print the visible portion of the schematic or the entire schematic. In the latter case this includes the white space around the components (see *Schematic.Schematic margin* below). As the name implies, this property also determines how much of the schematic is reported by a Report tool e.g. pasted into Word when generating a Word report.

Schematic.Print/Report scaling

This provides three options: print at a specified scale factor (see *Schematic.Print/Report scale*); print to fit page; or print at full scale (100%). If the option to fit to the page is chosen then the *Schematic.Print/Report margin* property also applies.

As the name implies, this property also determines the scaling used when a report is generated. If the option to fit to the page is chosen then the current Aquator printer page setting is used to determine the scaling. This may not be the same printer page setting in the application used to view the report.

Schematic.Print/Report scale

If the schematic is to be printed at a fixed user-specified scale factor (see *Schematic.Print/Report scaling* for alternatives) then this property provides the scale factor to use.

Schematic.Print/Report margin

If the printout or report scaling option (see *Schematic.Print/Report scaling*) is to fit to the page then this property sets the margin which will not be used. The units twips or one-twentieth of a point i.e. 1440 twips = 1 inch

Replicating Components

A technique for rapidly replicating components is provided by context menu *Copy* and *Paste* commands. The technique involves

1. selecting one or more components
2. copying the selected components into a hidden buffer
3. pasting the components from the buffer onto the schematic one or more times

Pasted components are an exact copy, including all properties (apart from the Display name), parameters, states, sequences, variables, sets of each of these if multiple sets exist, and any VBA custom code.

Note that *Cut* and *Paste* (as opposed to *Copy* and *Paste* described here) is not supported.

Selecting components

This is done in the usual way. Clicking on one component selects just that component. Clicking on other components while holding the Control key selects additional components. Alternatively draw a rubber band rectangle by clicking in an empty space on the schematic and, while holding the mouse button down, drag the corner of the rectangle which appears. This selects all components inside the rectangle.

Copy the selected components into the paste buffer

With components selected as described previously, hold the Control key down, right-click on the schematic, and choose *Copy* from the popup context menu which appears. If the Control key is not down then just the one component selected by the right-click, if any, will be copied.

Paste the components onto the schematic

Right-click anywhere on the schematic and choose *Paste* from the popup context menu which appears. By default a ghost image of the components to be pasted will appear. Position this image by moving the mouse and then left-click once the position is satisfactory. A right-click, or pressing the Escape key, will abort the paste operation.

User-defined parameter and state value editing groups

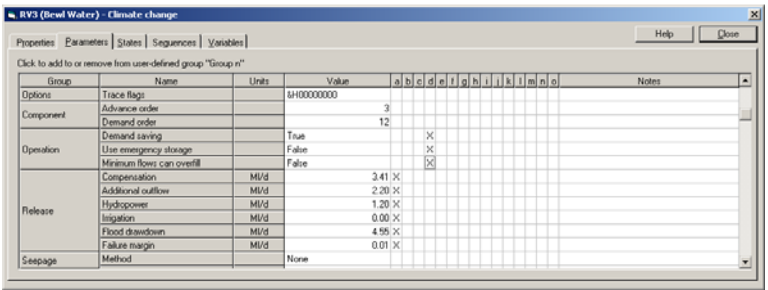
Parameter and state values associated with a component or constraint are normally entered by editing the values shown in the *Parameters* or *States* grid for the component or constraint. This makes it relatively easy to enter all the required data for a model in the right place.

Subsequently you may wish to experiment by changing just one or two parameter values for several components. User-defined editing groups allow you to edit parameter and state values from more than one component or constraint in one place.

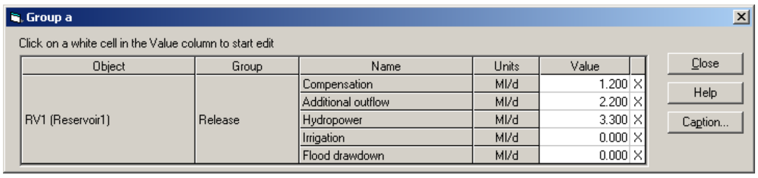
There are 15 user-defined editing groups labelled a to o whose default names are Group a to Group o respectively. These show up as 15 menu bar items (Edit...User defined) and also as 15 button menu items, shown by clicking the down-arrow on the group button.

Clicking on the button itself will redisplay the editing dialog for the group last edited.

Initially these groups are empty. To add parameters or states to the editing group click in the columns headed *a* to *o* on any parameters or states edit grid.



If you now select *Group a* from the *Edit...User-defined* menu, or from the button dropdown list, you can edit all those values placed into the first group.



You can assign your own names to these editing groups either by clicking the *Caption* button on this dialog or editing the 15 project properties *User-defined.Group 1 name* etc.

Fast schematic popup information grids

This section describes the various mouse and key combinations that enable rapid viewing of parameters and states.

These are all accessed by holding the left mouse button down while pressing various key combinations. Once the left mouse button is released any visible finformation grid is hidden.

If no keys are pressed, or if the right mouse button is used, then no popup information grid will be shown. If a grid is shown it will apply to the object under the mouse cursor, or the model itself if not hovering over a component icon.

To view parameter values hold the *Alt* key down while pressing the left mouse button. To view state values hold the *Shift* key down.

If the component selected has constraints then the parameters or states of each constraint may be successively viewed by pressing and releasing the *Ctrl* key down while still holding the other key and the mouse button down.

DC1 Parameters		Value
Group	Name	
Options	Animate on	True
	Enabled	True
	Events on	True
	Diagnostics on	True
	Trace on	False
	Trace flags	&H00000000
Component	Advance order	1
	Demand order	1
General	Demand	292.000
	Demand factor	1.000
	Apply demand saving	True
Fail	If demand not met	True
	Criterion	0.10

If the model is running but paused these popup information grids can still be shown to examine parameter and state values.

In addition, however, when paused, if no key is held down, then variable values are shown. Simply click and hold on any component o examine the component's variables, or the background to examine model variables.

How to use VBA with Aquator

Introduction

Microsoft® Visual Basic® for Applications (VBA) is included with Aquator to enable you to customise the behaviour of your Aquator model and of Aquator itself. These notes should be used in conjunction with:

- The main Aquator documentation and help which describes how the model and its components operate
- The VBA help provided by Microsoft which explains the use of VBA in a general (i.e. in a non Aquator-specific) way (Press **F1** or use the Help menu in VBA)

You use VBA to write your own code, often called custom code in this document. The visual help provided by VBA makes this a relatively easy process to learn and understand.

Your custom code can execute while the model runs and in this way you can add your own operating rules to a model and modify the calculations. Or you can cause your custom code to run at other times, for example to modify parameters or examine the results from a model run.

You can also use VBA from within another application like Excel to load Aquator and run an Aquator model. See *How to use Aquator from another application*.

The use of VBA can vary from the very simple such as a single line of code to change the value of a parameter through to many thousands of lines of code. You have the full power of VBA available and for example as an advanced user, could:

- Completely replace the built-in model calculations with your own.
- Design your own forms to appear when the model executes.
- Start other programs running. For example starting *Excel*, loading a spreadsheet and then transferring data to and from Aquator.

VBA included with Aquator is identical to the VBA included with Microsoft products such as *Word* and *Excel*. If you are familiar with writing macros in *Excel* or *Word* and you have an understanding how Aquator operates, you should find these notes sufficient to get started with Aquator customisation.

If you are unfamiliar with VBA in other products, the online help provided with VBA will be of assistance. There are also many books available on the subject. Also there are over 3 million people out there who are VBA programmers, so advice may not be far away.

If you are unfamiliar with Aquator, it would be a good idea to set up a simple model and run it before starting with VBA. It would also help to have studied the Aquator documentation.

This documentation will concentrate on using VBA to modify model execution as this is most likely to be needed. See *Using VBA to customise the way the model executes.*

Other uses of VBA are considered under the section *Using VBA when the model is not running.*

VBA projects and Aquator models

Each loaded Aquator model automatically creates a VBA project of the same name. You view the Aquator model from within the Aquator main window and you view the corresponding VBA project within the VBA IDE (Integrated Development Environment) window. See *Showing VBA from within Aquator*.

Initially, only the Model object of a new Aquator project exists as an object in the corresponding VBA project, but right-clicking on any component on the Aquator schematic and choosing *Customize...* from the popup menu will add the selected component to the VBA project. Thus only the components which are to have custom code added to them need to be added to the VBA project.

If a component has been added to the VBA project then its constraints, if any, can also be added and customized. Right-click on the component and select *Constraints...* and in the resulting dialog you can customize and uncustomize any constraint i.e. add or remove any constraint from the VBA project.

If custom code is added to a VBA project then saving the Aquator project saves the custom code with the project. In fact the project can be saved from the VBA IDE, without switching back to Aquator, with the minor disadvantage that no progress bar or hourglass cursor appears to indicate the progress of the save operation.

The VBA General project

In addition to VBA projects which corresponds to loaded Aquator projects, a VBA project named **General** will be visible in the VBA

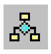
IDE. This project will always be available even if no Aquator projects are loaded.

When Aquator exits the **General** project is automatically saved in a file Aquator.VBA stored in the default projects folder. This means that macros, custom code, custom forms, etc. held in the **General** project are always available, independent of which models and databases are loaded.

The **General** project makes anything stored in it available for use in any loaded project, effectively sharing code and removing the need to add the same custom code to multiple projects. In addition it acts as a place to store custom code such as macros which are to be available even if no Aquator models are loaded.

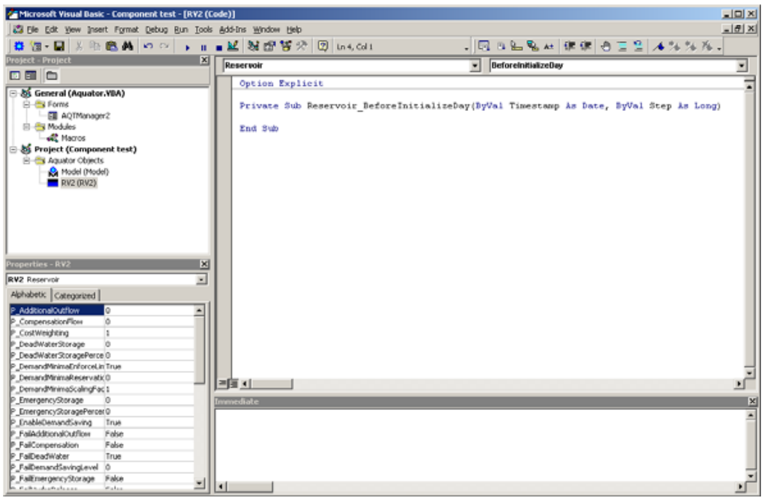
Showing VBA from within Aquator

Show VBA from within Aquator by either:

- Menu item **Tools – Macro – Visual Basic Editor …**
- Shortcut key **[CTRL] F11**
- Clicking on the  toolbar button

Alternatively, if you are using VBA to customise the way the model executes, right click on the component to be customized and select menu item *Customize*. (Note that menu item *Uncustomize* can be used to remove custom code for a component.)

Either action opens the main VBA window, often referred to as the VBA IDE or Integrated Development Environment. It is an independent window and can be closed and reopened as often as desired without any side effects. In fact, it is never really opened and closed, only hidden and shown.



This screen shot shows the appearance of the VBA IDE after one project "Component test" has been opened by Aquator. The child window at upper-left shows a hierarchical view of all VBA projects. Besides the "General" VBA project there is a VBA project that corresponds to the open Aquator project and it has two objects – the Model and reservoir RV1 – which have been customized.

Each customized object has its own child window where its custom code is written. This screen shot shows the child window for reservoir RV1 (the largest child window at upper-right).

At lower-left a so-called Properties child window displays the parameters and states of the currently selected object. At lower-right is a so-called Immediate window where you can type expressions and have them evaluated immediately.

Note the unfortunate duplication of the use of the word "Properties" in the previous paragraph. In VBA this refers to any item belonging to the object whose value you can alter, like a parameter or state value. But Aquator uses "Properties" to refer to those items that are not parameters or states i.e. an Aquator property doesn't affect the numerical calculations.

There are other menu commands under **Tools – Macro**. These correspond to the creation and use of macros discussed in *Using VBA when the model is not running* *(p. 80)*.

If you are executing a model and do not need the VBA IDE open (i.e. you are not debugging your code), model execution times can be reduced by closing the VBA IDE window.

Using VBA to customize the way the model executes

Aquator moves water within a model according to its own set of rules. In some situations these rules may be adequate in themselves to define how the system operates. In other, more complicated cases, Aquator's rules will not be good enough and it will be necessary to 'program in' the desired behaviour. Most situations requiring this programming will be where there are unique or one off relationships between components.

There are three programming topics to help you program Aquator in VBA:

- How do I reference components and their data in my VBA code?
- Advice on writing VBA code
- Where should I place the VBA code?

Referencing components and data is considered below in Referencing objects; Tips are given in Coding advice and placement of code is considered below in Code location.

Referencing objects

You may reference components within the project directly or indirectly.

Direct referencing

In order to make a direct reference, the component must be added to the VBA project by right clicking and selecting menu item **Customize…**. Note the component now appears in the VBA project window and a code window has been created for that object.

Your VBA code can now refer to the object and its parameters etc directly. For example if Demand Centre DC1 has been added to the project it may be referred to from *any* module or form *in the same project* by *DC1.xx*x where *xxx* is a property or method. If you are coding in *DC1*'s own code window you can simplify this further by referring to *Me.xxx*. The following code will work anywhere in the VBA project:

```
Public Sub VBADemo()
  Dim fActualDemand As Single

  fActualDemand = DC1.P_Demand * DC1.P_DemandFactor

End Sub
```

The following version will only work correctly in the code window for *DC1*:

```
Public Sub VBADemo()
  Dim fActualDemand As Single

  fActualDemand = Me.P_Demand * Me.P_DemandFactor

End Sub
```

In both of the above examples we are multiplying the DC's *General.Demand* parameter by its *General.Demand factor* parameter to set the value of the local variable *fActualDemand*.

If you have a *DC1* in your project you can cut and paste the above code into your project. Put a break point (**F9**) on the line:

```
  fActualDemand = DC1.P_Demand * DC1.P_DemandFactor
```

and with the cursor in the *Sub*, press **F5** to execute the code. Execution should pause on the line with the breakpoint. Let the mouse hover over *DC1.P_Demand* and *DC1.P_DemandFactor* and you will see the values of these properties as entered in your project. Press **F5** again to let the execution complete.

Note also the VBA *Properties* window shows the same values. You can change a value here and it will be reflected immediately in Aquator (and vice versa).

Indirect referencing

If the component is not included in the VBA project, it is still possible to reference it and its data indirectly using the *Model object* which is globally available. The example above, in direct referencing, could have been written:

```
Public Sub VBADemo()
  Dim fActualDemand As Single

  fActualDemand = Model.Components("DC1").Parameters("General.Demand"). _
                  Value * Model.Components("DC1"). _
                  Parameters("General.Demand factor").Value

End sub
```

Try pasting this code into your project and running. Note you get exactly the same values for the two Aquator parameters and the local variable *fActualDemand* after the multiplication has executed.

With indirect referencing, note that *Components*, *Constraints* and *Groups* are referred to by their *Name* and not *Display name*. This is because the user can change the *Display name* whereas the *Name* is fixed.

Coding advice

This first block of advice deals with *Parameters, States, Variables, Properties* and *Sequences.*

Parameters

You may use VBA to get the value of any parameter value using the indirect method (see *indirect referencing*). The following shows how to get the value of a parameter on any component:

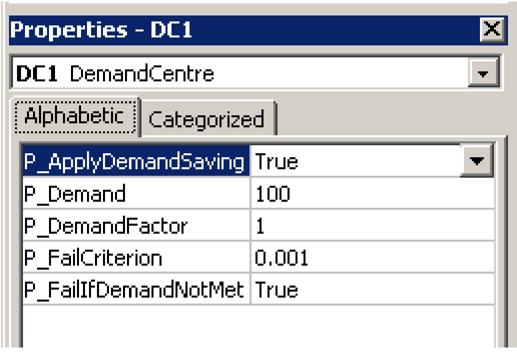
```
Public Sub VBADemo()
  Dim vValue As Variant

  vValue = Model.Components("XXX").Parameters("YYY.ZZZ").Value

End Sub
```

Where: *XXX* is the component name, *YYY* is the group name and *ZZZ* is the item name.

Quite often parameters are made more readily available to the VBA programmer by *direct referencing* as described above. If a parameter is available by this method it also appears in the VBA properties window when the code for the object is in the current VBA code window:



You can change the value of a parameter here by typing the new value in the second column:

You can also change the value of an object's parameter in VBA code by one of the three ways shown below (each method has the same effect):

```
Public Sub VBADemo()

  ' Method 1
  Me.P_DemandFactor = 1.234

  ' Method 2
  DC1.P_DemandFactor = 1.234

  ' Method 3
  Model.Components("DC1").Parameters("General.Demand factor").Value = 1.234

End Sub
```

In summary a parameter may always be inquired and/or changed by:

- Editing the parameter in Aquator
- In VBA code by the indirect method (Method 3 in the above example)

If direct referencing a parameter is allowed the following methods can be used:

- Editing the parameter in VBA
- In VBA code by the two direct methods (Method 1 and 2 in the above example)

Generally simple single number and Boolean parameters are available by direct referencing. More complex parameter types (such as arrays and enumerations) are not.

States and variables

States and variables can be inquired in the same indirect way as parameters. For example:

```
Public Sub VBADemo()

  vValue = Model.Components("XXX").States("YYY.ZZZ").Value
  vValue = Model.Components("XXX").Variables("YYY.ZZZ").Value

End Sub
```

Do not change the values of any object's state or variables data items. This may well cause unwanted side effects in the calculations.

Properties

One more level of indirection is required to get the value of a property. Note the additional *.Base* in the following example:

```
Public Sub VBADemo()
  Dim strDisplayName As String

  strDisplayName = Model.Components("DC1").Base. _
                     Properties("Component.Display name").Value

End Sub
```

Sequences

A *Sequence* is an object's requirement for time series data. These data can be provided by a full time series or a *Profile* (of annually repeating values). In VBA it is possible to get either a single daily value from a time series or profile assigned to a sequence.

Single values are returned for one day as follows:

```
Public Sub VBADemo()
  Dim bGotValue As Boolean
  Dim fDemandValue As Single

  ' Time series
  bGotValue = Model.Components("DC1").Sequences("Demand sequence") _
              .ActiveTimeSeries.GetDataForDay(CDate("1/4/1994"), _
                                                      fDemandValue)
  ' Profile
  bGotValue = Model.Components("DC1").Sequences("Demand sequence") _
              .ActiveProfile.GetDataForDay(CDate("1/4/1994"), _
                                                      fDemandValue)

End Sub
```

In the above example if *bGotValue* is returned = *True* then a valid value is in *fDemandValue* after the call. If a time series or profile is not assigned, *.ActiveTimeSeries* or *.ActiveProfile* are set to *Nothing*.

A large number of values are returned by the method *GetValues* (rather than *GetDataForDay*).

Array-type data items

Parameter and state values stored as Variants can contain an entire two-dimensional array. The first subscript on the array is the row number, the second the column number.

An empty array will be dimensioned like

(0 to 0, 0 to 0)

while a non-empty array is dimensioned like

(1 to Rows, 1 to Cols)

i.e. the *UBound* function always returns the number of rows and number of columns.

An example of how to access an array parameter follows.

```
Dim vParameter as Variant
Dim fMaxStorage As Single
vParameter = RV1.Component.Parameters("Shape.Level Area _
                                      Storage|Level|Area|Storage|").Value
fMaxStorage = vParam(1, 3)
```

Names of parameters, states, sequences, and variables

We have described the syntax of direct referencing and indirect referencing e.g. for the parameter which specifies a constant demand for a demand centre

```
DC1.P_Demand
```

and

```
Model.Components("DC1").Parameters("General.Demand").Value
```

Here we want to explain why there are two ways of getting at every parameter (and state, sequence, and variable) and the naming conventions that have been adopted.

In the indirect method example above the word **Parameters** refers to what is known in VBA as a Collection. VBA requires that we refer to members of a collection either by a numerical index like **Parameters(1)** or by name like **Parameters("General.Demand")**. Aquator components use names in a *Group.Name* format so that when displayed in an editing grid parameters in same *Group* appear together.

The same applies to states and variables. For example, the time of travel of a river reach is referred to like **States("Flow.Travel time").Value** while the calculated reservoir storage is referred to like **Variables("Storage.Calculated")**.

Sequences are similar but the names are not grouped, for example a reservoir rainfall sequence is referred to like **Sequences("Rainfall")**.

The advantages and disadvantages of using collections and indirect naming versus using direct naming are mirror images of each other, which is the essential reason for providing both.

- flexibility: the names of items in a collection can include spaces and the period character, as used in the *Group.Name* convention, but directly-addressed names must be a valid VBA identifier consisting of letters of the alphabet and the digits 0-9 only (first character must be a letter)
- extensibility: you can add your own parameters to the Parameters collection, your own variables to the Variables collection, and so on, but directly addressed names cannot be added
- speed of execution: every time an item is referenced by name its collection must be searched to find the item with the specified name, which is slow, whereas a directly-addressed parameter, state, sequence, or variable access is fast
- ease of coding: you may make a mistake when typing strings like "General.Demand" and you have to know what to type either by studying the documentation or examining the names in Aquator; conversely, as soon as you type the period character in a line of code like "DC1." VBA will present a list of known directly-addressed names like P_GeneralDemand; use the arrow keys to scroll to the name you want and then hit the Tab key to paste in the name with no possibility of error

Directly-addressed parameters, states, sequences, and variables obey a naming convention whereby a prefix shows which collection the name is associated with.

Item Type	Name Prefix
Parameter	P_
State	S_
Sequence	Q_
Variable	V_

Not only does this distinguish between the different types, it makes it quicker to select the required name from the list which appears in VBA when you type the period in a string like "DC1." – if you then press "P", for example, the parameter names become visible.

Code location

Where do we place our VBA code that modifies Aquator's calculations as the model runs? The answer is that each component fires *events* at various stages during model execution and we can place our code in these events. For *events* to be fired the Model parameter *Options.Events on* must be set = *True* and the component parameter of the same name must also be set = *True*.

You can see a list of events that each component has available by:

- Right clicking on the component on the schematic
- Selecting menu item *Customise …*
- The VBA form is now shown with the code window for the component active.
- The code window has two drop down list boxes on the top, the drop down list on the left says '(General)'. Change this to the next item on the list. (The name will vary between component types; for a demand centre the name is *DemandCentre*, for a water treatment works it is *WaterTreatmentWorks* etc.)
- Now drop down the list on the right hand side. Items appear such as *AfterFinish*, *AfterLoad* and *BeforeFinishDay*. This is a complete list of events for which you may insert VBA code. These events occur at different stages during model execution.

If you select an event, a template is pasted into the code window into which you may type your VBA code. For example, selecting the *BeforeInitializeDay* event for a demand centre produces:

```
Private Sub DemandCentre_BeforeInitializeDay(ByVal Timestamp As Date, _
                                             ByVal Step As Long)

End Sub
```

The events that are available depend on the *Interfaces* that each component type supports. *Interfaces* define what a component is capable of. All components support at least two interfaces that create events:

- *IBaseObject* (Every Aquator object has this interface)
- *IComponent* (Active Aquator components have this interface)

IBaseObject has just two events; *AfterLoad* and *BeforeUnload*

IComponent has 16 events

Each event is listed below in the order in which they occur:

Event	Timing
AfterLoad	As soon as the object has been loaded from the database
BeforeInitialize	Before first day of the model run – pre-component initialisation
AfterInitialize	Before first day of the model run – post-component initialisation
BeforeStart	Before first day of the model run – pre-component start
AfterStart	Before first day of the model run – post-component start
BeforeInitializeDay (*A good place to alter a component's parameters*)	Before every day of the model run – pre-component initialisation
AfterInitializeDay (*You may set your own demand saving level here*)	Before every day of the model run – post-component initialisation
BeforeStartDay (*[**W**] You can change the demand saving factor supplied to the component here.*)	Before every day of the model run – pre-component start
AfterStartDay (*[**W**]*)	Before every day of the model run – post-component start
BeforeFinishDay (*[**W**]*)	After every day of the model run – pre-component finish
AfterFinishDay (*[**W**]*)	After every day of the model run – post-component finish
BeforeTerminateDay	After every day of the model run – pre-component terminate
AfterTerminateDay (*If you have altered the water balance, this is the place to report what you have done by changing the value of parameters:* *AmountAdded* *AmountStored* *AmountLeaked* *AmountLost* *You can also alter the cost returned and set the status level returned by the component to one of the following:* *Status = aqtStatusError* *Status = aqtStatusFailure* *Status = aqtStatusOK* *Status = aqtStatusWarning*)	After every day of the model run – post-component terminate

BeforeFinish	After last day of the model run – pre-component finish
AfterFinish	After last day of the model run – post-component finish
BeforeTerminate	After last day of the model run – pre-component terminate
AfterTerminate	After last day of the model run – post-component terminate
BeforeUnload	Just before the component is unloaded as the project is closed

Note: [**W**] = components are only allowed to move water at this time. VBA code that causes water to be moved must be placed only in these four events.

In general it is better to locate the customisation code for a component in the code module provided for that component. If, for example you wished to alter the compensation release from a reservoir on a day by day basis, it would be advisable to place this VBA code in the code module for the reservoir concerned.

As a general rule, non-demand centre type components receive their events before demand centre components. This is because all requests for supply originate from demand centres and components on the supply chain need to be in a ready state to respond.

As an example, let us suppose that we want to alter a reservoir's compensation release when the level in the reservoir was less than 40% full. The VBA code would be located in the *BeforeInitializeDay* event (so the reservoir will use the correct compensation value for the day in question). The code might look like this:

```
Private Sub Reservoir_BeforeInitializeDay(ByVal Timestamp As Date, _
                                          ByVal Step As Long)
  If Me.V_StorageCalculatedPC < 40! Then
    Me.P_CompensationFlow = 45.3
  Else
    Me.P_CompensationFlow = 60.2
  End If
End Sub
```

Note that events that occur every day during a model run always let you know what day is being processed (in *Timestamp*) and the number of days into the model run (in *Step*). You could use this information to control your VBA code. In the example above, suppose we only wanted to reduce the compensation in April if the reservoir was less than 40% full. The code would now look like this:

```
Private Sub Reservoir_BeforeInitializeDay(ByVal Timestamp As Date, _
                                          ByVal Step As Long)
  If Month(Timestamp) = 4 And Me.V_StorageCalculatedPC < 40! Then
    Me.P_CompensationFlow = 45.3
  Else
    Me.P_CompensationFlow = 60.2
  End If
End Sub
```

Using VBA when the model is not running

The key differences between using VBA when the model is running and when the model is not running are

- what your code can access
- when your code executes
- where your code is located

What your code can access

Almost everything in an Aquator model can be accessed by VBA code irrespective of whether or not the model is running, but there are a couple of things to be aware of. It doesn't make much sense to look at the values of variables except during model execution. And neither does it make sense to look at the model results unless a model run has been completed.

When your code executes

During a model run custom code executes by handling ('trapping') an event like *BeforeInitializeDay*, as described previously. If the model is not running then these events never occur ('fire') and instead one of two user actions trigger execution of custom code.

Both these user actions can trigger execution of the same custom code and which you use depends mostly on which window (Aquator or VBA) is in the foreground.

In the VBA IDE you put the insertion point (cursor) somewhere in the subroutine you wish to execute, by clicking, and then press F5 to run. You can also press F8 to step into the code i.e. to control execution line by line. See the Debug menu for other options.

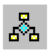
If the Aquator window is in the foreground then custom code can be executed by running a macro. A macro is simply a subroutine with no arguments which is “Public” and located in a VBA module.

```
 Public Sub MacroDemo()

  ' Macro shortcut key: Ctrl + M
  ' Macro description : Demonstration macro changes a parameter value

  Model.Components("GW1").Parameters("Options.Events on").Value = False
End Sub
```

There are several ways to execute a macro:

- the  toolbar button has an adjacent small down-arrow symbol; clicking this arrow presents a list of macros limited to ten in number, ordered according to the order they are found in, first, the General VBA project, then in the active VBA project (corresponds to the project you are looking at in the Aquator window)
- the *Tools...Macro...Macros...* menu command shows a dialog listing all macros in any selected module; from this dialog you can run the macro or step into (debug) it
- a macro can have a shortcut key associated with it (the above example has a shortcut key combination of the Control key plus upper-case 'M'); press the shortcut key combination to run the macro

Where your code is located

If you adopt the technique of pressing F5 to run custom code from the VBA IDE then you can place your code in any form or module. But to run a macro from the Aquator main window your macro code must be located in one of the following places:

- a module in the VBA **General** project
- a module in the VBA project that corresponds to the active Aquator project (the project in the foreground window if you have multiple projects open)

If the action of the macro does not affect any project at all, or can be applied to any project, then place the macro in a module in the **General** project. A macro which should only execute in the context of one particular project should be placed in a module of that project.

A macro in the **General** project can refer to the model object of the active Aquator project by referencing *ActiveProject.Model*:

```
Public Sub GeneralShowName()
  Dim strName As String
  strName = ActiveProject.Model.Name
  MsgBox "This project's name is " & strName, vbOKOnly or vbInformation
End Sub
```

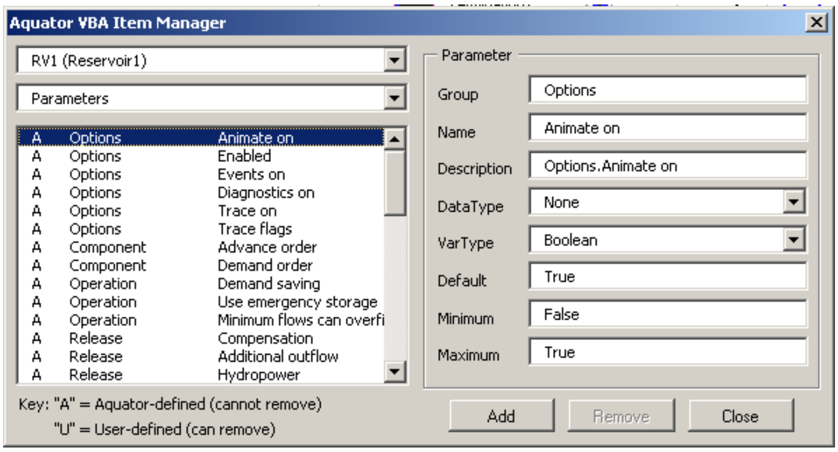
A macro in a VBA project associated with a particular Aquator project can refer to the Model object of that project directly, as in the following example:

```
Public Sub ProjectShowName()
  Dim strName As String
  strName = Model.Name
  MsgBox "This project's name is " & strName, vbOKOnly or vbInformation
End Sub
```

VBA Item Manager Form

An example Aquator VBA macro *VBAItemManager* is installed with Aquator. The macro and its associated form perform a useful function that will help you add your own parameters, variables, sequences etc to a model and its components.

When the macro is running the following form is displayed on top of your Aquator project:



This form and code behind it was written entirely in VBA within Aquator. In addition to demonstrating the use of VBA when the model is not running, the form will allow you to add (and remove) parameters, sequences etc to your model or any of its components. Data items added by yourself are not used by Aquator but are there ready for you to use within your VBA project. For example you could add a parameter to a reservoir to indicate a reduced compensation flow, this parameter could then be edited along with all other normal reservoir parameters by the end user. Your VBA code could inquire the value of the parameter when the model was running and use it in its calculations.

Similarly you could add your own variable to a component. By trapping the appropriate event (probably *AfterTerminateDay*) for that component you could set the value of the variable each day. That variable can then be used and plotted in exactly the same way as any built-in variable.

The macro to show the above form is called *VBAItemManager* and is located in the *Macros* module of the **General** project, which itself is stored in a file *Aquator.VBA* located in your default projects folder. If you accepted all the default options when Aquator was first executed then all of this will have been automatically set up for you. Otherwise you will need to add the form yourself.

A backup copy of the form is held in two files, *AQTManager3.frm* and *AQTManager3.frx,* located in the Aquator installation folder. Steps 1 to 5 that follow copy these files to your default projects folder if necessary, but this should have been done the first time Aquator was executed.

You can include the form above it in your project by:

1. With no projects open in Aquator select menu item **File – Projects folder**.
2. Check the box *Copy example VBA code modules and forms*.
3. Uncheck the other check boxes.
4. Select the destination folder and click *OK*
5. Files *AQTManager3.frm* and *AQTManager3.frx* will then be copied to your destination folder.
6. In VBA use menu item **File – Import** to select file *AQTManager3.frm*. Note it is added to your project under the **Forms** group in the **General** project of the VBA Project window
7. To show the form by executing a macro you need to write a macro something like the following in a module of the **General** project

```
Public Sub VBAItemManager()

  ' Macro shortcut key: Ctrl + V
  ' Macro description : show form to manage user-defined items

  AQTManager3.Show
End Sub
```

The form may then be used to add *Properties*, *Parameters*, *States*, *Variables* and *Sequences* to either the model or any component.

When naming an Aquator data item (*Property*, *Parameter*, *State* or *Variable*) use a period '**.**' to separate the group name from the item name. (e.g. *Pumping proportion.A3 (VBA)* where *Pumping proportion* is the group name and *A3 (VBA)* is the item name.) Sequences do not have groups. Make sure the name you supply is unique.

You will be coerced to include the text "VBA" in your name. If you force a name that does not include this text then we cannot guarantee that the next release of Aquator won't use the very name you have chosen. This will create a bad error when loading the project, so please include the "VBA" text in the name of any item you define.

This form supports six variant (VarType) types:

- Floating point number (type default value as a number with decimal point e.g. 1.23)
- Integer number (type default value as a number without decimal point e.g. 1996)
- Boolean (type default value as a string 'True' or 'False')
- String (type default value as anything)
- Date (type default value as a date like 23 Jan 2002)
- Enumeration (type the choices separated by '|' characters as in "|Demand order|Equitable|Equitable/Other sources|" in the Minimum field, the integer values corresponding to those choices like "|1|2|3|" in the Maximum field, and the default integer choice in the Default field)

Data items and sequences added by VBA programmer can also be removed by this form.

This macro and form is supplied as an addition to Aquator for which Oxford Scientific Software cannot guarantee support. Known limitations of the supplied form and code are:

- Data items and sequences cannot be added to constraints or groups (although Aquator allows this)
- Array data types are not supported by the form
- Error checking in the code is not thorough

Recording macros

Aquator provides a macro recording facility started by choosing the menu item *Tools...Macro...Record New Macro* or by pressing Ctrl + F9. See *Record Macro Dialog* for a description of the dialog that initiates macro recording.

One use for macro recording is as a learning tool. Once macro recording is on some (but not all) of your actions will result in VBA code being inserted into the macro.

Here is a partial list of actions which will generate code and the objects to which they apply.

Application

- CreateProject
- LoadProject
- Set ActiveProject
- Let OptionSetting
- RunMacro

Project and Model objects

- Unload
- Save
- Set ActiveScenario
- RunModel
- SwitchToNodeInTreeView

Scenarios

- Add
- Remove
- Rename

Scenario

- AddParameterSetRef
- RemoveParameterSetRef
- AddStateSetRef
- RemoveStateSetRef
- AddSequenceSetRef
- RemoveSequenceSetRef
- AddVariableSetRef
- RemoveVariableSetRef
- Set ActiveParameterSet
- Set ActiveStateSet
- Set ActiveSequenceSet
- Set ActiveVariableSet

ProjectSets

- Add
- Remove
- Rename

ComponentParameterSets

- Add
- Remove
- Rename

Assign

AnnotationItems

Add

Clone

Remove

ComponentItems

Add

AddSupplyType

AddFlowType

Clone

Remove

ComponentItem

AddConstraint

RemoveConstraint

Move

Let ConnectorAngle (Get)

AddInConnector

AddOutConnector

AddConnector

RemoveInConnector

RemoveOutConnector

RemoveConnector

Connect

Disconnect

GetMenuCommandInfo

ExecuteMenuCommand

General (works on all types of objects)

ModifyParameterValue

ModifyStateValue

ModifySequenceLink

ModifyVariableFlags

How to use Aquator from another application

So far we have described using VBA from within Aquator. It is also possible to use VBA to control Aquator from any other program that incorporates VBA, such as Excel or Word.

To do this you open the VBA IDE from the other application and write some code which will cause Aquator to run. Once Aquator is running you can use VBA to load or create a project, run a model, etc., exactly as if you were using VBA inside Aquator.

Often the choice of where the VBA code resides is arbitrary. You could use a VBA macro inside Excel to load and run an Aquator model, and then get model results into a spreadsheet. Equally you could use VBA inside Aquator to load and run Excel, and paste model results into Excel.

Loading and running a model

As a concrete example we will describe how to run an Aquator model from another application as a series of steps and also as some example VBA code.

- run the other application and open the VBA IDE (*Tools...Macro...Visual Basic Editor* or Alt + F11)
- add a reference to Aquator.exe by going to *Tools...References...,* check the "Aquator Water Resource Simulation" item
- add a reference to IAquator.dll by going to *Tools...References...,* check the "Aquator Public classes" item, and click OK
- add a module where some custom code can be written and start a new macro called, say RunAquator

```
Public Sub RunAquator()

End Sub
```

- add the code to create an instance of Aquator and show Aquator's main window

```
  Dim objAquator As Aquator.Application

  Set objAquator = New Aquator.Application
  objAquator.Visible = True
```

- open an Aquator database by specifying the folder (uses default projects folder if not specified) and the name of the database file

```
  Dim objDatabase As Aquator.Database

  Set objDatabase = objAquator.LoadDatabase("", "AquatorDemo1.mdb")
```

- load the list of projects that this database contains

```
  Dim nProjectCount   As Long
  Dim nIDList()       As Long
  Dim strNameList()   As String

  nProjectCount = objDatabase.GetProjectList(nIDList, strNameList)
```

- load the first project

```
Dim objProject  As Aquator.Project

Set objProject = objAquator.LoadProject(objDatabase, _
                   strNameList(1), False, False)
```

- run the model – the call to ModelRunStart doesn't return until the model run ends

```
Dim objModelRun   As Aquator.ModelRun

Set objModelRun = objProject.ModelRunStart()
```

- retrieve the model run log

```
Dim n        As Long
Dim strLog   As String

For n = 1 To objModelRun.LogSize
  strLog = objModelRun.ReadLog(n)
Next n
```

- retrieve the amount of water added to the system during the run

```
MsgBox "Amount added = " & objModelRun.ResultSet.AmountAdded
```

- show the value of a particular variable on the second day of the run

```
MsgBox "AB2 upstream flow on day 2 = " & _
 objModelRun.ResultSet.Results ("AB2.Flow.Upstream").GetValue(2), _
 vbOKOnly or vbInformation
```

Aquator implements a rather complete object model of properties, methods, and events which allows full control from another application. To learn more either review the object model diagram included in your documentation, or explore the object model itself using VBA.

Error codes

When controlling Aquator from another application, programming mistakes may result in the following errors being generated. In the following list each error has a public symbolic name prefixed *'aqt'* that can be used in VBA code.

The numerical value of each symbol can be used instead but to guarantee no conflicts with existing error codes these are large negative numbers e.g. *aqtErrorUnexpected = -2147220270* and so the symbolic name is more comprehensible and should be used instead.

aqtErrorAlreadyInScenario

The call to add a set to a scenario failed because the set is already in the scenario

aqtErrorAutomationProtected

The call to unlock a schematic or save a project failed because it is password-protected.

aqtErrorCannotAddBiDiConnector

The call to add a bidirectional connector to the specified component failed because this type of component cannot add any more bidirectional connectors

aqtErrorCannotAddInConnector

The call to add an input connector to the specified component failed because this type of component cannot add any more input connectors

aqtErrorCannotAddOutConnector

The call to add an output connector to the specified component failed because this type of component cannot add any more output connectors

aqtErrorCannotDeleteActiveScenario

The call to delete a scenario faiiled because the spcified scenario is active

aqtErrorCannotDeleteActiveSet

The call to delete the specified set failed because the set is active

aqtErrorCannotDeleteComponentSet

The call to delete a component set failed because the specified component set is used by at least one project set

aqtErrorCannotMakeSetActive

The call to make the specified set active failed because the set is not part of the active scenario

aqtErrorCannotMoveConnectedLink

The call to move a component failed because the component is link-type and connected to another component, and hence not free to move

aqtErrorCannotRemoveActiveSet

The call to remove a set from a scenario failed because the specified set is active

aqtErrorCannotRemoveBiDiConnector

The call to remove a bidirectional connector from the specified component failed because the specified connector is permanent and cannot be removed

aqtErrorCannotRemoveInConnector

The call to remove an input connector from the specified component failed because the specified connector is permanent and cannot be removed

aqtErrorCannotRemoveOutConnector

The call to remove an output connector from the specified component failed because the specified connector is permanent and cannot be removed

aqtErrorChartNotFound

The call to paste a chart onto the clipboard or to convert a chart to a metafile failed because the chart was not found

aqtErrorConnectedAlready

The call to make a connection failed because the connector is already connected

aqtErrorConnectIncompatibleTypes

The call to make a connection failed because the specified connectors are incompatible e.g. both input or both output etc

aqtErrorConnectNotLinkType

The call to make a connection failed because the connector being dragged does not belong to a link-type component (a component with two connectors joined by a line)

aqtErrorInvalidActiveProject

The call to switch to a different active project failed because the project specified does not exist

aqtErrorInvalidRunToDate

The call to run the model to a certain date failed because the date is invalid

aqtErrorMacroNotFound

The call to execute a macro failed because the specified macro was not found

aqtErrorMenuItemNotEnabled

The call to execute a components custom menu command failed because the menu command specified is disabled

aqtErrorMenuItemNotFound

The call to execute a components custom menu command failed because the menu command specified does not exist

aqtErrorModelAlreadyRunning

The call to run the model failed because the model is already running

aqtErrorModelIsRunning

The call to save or load the project failed because the model is running

aqtErrorModelNotRunning

The call to single-step, pause, or stop the mode run failed because the model is not running

aqtErrorNoBiDiConnectors

The call to add or remove a bidirectional connector to or from the specified component failed because this type of component cannot contain a bidirectional connector

aqtErrorNoBoundary

The call to move a connector around the component boundary failed because the component does not have a boundary i.e. it is a 1-dimensional link-type component

aqtErrorNoConstraints

The call to add or remove a constraint to or from the specified component failed because this type of component does not use constraints

aqtErrorNoInConnectors

The call to add or remove an input connector to or from the specified component failed because this type of component cannot contain an input connector

aqtErrorNoOutConnectors

The call to add or remove an output connector to or from the specified component failed because this type of component cannot contain an output connector

aqtErrorNoSuchBiDiConnector

The call to remove a bidirectional connector from the specified component failed because the specified connector was not found

aqtErrorNoSuchInConnector

The call to remove an input connector from the specified component failed because the specified connector was not found

aqtErrorNoSuchOption

The call to get or set an option failed because the specified option was not found

aqtErrorNoSuchOutConnector

The call to remove an output connector from the specified component failed because the specified connector was not found

aqtErrorNotConnected

The call to disconnect failed because the connector is not connected

aqtErrorNotInComponentSets

The call to make a particular component set active failed because the specified set is not in the specified components sets

aqtErrorNotInScenario

The call to remove a set from a scenario failed because the set is not in the scenario

aqtErrorProjectNotFound

The call to load a specified project failed because the project was not found in the database

aqtErrorUnexpected

An unexpected error has been detected, the remainder of the error message will give more information

Aquator Reference

This reference section gives details on general aspects of Aquator. For details on specialist items also consult the chapters on:

Annotations

Components

Connectors

Constraints

Exporters

Groups

Importers

Reporters

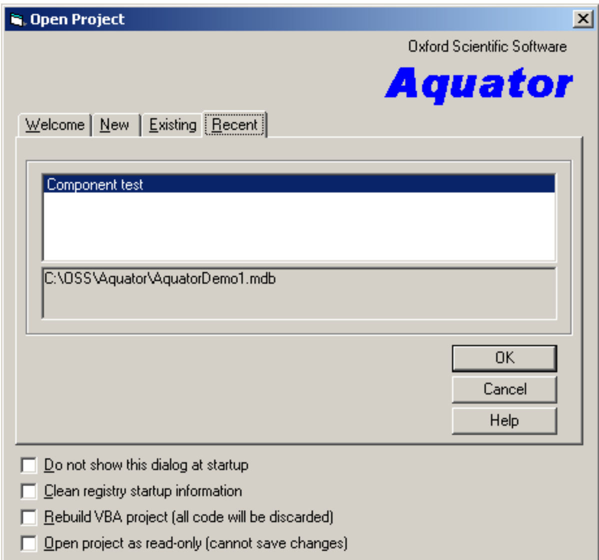
HYSIM Catchment

Starting Aquator

This section describes how Aquator presents to the user the organisation of data by Projects and Databases.

Startup Dialog

This dialog can appear when Aquator starts although an option can suppress its appearance (see *Options Dialog)*



The behaviour of the dialog depends on the command line and the setting of the option which determines which tab appears by default. The possibilities are:

Welcome Tab

The Welcome tab appears by default only when Aquator is run for the very first time.

New Tab

The New tab appears by default if there is no command line and the option to default to the *Existing Tab* is false.

Existing Tab

The Existing Tab appears by default if a database is named on the command line; the list box will contain all projects in that database.

Recent Tab

The Recent tab appears by default either if a project is named on the command line or if the option to default to the *Existing Tab* is True.

To name a database on the command line specify the full path to the database, in quotes if necessary i.e. if the path has embedded spaces

To name a project on the command line concatenate the full path to the database with the name of the project separated by a backslash, entirely enclosed in quotes if the command line argument has embedded spaces

Do not show this dialog at startup

If checked, this dialog will not appear the next time Aquator starts.

Clean registry startup information

If checked, will remove the recent project and database names from the Windows registry so that this dialog will be 'clean' of previous project and database names.

Rebuild VBA project (all code will be discarded)

If checked then any VBA customization code in the project will be discarded when the project is loaded into memory. To delete VBA customization code entirely the project must then be saved i.e. this option does not alter the copy of the project in the database.

Open project as read-only (cannot save changes)

If checked then the Save command on the File menu and the Save button on the toolbar will be disabled. This option serves only to guard against accidentally saving a project which has been modified. This read-only attribute can be added or removed at any time using the File...Read-Only command.

Projects and Databases

An Aquator **project** defines a **model** of a water resource system plus all the information needed to use the model. Often we use the words **project** and **model** interchangeably.

A project is stored in a single **database**. The version of Aquator current at the time of writing uses a Microsoft Access database (MDB) file for this purpose but future versions may use other database systems. You do not need to have Access installed since Aquator contains all the functionality required for reading and writing Access databases.

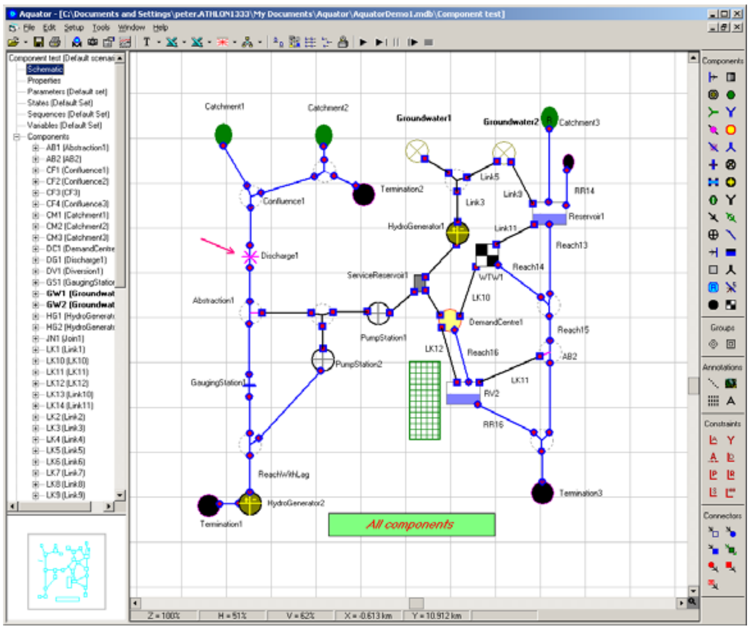
An Aquator **database** can contain

- any number of projects
- time series and profile data, shareable between projects
- pictures used to annotate model schematics, shareable between projects.
- the results from any number of model runs, shareable between projects.

Aquator is a *Multiple Document Interface (MDI)* application. This means that you can open more than one project within Aquator by selecting *File … Open* from the *File* command on the *Aquator Menu Bar* where each project has its own child window within the *Main Aquator Window*.

Normally a user would have a single database with many projects contained within it.

Main Aquator Window



Within this window there is:

- The **Aquator Menu Bar** at the top for accessing menu commands *(see p. 103)*.
- The **Aquator Toolbar**, just below the menu bar, provides shortcut buttons for commonly used actions *(see p. 97)*.
- The **Aquator Toolbox** adjacent to the right-hand edge provides access to the objects used to construct a model.

The window contains three panes:

1. The **Aquator Tree View Pane** along the upper-left edge presents all the objects and properties of the project in a hierarchical *Microsoft® Windows Explorer* format.
2. The **Aquator Information Pane** is the large central area which displays information for whichever node is selected in the Tree View Pane; in the above example the model schematic has been selected.
3. The **Aquator Thumbnail Pane** at lower-left displays a miniature image of the entire model schematic plus a rectangle showing the portion of the schematic visible in the information pane; dragging the rectangle with the mouse allows easy navigation around a large schematic.

Aquator Data Value Types

Aquator distinguishes between 5 different categories of data that can be assigned to the model or any one of its components.

They are:

1. *Property* – a non-result changing value describing the model or component.
2. *Parameter* – a value which affects behaviour of the model or component but typically is not changed between model runs.
3. *State* – a value which affects behaviour of the model or component and typically changes during and between model runs*)*.
4. *Variable* – a value computed daily during a model run i.e. results.
5. *Sequence* – sequential daily hydrological or other data input to the model or a component.

General User Interface Rules

The following general rules apply consistently throughout the user interface

Right Mouse Button Click

Use the Right mouse button to bring up context-sensitive menu commands.

Left Mouse Button Click

Use the Left mouse button to select components, data fields etc.

Entering Data In Dialog

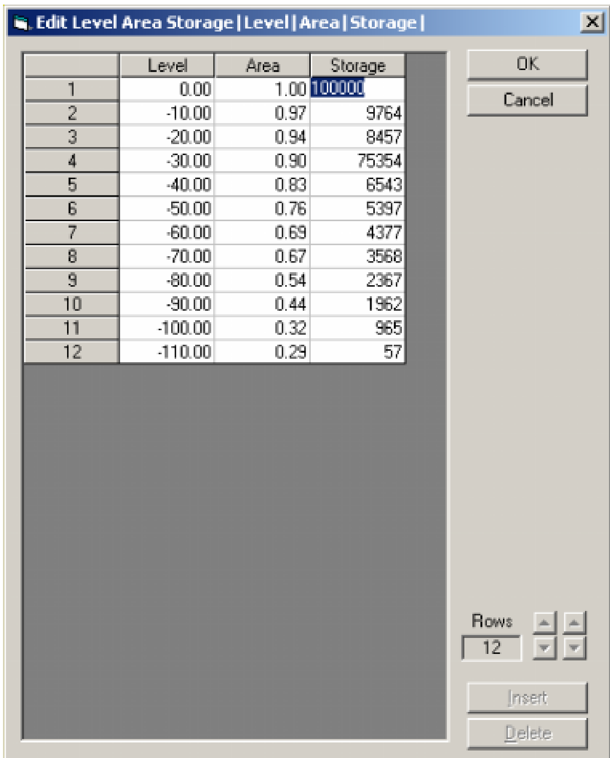
When entering data in a field of a dialog, the following keys terminate the data entry with corresponding actions.

Key	Action
Enter	Accept changed value
Enter...Enter	Accept changed value and close dialog
Tab	Move to next control button

Within a dialog, those fields that the user may be able to edit are displayed with 3 different background colours:

Colour	Data Requirement
White	Data mandatory in this field.
Grey	Data optional in this field.
Black	Data not applicable in this field.

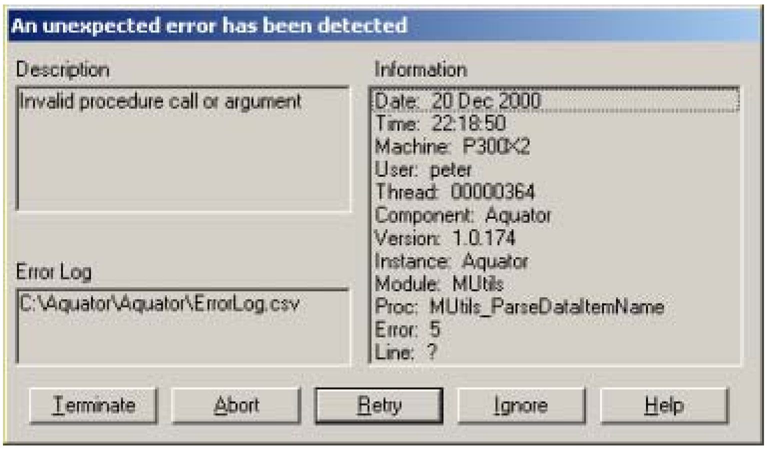
Array Edit Dialog



	Level	Area	Storage
1	0.00	1.00	100000
2	-10.00	0.97	9764
3	-20.00	0.94	8457
4	-30.00	0.90	75354
5	-40.00	0.83	6543
6	-50.00	0.76	5397
7	-60.00	0.69	4377
8	-70.00	0.67	3568
9	-80.00	0.54	2367
10	-90.00	0.44	1962
11	-100.00	0.32	965
12	-110.00	0.29	57

This dialog appears when a parameter that is actually an array is edited. The number of columns corresponds to the fixed second dimension of the array. The number of rows can be changed using the up-down buttons unless the parameter is read-only or the number of rows is fixed, in which cases these buttons will be disabled.

Error Dialog



This dialog appears if Aquator detects an unexpected error. Report this to Oxford Scientific Software Ltd by email to **support@oxscisoft.com**, attaching the ErrorLog.csv file from the default project folder directory. This file will contain the textual information in the above dialog.

You may choose to

- Terminate: end the application immediately; this is the safest option
- Abort: abort the operation where the error was detected; often this allows Aquator to continue running but normally you should save your project and restart
- Retry: retry the operation that caused the error; rarely will this recover unless the problem is temporary e.g. an unexpected database lock
- Ignore: ignore the error and continue the current operation; use with great caution

Components

Symbols for all the available Components and Annotations for the current project are to be found on the Aquator Toolbox. Any number of components may be added to the water system schematic by left clicking on the corresponding symbol and dragging it to the required location on the schematic.

For detailed information on the behaviour of each type of component, consult the Chapters on:

Annotations

Components

Rules For Linking Components

1. Linking components are used to connect fixed components together. The standard linking components in Aquator are:

   Link

   Reach

   Gauging Station

   Discharge

2. A fixed component and a linking component are connected at their **connection point**.
3. Connection points may be squares or circles either red or blue. The shape and colour have the following significance:

   Red connection points are **Outputs**

   Blue connection points are **Inputs**

   Circle connection points are **Natural**

   Square connection points are **Man-made**

4. Fixed component outputs (**Red**) can only be connected to linking component inputs (**Blue**).
5. Fixed component inputs (**Blue**) can only be connected to linking component outputs (**Red**).
6. A Natural connection point (**Circle**) can only be connected to another Natural connection point (**Circle**).
7. A Man-made connection point (**Square**) can only be connected to another Man-made connection point (**Square**).
8. When a connection has been made the connection point appears in red with a blue border
9. Connections remain in place when the fixed component is moved on the schematic.

Connecting Components

Ensure that the components to be connected follow the **Rules for Linking Components** and then carry out the following.

1. Left click on the linking component input or output connection point and, with the mouse button held down drag it towards the output or input connection point of the fixed component.
2. Notice that as you get close to the fixed component connection point**,** the size of the linking component connection point doubles. At this point release the mouse button and the end of the linking component will drop onto

the fixed component. A connection is made if the connection point is coloured red with a blue border. Also that if you drag the fixed component over the schematic, the connection remains in place.

Disconnecting Components

Right click on the connection point and from the drop down menu select **Disconnect**.

Adding or Removing Connection Points

To add another connection point to a component, right click on the component and from the drop down menu select **Add Input** or **Add Output** (depending on the type of component).

Some components have a fixed number of connection points (e.g. Links and Joins). In these situations the **Add …** menu commands will be greyed-out.

How to use On-Line Help

Aquator is provided with comprehensive on-line help. This can be initiated by either one of the Help menu commands or by pressing the F1 key.

Help Files

The on-line Help system comprises of a set of files that describe each Chapter in the Aquator printed manual (e.g. Aquator, Components etc.) The relevant Help File is automatically opened when using *Context Sensitive Help*.

The Help Window

The Help Window opens with three panes:

1. Toolbar – contains buttons for Hide/Show, Locate, Back and Print
2. Left hand pane – tabs for Contents, Index and Search.
3. Right hand pane – help text with hyperlinks to other help topics and navigation buttons

Hide/Show

Use this to hide or show the Left-hand pane.

Locate

Use this to locate the current help topic in the Table of Contents.

Back

Use this to move back to the **Previously Viewed** help topic.

Print

Use this to print the current help topic.

Contents Tab

Displays the Help Contents in hierarchical view. Click on each section heading to see sections within it. This will also display the associated help text.

Index Tab

Displays Help Index in alphabetical list. Click on any item to display help topic

Search Tab

Facility to find all help topics containing keyword. Type the keyword(s) you wish to search on (e.g. 'Components') and click on the List Topics button. After list is displayed, double click on any item to display the help topic. The keywords within the topic will be White text blocked in Blue.

Help Text and Hyperlinks

Click on any text displayed with Blue Underline to jump to the referenced help topic.

Help Text Navigation Buttons

These are displayed at the top of each help topic. Clicking on them will allow you to:

1. Display **Contents**.
2. Move back to **Previous** help topic in help manual.
3. Move forward to **Next** help topic in help manual.

Tool Tips

The standard Tool Tips facility is supported for all buttons on the *Aquator Toolbar* and all Components on the **Aquator Toolbox**.

To operate, make sure the Aquator Main Window is in focus (by clicking on the bar at the top of the window if necessary) and then move the mouse cursor over a button or component without clicking and leave it there. After one second, a yellow text box will pop up displaying the name of the button or component.

Context-Sensitive Help

To view the help topic for any specific item in Aquator (e.g. Component or Field in a Dialog), click on the item and press F1. This will automatically open the relevant help file and display the corresponding help topic.

Aquator Menu Bar

This is the menu bar at the top of the Main Aquator Window. It provides access to the menu commands:

- **File New** … to create a new project *(see p. 165)*.
- **File Open** … to open an existing project *(see p. 168)*.
- **File Close**... to close a project *(see p. 142)*.
- **File Save**... to save a project
- **File Save As**... to save a copy of the project
- **File Save As Text**... to save the project as a text file
- **File Import VBA Code**... to import VBA code
- **File Export VBA Code**... to export VBA code
- **File Reload VBA Code**... to reinitialize the VBA subsystem
- **File Read Only**... to toggle the read-only flag
- **File Print**... to print the contents of the information pane, *or* the contents of all checked nodes in the project tree view
- **File Print Setup**... to set up and/or change the printer
- **File Print Preview**... to preview the appearance of printed information
- **File Database** … to manage an existing database *(see p. 151)*.
- **File Model Runs**... to store model runs or to retrieve previously-stored runs
- **File Project Differences**... to compare projects *NEEDS LINK)*.
- **File Projects Folder ...** to setup projects folder *(p. 151).*
- **Edit Copy**... to copy all selected components to the internal paste buffer
- **Edit Paste**... to paste the contents of the internal paste buffer onto the schematic
- **Edit Remove**... to remove all selected components
- **Edit Find**... to find on the schematic a specified component or all components of a specified type *(see p. 157)*.
- **Edit Find Again**... to find the component last searched for again.
- **Edit Find Previous**... to find the previous component of the last-specified component type.

- **Edit Find Next**... to find the next component of the last-specified component type
- **Edit Selected**... to edit or compare the components which have been selected on the schematic *(see p. 180)*.
- **Edit Components**... to select, edit, and compare multiple components *(see p. 180)*
- **Edit Chart variables**... to edit how charts will appear *(see p. 190)*.
- **Edit User defined**... to show a custom editing dialog.
- **Setup Model**… to change the model *(see p. 161)*.
- **Setup Routes**… to add or remove auto-generated custom code that can control the use of available routes between demands and supplies.
- **Setup Component Order**… to change the order in which components move water during a model run.
- **Setup Demand Saving Groups**… to set up demand saving groups.
- **Setup Diagnostics and Trace** … to set up trace and diagnostic options *(p. 152)*.
- **Setup Keyboard** … to choose accelerator keys *(p. 159)*.
- **Setup Toolbox** .... to change available tools *(see p. 176)*.
- **Setup Data Types**.... to change the number of decimal places used to display different types of data.
- **Setup Options** .... to change Aquator options *(see p. 168)*.
- **Setup Scenarios**... to setup scenarios *(see p 178)*.
- **Setup Sets**... to setup sets *(see p. 178)*.
- **Setup Tools**... to setup for Report, Import and Export tools *(see Chapters on Aquator Importers and Exporters).*
- **Tools Report**... to generate reports
- **Tools Import** …for time series or profile data.
- **Tools Export**… for time series or profile data.
- **Tools Analyze**... to run an analysis tool *(see Chapter on Aquator Analyzers).*
- **Tools Macro**... to run or record a macro, or show the Visual Basic for Applications Integrated Development Environment (VBA IDE) *(see p 84)*.

Aquator Toolbar

This provides the following shortcut buttons for commonly used program actions:

 **New** Project *(see p. 165)*.

 **Open** a Project *(see p. 168)*.

 **Save** current Project to Database.

 **Print** current node or all checked project nodes *(p. 168)*.

 **Setup Model** Parameters, States etc. *(see p. 161)*.

 **Setup Diagnostics and Trace** options *(see p. 152)*.

 **Setup Keyboard** accelerator keys *(p. 159)*.

 **Setup Toolbox** *(see p. 176)*.

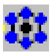
 **Setup Options** for Aquator *(see p. 168)*.

 **Chart variables** *(see p. 190)*

 **Group** user defined edit group *(see p. 64)*.

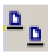
 **Report** results *(see Chapter on Reporters)*.

 **Import** data to Time Series or Profile Nodes.

 **Export** data to Time Series or Profile Nodes.

 **Analyze** current project *(see Chapter on Analyzers)*.

 **Macro** run *Microsoft® Visual Basic® Editor* *(see p 84)*.

 Display **Project Node** attributes *(see p. 106)*.

 Display **Schematic Node** *(see p. 107)*.

 Display **Top Level Time Series Node** *(see p. 115)*.

 Display **Top Level Profiles Node** *(see p. 115)*.

 **Lock** the location of all components on the schematic.

 **Run** the model from start date to finish date *(see p 123)*.

Run To a specified date *(see p. 161)*.

Pause the model run.

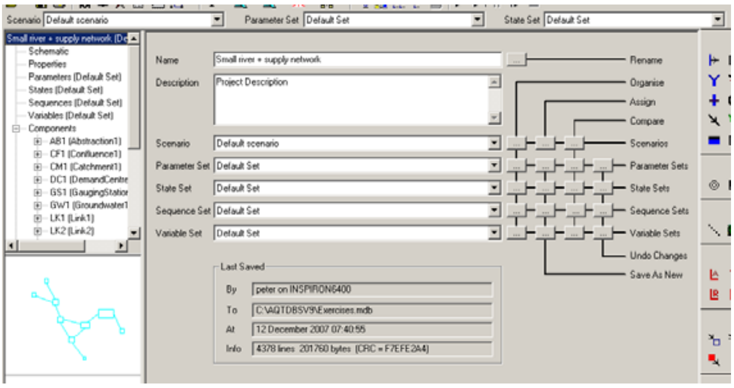
Step the model run one day.

Stop the model run.

Aquator Tree View Pane

Every node in the Tree View Pane will present data in the Aquator Information Pane.

Project Node



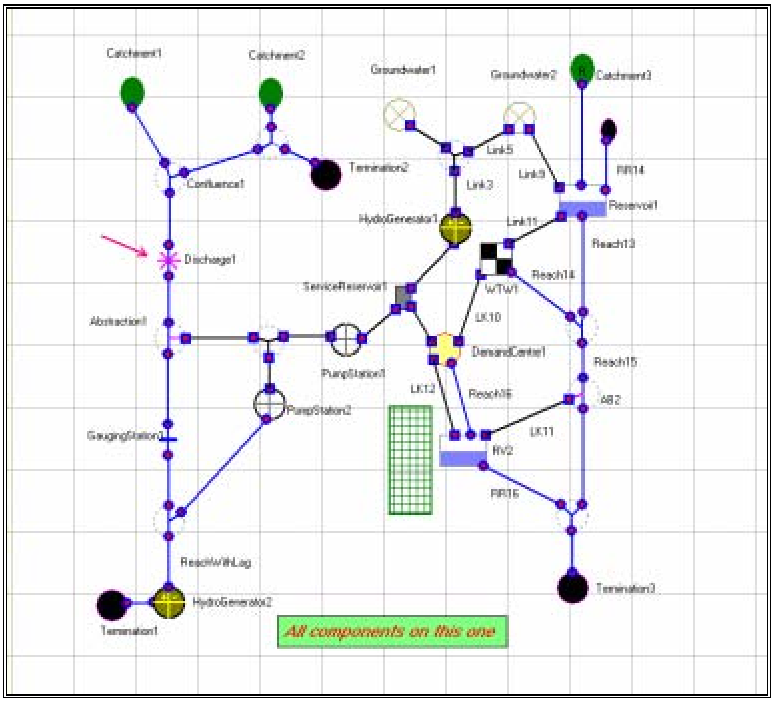
The top-level node in the project tree view shows:

- the project name, which can be changed by clicking on the button adjacent to the box showing the current name,
- project description which contains some arbitrary text which gives any additional required information; type directly into the box holding the description,
- the current *scenario;* click on the drop-down list to select a different scenario as the currently active one; scenarios may be added and removed using the buttons adjacent to the drop-down list,
- the current *parameter set*; click on the drop-down list to select a different parameter set as the currently active one; parameter sets may be added and removed using the buttons adjacent to the drop-down list,
- the current *state set*; click on the drop-down list to select a different state set as the currently active one; state sets may be added and removed using the buttons adjacent to the drop-down list,

- the current *variable set*; click on the drop-down list to select a different variable set as the currently active one; variable sets may be added and removed using the buttons adjacent to the drop-down list,
- the current *sequence set*; click on the drop-down list to select a different sequence set as the currently active one; sequence sets may be added and removed using the buttons adjacent to the drop-down list,
- some read-only information about the state of the project in the database.

See *Scenarios and Sets* for more information.

Schematic Node



The schematic representation of the project shows all of the components that have been added to the model. You can:

- drag and drop additional components from the Aquator Toolbox onto the schematic,
- right-click on any component, or on the background, to bring up a context sensitive menu of additional commands,

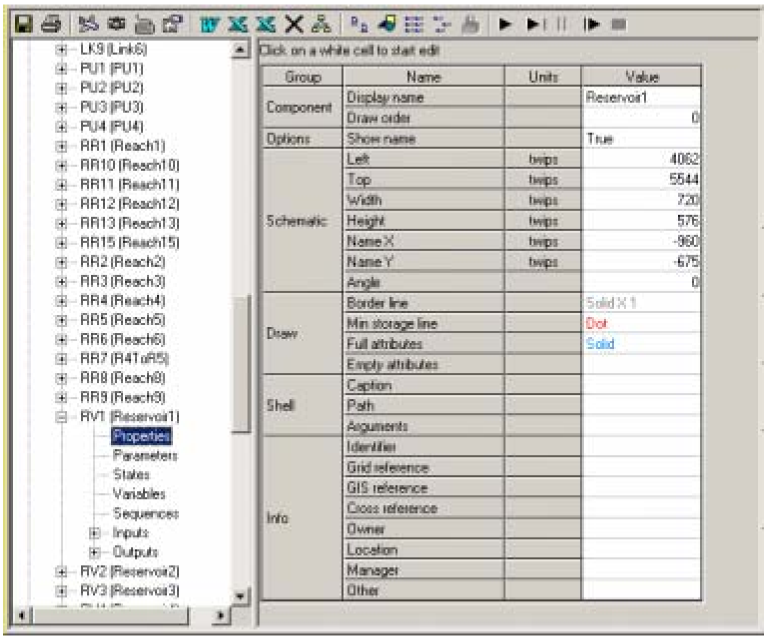
- scroll horizontally and vertically either by using the scroll bars or the keys assigned to the scroll function (by default these are the cursor keys),
- zoom in and out either by clicking on the magnify image in the lower-right corner or by pressing the keys assigned for zooming (by default these are the Page Up and Page Down keys),
- click on a component to select it; hold the shift or control keys down to select multiple components; or drag a rubber band box by clicking on the background and moving the mouse while holding the left mouse button down,
- move multiple components by selecting them and then dragging with the left mouse button held down.

When a model run is executed the schematic can optionally show the state of the system each day by:

- showing the value of one variable for each component; to select which variable is shown go to the Variables node of the component, or right-click on the component and choose the *variables* menu item; in the resulting dialog click in the *Flags* column for the variable and check *Display value on schematic during model run* in the dialog that appears,
- allowing some components to animate themselves; for instance reservoirs appear to fill and release, demand centres change colour when their demand is not met.

Note: these animation options slow execution significantly.

Properties Node



There is a *Property Node* for the project and each component in the project.

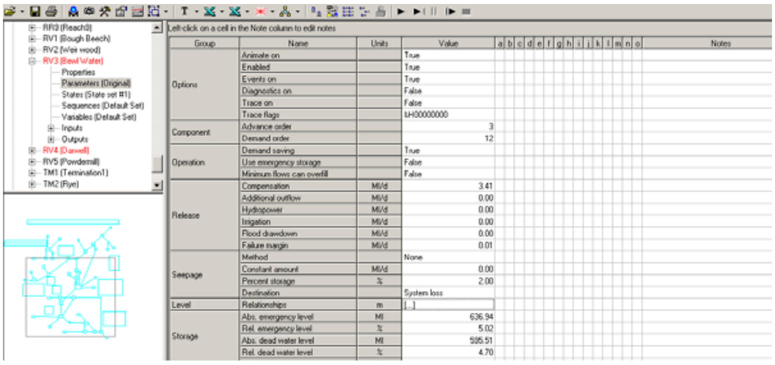
Each property has one unique value. The only way to change a property is to explicitly change an individual value i.e. properties do not come in sets and so there is no facility for choosing a property set which implicitly changes property values.

To set any Property for a Component or the Schematic, either select the corresponding Properties Node in the Aquator Tree View Pane or Right Click on the Component in the Schematic and select *Properties* from the drop down menu.

The corresponding value may be edited in the Properties Window, see *General User* Interface Rules *p.97*.

For a detailed description of the Properties available to the different types of Component see the Chapter on **Aquator Components**

Parameters Node



There is a *Parameter Node* for the project and each component of the project.

Click in a white cell under the *Value* column to edit the value. Click in a cell under the columns headed *a* to *o* to add or remove the parameter from the corresponding user-defined editing group.

Click in a white cell under the *Notes* column to enter arbitrary textual information that is to be stored along with the parameter value.

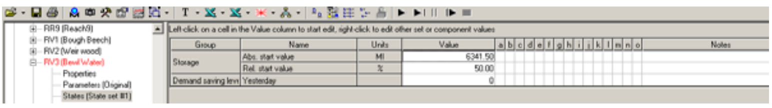
A complex model may have hundreds or even thousands of parameters. Parameters are organised into *parameter sets* which allow you to switch easily and securely between different, validated, sets of parameter values.

To set any Parameter for a Component or the Schematic, either select the corresponding Parameters Node in the Aquator Tree View Pane or Right Click on the Component in the Schematic and select *Parameters* from the drop down menu.

The corresponding value may be edited in the Parameters Window, see *General User* Interface Rules *p.97*. Click in a white cell under the *Value* column to edit the value. Click in a cell under the columns headed *a* to *o* to add or remove the parameter from the corresponding user-defined editing group.

For a detailed description of the Parameters available to the different types of Component see the Chapter on **Aquator Components**.

States Node



There is a *State Node* for the project and each component of the project.

Click in a white cell under the *Value* column to edit the value. Click in a cell under the columns headed *a* to *o* to add or remove the parameter from the corresponding user-defined editing group.

Click in a white cell under the *Notes* column to enter arbitrary textual information that is to be stored along with the state value.

A complex model may have hundreds of states. States are organised into *state sets* which allow you to switch easily and securely between different, validated, sets of state values.

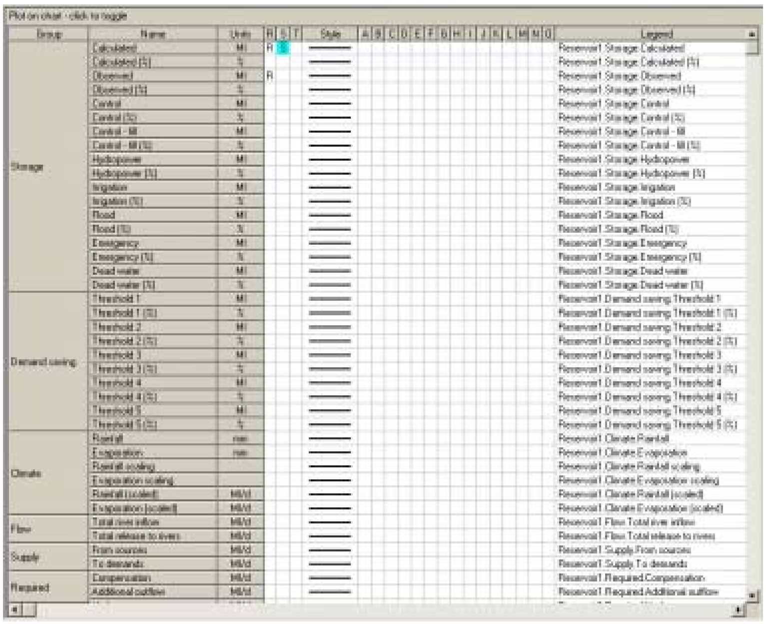
At the end of a run and/or at one specified date during a run, the entire state of the model and all its components can be optionally captured into a new state set. This will be automatically named using the original state set name and the date on which the state was captured. To set this up go to the *States* tab of the dialog that appears when you choose the *Setup...Model* menu command, or go to the *States node* of the project.

To set any State for a Component or the Schematic, either select the corresponding States Node in the Aquator Tree View Pane or Right Click on the Component in the Schematic and select *States* from the drop down menu.

The corresponding value may be edited in the States Window, see *General User* Interface Rules *p.97*. Click in a white cell under the *Value* column to edit the value. Click in a cell under the columns headed *a* to *o* to add or remove the state value from the corresponding user-defined editing group.

For a detailed description of the States available to the different types of Component see the Chapter on **Aquator Components**.

Variables Node



There is a *Variable Node* for the project and each component of the project.

Since variable *values* are set during the model run it follows that it is the behaviour of the variable that is set up in the variables node. Specifically:

- clicking in the *Style* column of the grid allows the plotting attributes of the variable to be set; if the variable is plotted on more than one chart then the same plot attributes are always used (see *Line Attributes Setup Dialog*, *p. 194*),
- clicking in either the *R S* or *T* column allows you to select the variables behaviour during a model run:

  R*:* to capture the **Result** of the variable each day of a model run; this must be enabled to plot the variable etc., but is optional in order to reduce the potentially very large amount of redundant data that would otherwise be generated,

  S: to display the value on the **Schematic** during a model run; this option is signalled on the grid by colouring the appropriate cell **Cyan**; only one variable can be so animated; animation slows execution and can be turned off by a project parameter setting,

  T: to import into the database as a **Time** series the variable values captured during a model run; if enabled this will happen automatically and the name used for the new time series will be constructed from the component name and the variable name; an existing item series of that name will be silently overwritten each time the model runs; this facility allows you to import e.g. calculated flows back into the database for analysis or further modelling,

- clicking in one of the Plot columns (*A to O*) of the grid allows you to choose which charts the variable will be plotted on at the end of a run (see: *Charts Setup Dialog, p. 192*),
- clicking in the *Legend* column of the grid allows the plotting legend to be changed from the dafault.

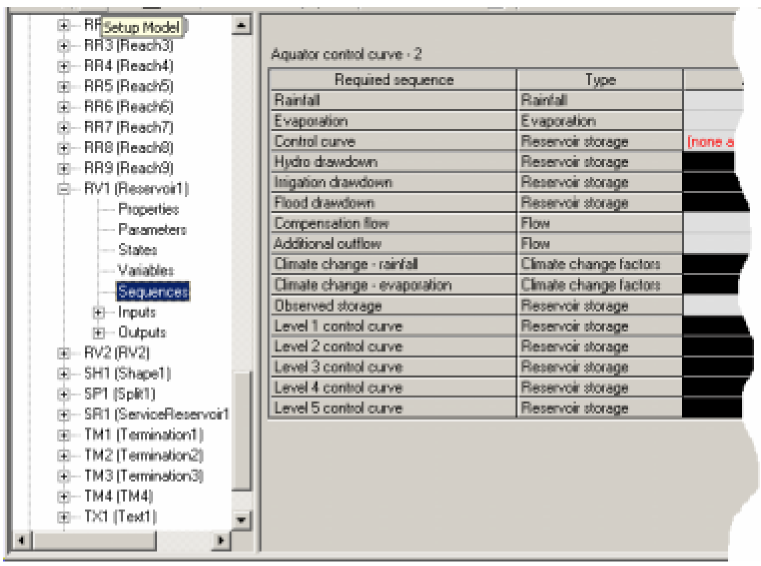
A complex model may have hundreds or even thousands of variables. Variables are organised into *variable sets* which allow you to switch easily and securely between different, validated, sets of variable properties. For example, an appropriate choice of variable setup for prediction purposes, another for calibration, and a third for climate modelling.

To set any Variable for a Component or the Schematic, either select the corresponding Variables Node in the Aquator Tree View Pane or Right Click on the Component in the Schematic and select *Variables* from the drop down menu.

The corresponding value may be edited in the Variables Window, see *General User* Interface Rules *p.97*.

For a detailed description of the Variables available to the different types of Component see the Chapter on **Aquator Components**.

Sequences Node



There is a *Sequence Node* for the project and many of the components of the project (not all types of component can have sequence data). This node holds all the information for a component's sequences collection.

A sequence requirement is satisfied by making a link to either a *Time Series* or a *Profile* or both. Time series and profiles exist independently of components and can be linked to more than one component simultaneously.

During a model run sequence data is supplied as:

- the time series value if a time series has been linked and the required value for the day is not marked as missing
- use the profile value if a profile has been linked and the required value for the day is not marked as missing
- else zero

To link time series and profiles to a sequence requirement go to the Sequences node, or right-click on the component and choose the *Sequences...* menu command.

In the sequences grid the cells in the columns marked *Time Series* and *Profiles* are coloured as described in *General User Interface Rules**, p. 97*.

Making a link is a two-step process designed to overcome the problem of finding the required time series or profile in a database that can hold thousands of time series and profiles.

When first setting up the component *right*-click on a cell to choose a subset of all the available time series or profiles of the required data type. This can be as few as one or as many as all of them.

Subsequently *left*-click on a cell to choose one time series or profile from the subset. Thus changing the link even on a daily basis becomes straightforward once the subset has been constructed.

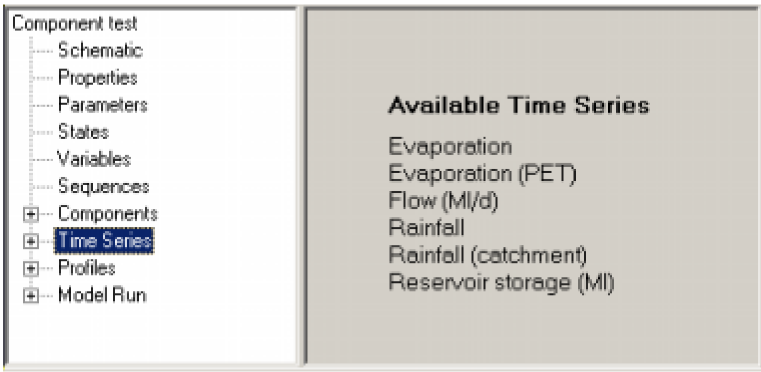
The current version of Aquator allows you to choose a time series or profile whose data type is compatible with the data type required by the sequence i.e. the conversion factor is unity. Later versions of Aquator will silently transform data type as required.

To set the Sequences for any Component or the Schematic, either select the corresponding node in the Aquator Tree View Pane or Right Click on the Component in the Schematic and select *Sequences* from the drop down menu.

The corresponding value may be edited in the Sequences Window, see *General User* Interface Rules *p.97*.

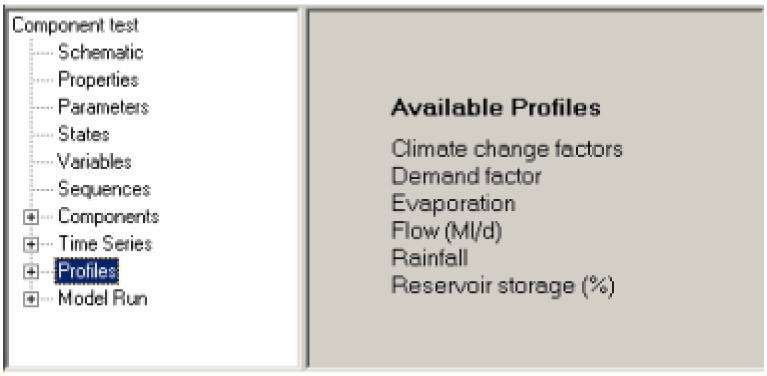
For a detailed description of the Sequences available to the different types of Component see the Chapter on **Aquator Components**.

Top Level Time Series Node



This node acts as a parent node, holding child nodes that represent each time series in the database.

Top Level Profiles Node

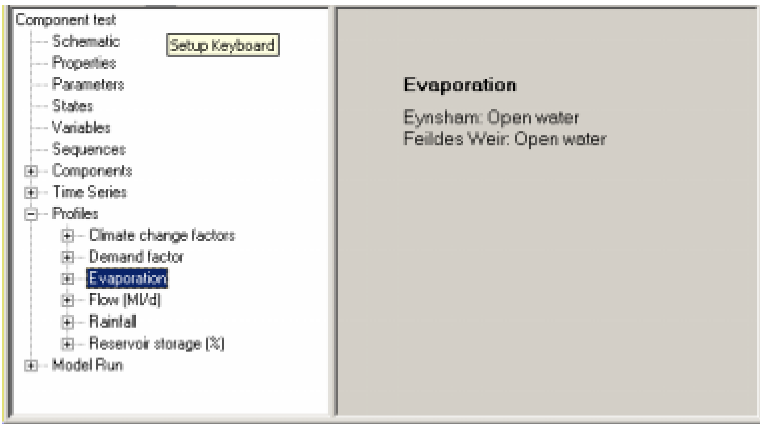


This node acts as a parent node, holding child nodes that represent each time series in the database.

Time Series Parent Node

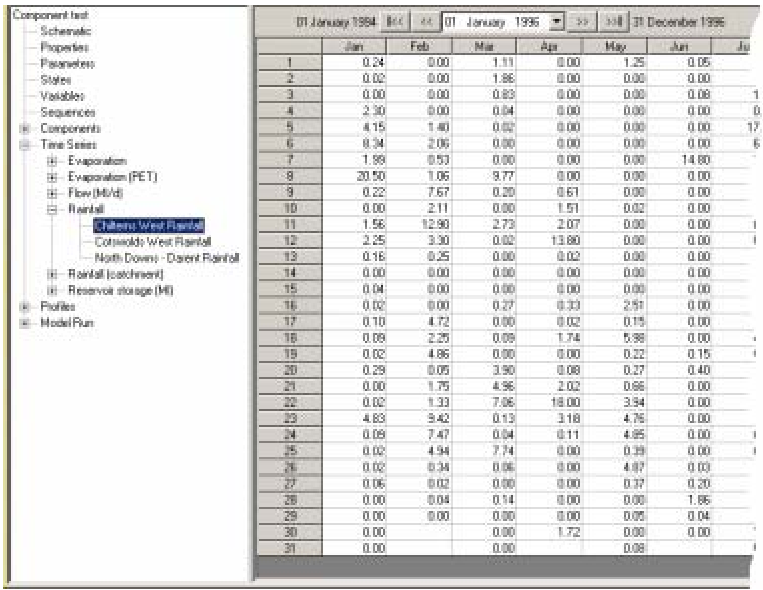
This node acts as a parent node, holding child nodes that represent each time series in the database of a particular data type.

Profiles Parent Node



This node acts as a parent node, holding child nodes that represent each profile in the database of a particular data type

Time Series Node



This node holds all the information associated with one **Time Series**.

Time series can be daily or monthly. In the latter case 12 *daily* values are used in turn for each day of the 12 months in a year.

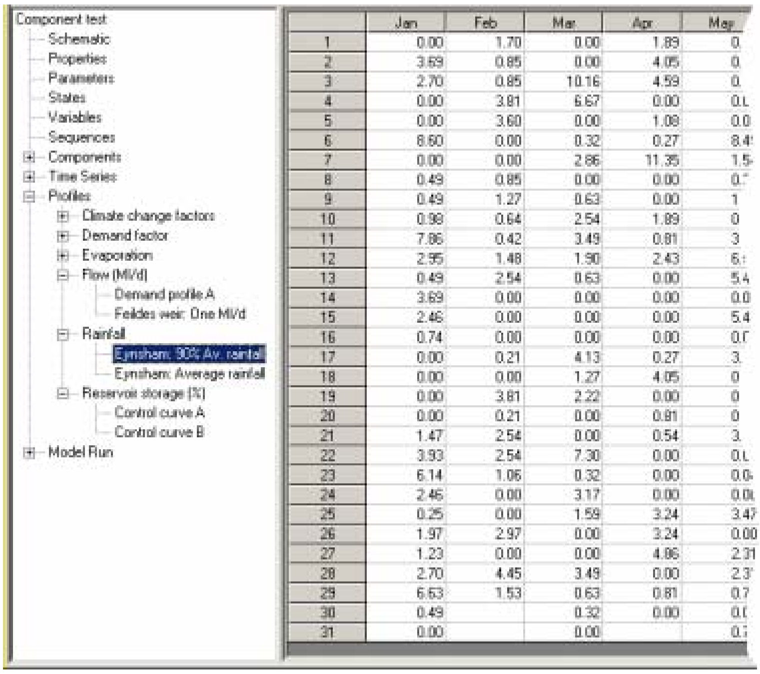
The available data can be viewed by clicking on the buttons at the top of the information pane to scroll to different dates. One year of data is shown at a time.

Time series can be linked to components that require data during a model run by going to the *Sequences node* of the component, or by right-clicking on the component in the schematic view and choosing the *Sequences* menu item.

To import or export data use the Importer or Exporter commands available on the Tools menu.

Click on the *Chart* button to plot the data.

Profile Node



	Jan	Feb	Mar	Apr	May
1	0.00	1.70	0.00	1.89	0.
2	3.69	0.85	0.00	4.05	0.
3	2.70	0.85	10.16	4.59	0.
4	0.00	3.81	6.67	0.00	0.
5	0.00	3.60	0.00	1.08	0.0
6	8.60	0.00	0.32	0.27	8.4
7	0.00	0.00	2.86	11.35	1.5
8	0.49	0.85	0.00	0.00	0.
9	0.49	1.27	0.63	0.00	1
10	0.98	0.64	2.54	1.89	0
11	7.86	0.42	3.49	0.81	3
12	2.95	1.48	1.90	2.43	6.
13	0.49	2.54	0.63	0.00	5.4
14	3.69	0.00	0.00	0.00	0.0
15	2.46	0.00	0.00	0.00	5.4
16	0.74	0.00	0.00	0.00	0.
17	0.00	0.21	4.13	0.27	3.
18	0.00	0.00	1.27	4.05	0
19	0.00	3.81	2.22	0.00	0
20	0.00	0.21	0.00	0.81	0
21	1.47	2.54	0.00	0.54	3.
22	3.93	2.54	7.30	0.00	0.
23	6.14	1.06	0.32	0.00	0.0
24	2.46	0.00	3.17	0.00	0.0
25	0.25	0.00	1.59	3.24	3.47
26	1.97	2.97	0.00	3.24	0.00
27	1.23	0.00	0.00	4.86	2.31
28	2.70	4.45	3.49	0.00	2.3
29	6.63	1.53	0.63	0.81	0.7
30	0.49		0.32	0.00	0.
31	0.00		0.00		0.

This node holds all the information associated with one **Profile**.

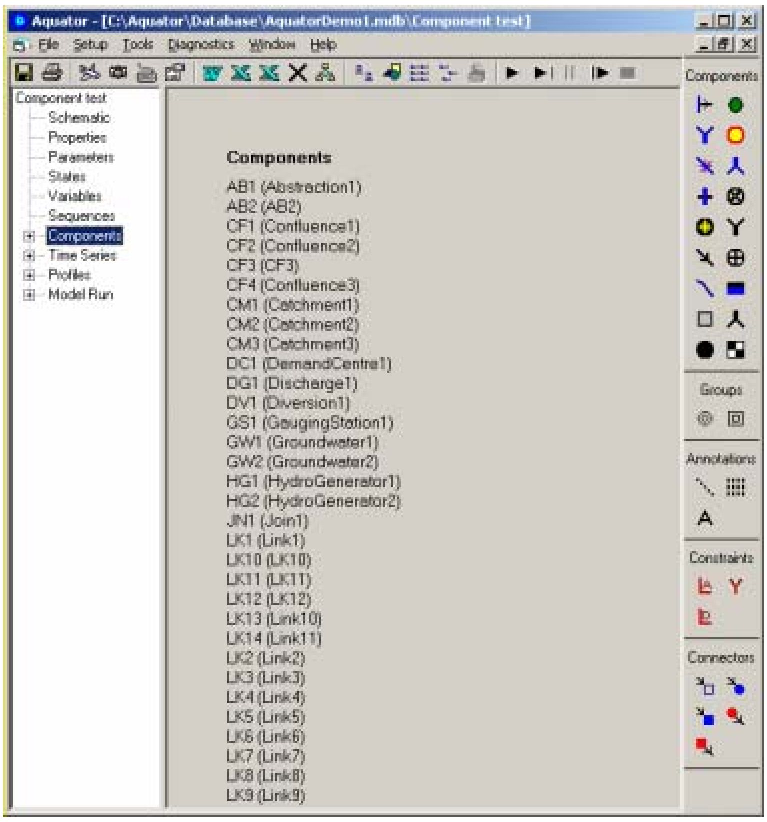
Profiles can be daily or monthly. In the former case the profile always consists of 366 values with the value for 29 February not being used in non-leap years. In the latter case 12 *daily* values are used in turn for each day of the 12 months in a year.

Profiles can be linked to components that require data during a model run by going to the *Sequences node* of the component, or by right clicking on the component in the schematic view and choosing the *Sequences* menu item. During a model run the profile values are re-used each year.

To import or export data use the Importer or Exporter commands available on the Tools menu.

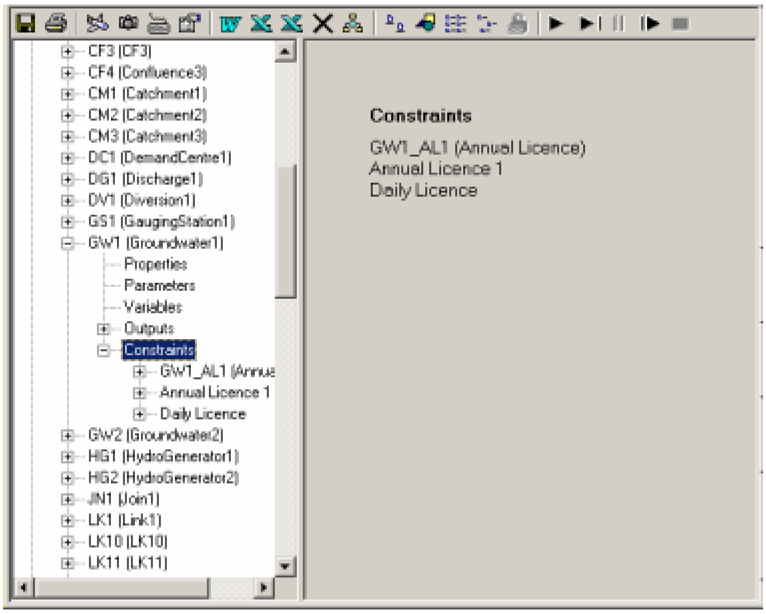
Click on the *Chart* button to plot the data.

Components Node



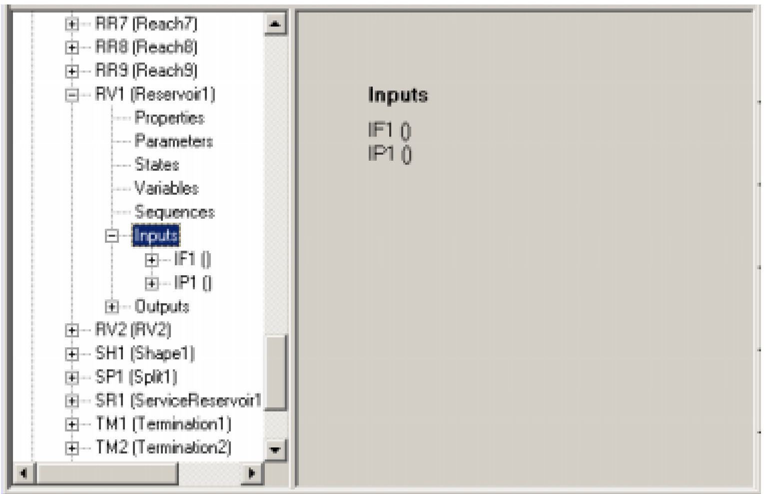
This node acts as a parent node, holding child nodes that represent each component in the project. A new component node is created each time a component is dragged from the Aquator Toolbox and dropped on the schematic.

Constraints Node



This node acts as a parent node, holding child nodes that represent each constraint that has been added to the parent component. To add a new constraint right-click on the component in the schematic view and choose the *Constraints...* menu item. Alternatively drag a constraint tool from the Aquator Toolbox and drop it on the component.

Inputs Node



This node acts as a parent node, holding child nodes that represent each input connector of the parent component. Adding a new input to a component requires consideration of the following actions and limitations.

To add an input connector of the type used by the component as a default input connector:

- right-click on a component, which may or may not show an active *Add input* context menu command; if it is unavailable the component does not allow additional default inputs to be added; if available simply click on the menu command to add an input of the default type
- alternatively, drag an input connector from the Aquator Toolbox and drop it on the component; if it is acceptable the *Stop* cursor will change to the icon representing the connector when the mouse hovers over a component that allows inputs to be added; release the mouse button to add the input connector

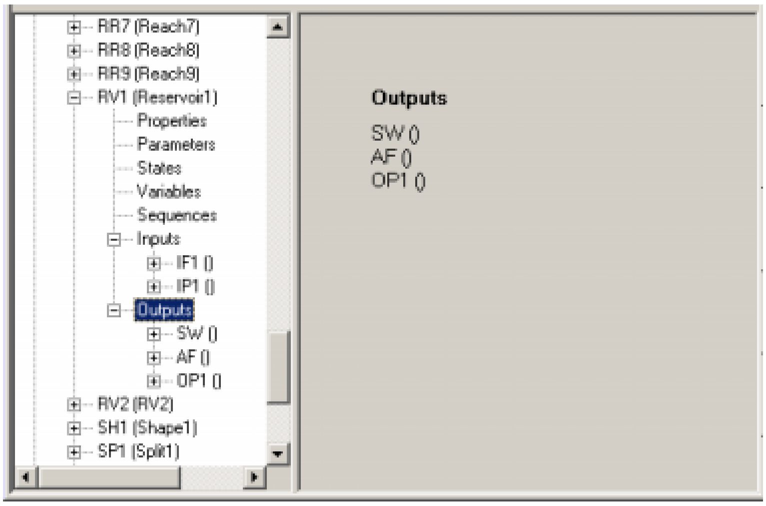
There are *two* fundamentally different types of connector – flow and supply connectors – each of which comes as input or output. By default inputs are coloured blue.

Flow connectors are used by natural objects such as river reaches while supply connectors are used by man-made objects such as pipes. Each component will at most allow one type to added by default.

A component may or may not allow additional inputs of the other type. If so, additional items will appear on the right-click context menu for that component. For example, a reservoir component allows supply-type inputs to be added by default but it also allows flow-type inputs to added to accept additional river inflows.

Note: demand centres have special input connectors that allow the specification of a minimum demand via that connector only. This has a slightly different icon in the Aquator Toolbox from a normal supply-type input.

Outputs Node



This node acts as a parent node, holding child nodes that represent each output connector of the parent component. Adding a new output to a component requires consideration of the following actions and limitations.

To add an output connector of the type used by the component as a default output connector:

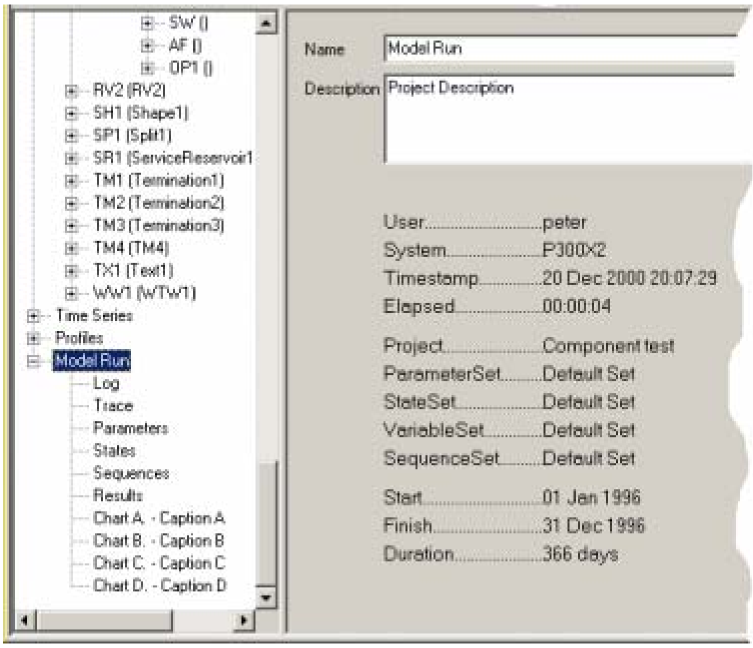
- right-click on a component, which may or may not show an active *Add output* context menu command; if it is unavailable the component does not allow additional default outputs to be added; if available simply click on the menu command to add an output of the default type
- alternatively, drag an output connector from the Aquator Toolbox and drop it on the component; if it is acceptable the *Stop* cursor will change to the icon representing the connector when the mouse hovers over a component that allows outputs to be added; release the mouse button to add the input connector

There are *two* fundamentally different types of connector – flow and supply connectors – each of which comes as input or output. By default outputs are coloured red.

Flow connectors are used by natural objects such as river reaches while supply connectors are used by man-made objects such as pipes. Each component will at most allow one type to added by default.

A component may or may not allow additional outputs of the other type. If so, additional items will appear on the right-click context menu for that component. For example, a reservoir component allows supply-type outputs to be added by default but it also allows flow-type outputs to added for special purposes

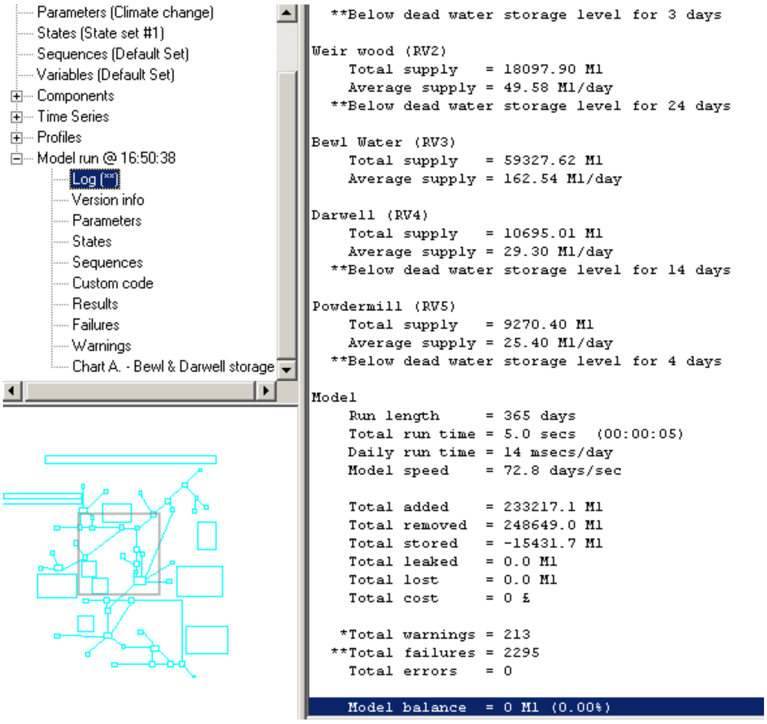
Model Run Node



Each time the model is run this parent node is created to hold all the information associated with the run. The parent node itself holds some summary information, such as who did the run and how long it took, together with a user-settable identifying name.

Child nodes hold the run log (*Model Run Log Node*), an audit trail of data fed into the model (*Model Run Data Node*), the results *(Model Run Results Node*), diagnostic information (*Model Run Trace Node*), and charting output (*Model Run Plots Node*).

Model Run Log Node



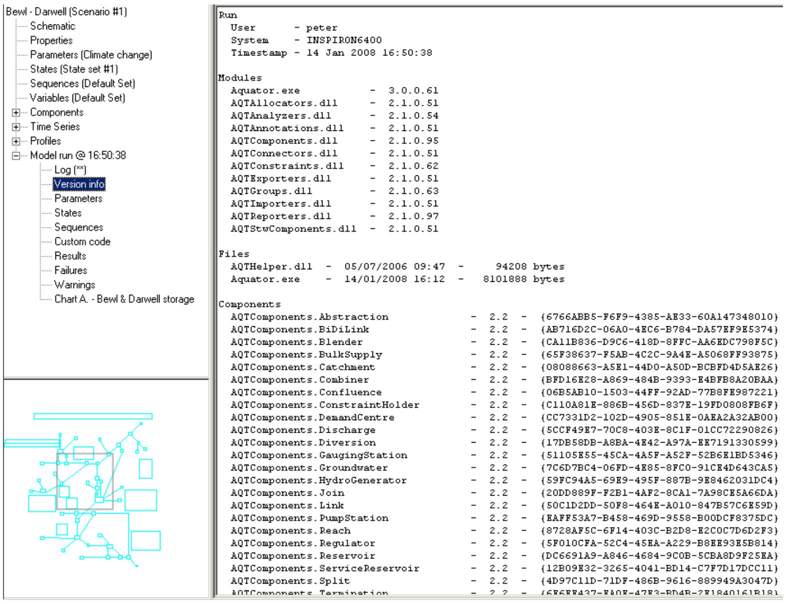
Information gathered from components and from Aquator during a model run is presented in this node.

Components are free to insert any useful information into the log. The VBA developer can use the Model property of a component to write additional information into the log.

Aquator writes some summary information at the bottom of the log showing for example the calculated water balance. This should always be zero (within floating point roundoff accuracy) showing that all movements of water have been properly accounted.

Note that warnings, failures, and errors are flagged by one, two, and three asterisks prespectively. If the number of these exceeds zero then asterisks are also added to the word "Log" in the project treeview. If a '!' character is also appended then the run terminated early, either by user intervention or because the maximum count of failures etc. was exceeded.

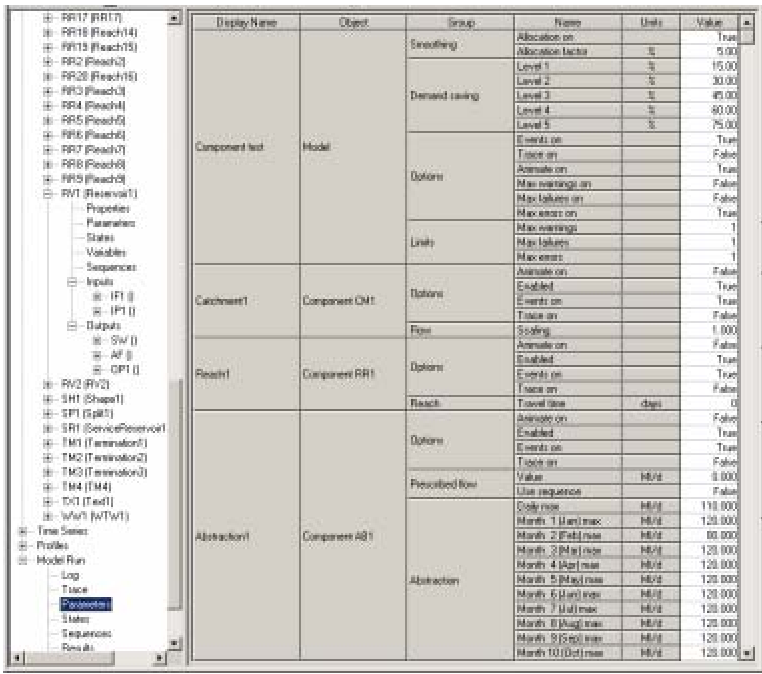
Model Run Version Info Node



This optional output node from a model run is enabled or disabled by the *Capture Version Info* option parameter and contains technical information that records the precise version of every software component used in the model run.

It is strongly recommended that this option is turned on if the model run results are to be archived for regulatory, planning, or any other important purpose.

Model Run Data Node



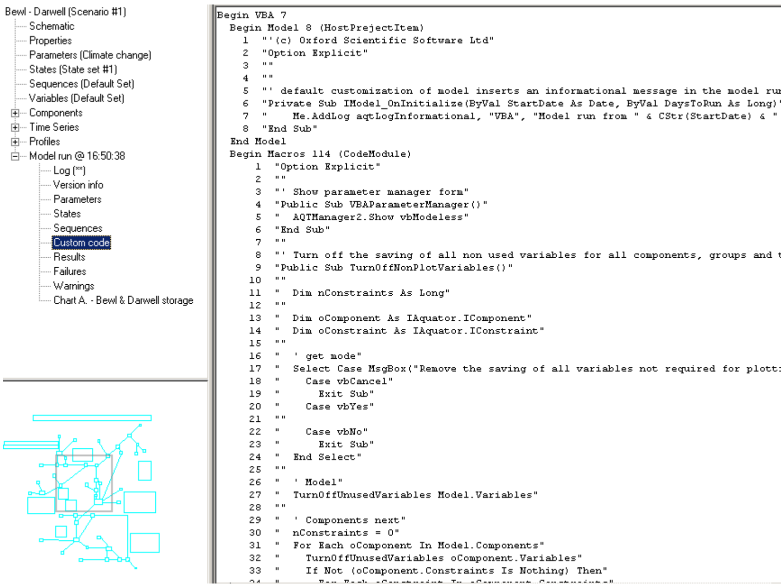
These nodes provide an audit trail of all the data that went into a model run.

The *Parameters node* collects together every parameter value used by the model and by all the components.

The *States node* collects together every state values used by the model and by all the components.

The *Sequences node* collects together every sequence link used by the model and by all the components.

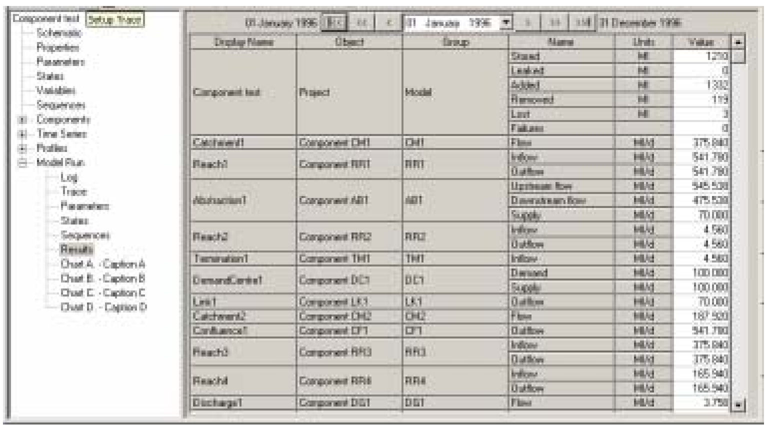
Model Run Custom Code



This optional output node from a model run is enabled or disabled by the *Capture VBA Code* option parameter and contains a copy of the custom code in the project.

It is strongly recommended that this option is turned on if the model run results are to be archived for regulatory, planning, or any other important purpose.

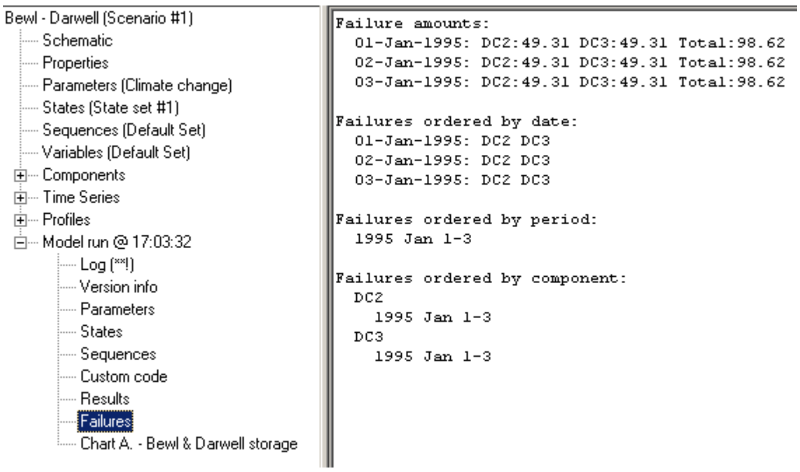
Model Run Results Node



During a model run daily data is saved from those variables marked as *Capture this variable during a model run*. By only capturing selected variables the potentially enormous memory requirements of a long run of a complex model can be reduced.

This node presents a grid of the captured variables with one column showing the values for one day.   The buttons at the top allow you to scroll backwards and forwards through the model run period.

Model Run Failures Node

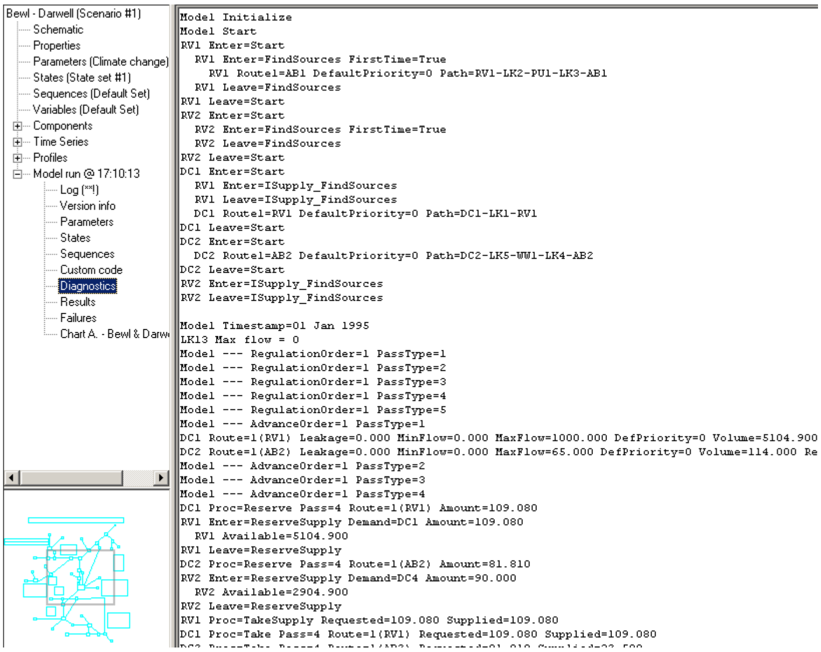


This node contains an analysis of any failures detected during a model run. Similar nodes are created for warnings and for errors.

The failures are analyzed in four ways:

- detailed daily summary, including failure amounts
- daily failures ordered by date
- periods (consecutive days) of failure ordered by date
- periods (consecutive days) of failure ordered by the failing component

Model Run Diagnostics Node



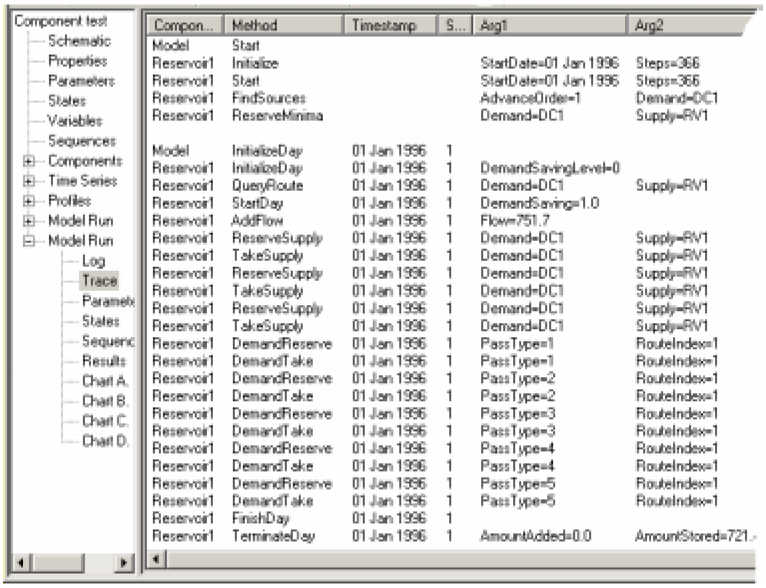
This diagnostic node is used to hold information emitted by selected components.

Diagnostic output is only generated when single-stepping. There are two stages to producing diagnostic information:

- Diagnostics must be enabled at the model or project level; either go to the project *Parameters node* and toggle *Options.Diagnostics on* or go to the *Parameters* tab of the *Setup...Model* dialog and tick the *Diagnostics* check box; the intention is that this serves as a global toggle without disturbing the other, potentially numerous, per-component settings;
- Diagnostics must be enabled for the required components; either go the *Parameters node* of the component or right-click on the component in schematic view and choose the *Parameters* menu command; toggle the *Options.Diagnostics On* parameter as required;

These tasks can be accomplished for the project and multiple components simultaneously by using the *Diagnostics and Trace dialog* raised by the *Setup...Diagnostics & Trace* menu command or clicking on the *Diagnostics & Trace* toolbar button .

Model Run Trace Node



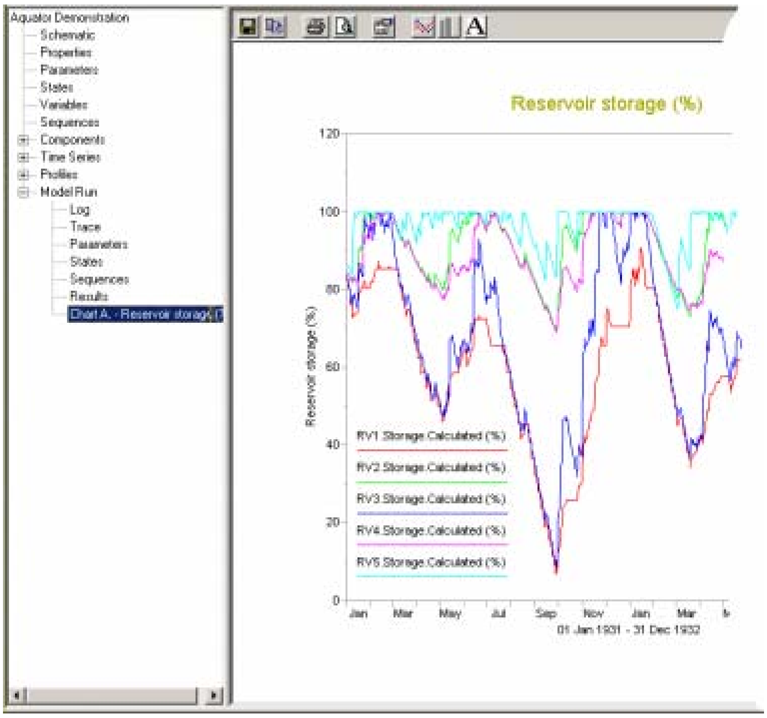
This diagnostic node is used to hold information collected by tracing the execution of selected actions by selected components, in single step mode only.

There are three stages to producing trace information:

- Tracing must be enabled at the model or project level; either go to the project *Parameters node* and toggle *Options.Trace On* or go to the *Parameters* tab of the *Setup...Model* dialog and tick the *Trace On* check box; the intention is that this serves as a global toggle without disturbing the other, potentially numerous, per-component settings;
- Tracing must be enabled for the required components; either go the *Parameters node* of the component or right-click on the component in schematic view and choose the *Parameters* menu command; toggle the *Options.Trace On* parameter as required;
- In the same dialog, click on the *Options.Trace Flags* parameter and then on the button that appears in this cell; a dialog appears which allows you to choose which actions to trace.

These tasks can be accomplished for the project and multiple components simultaneously by using the *Diagnostics and Trace dialog* raised by the *Setup...Diagnostics & Trace* menu command or clicking on the *Diagnostics & Trace* toolbar button .

Model Run Plots Node



This node holds charts produced at the end of a model run. To produce a chart select the variables whose values are to be plotted by going to the *Variables node* of a component or right-clicking on the component in schematic view and choosing the *Variables* menu command.

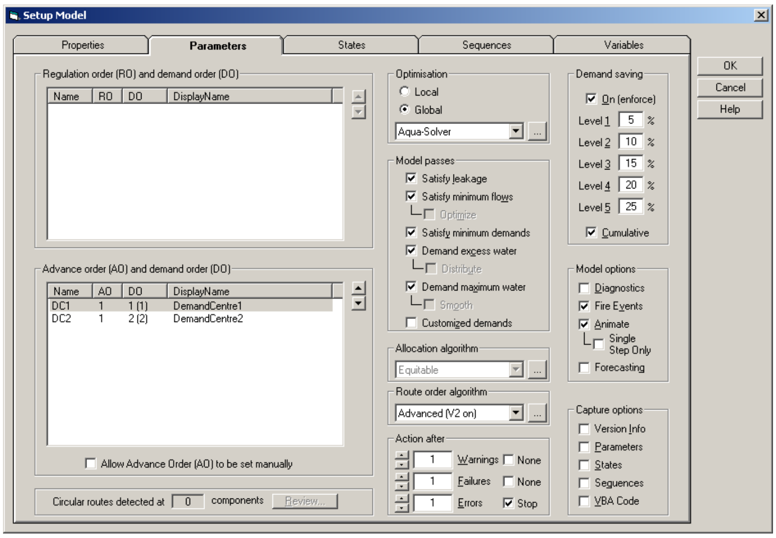
Click on a cell in the *Plots* column for a variable and choose one or more of the automatically-generated plots which are identified by letters **A**, **B**... etc. Each plot may be individually set up with a title etc.

Aqua Solver

Setup and use of Aqua Solver global optimisation

In a new project global optimisation and the *Aqua Solver* global optimiser are enabled by default. In an existing project (created by Aquator V3.0 or earlier) local optimisation is the default.

To enable or disable go to the *Parameters* tab of the *Model Setup* dialog which can be shown by clicking the *Setup Model* button on the toolbar or by using the *Setup...Model* menu item. The *Global* option button enables global optimisation (i.e. use *Aqua Solver*) and the dropdown list below it selects the global optimiser.



It is highly recommended that the *Allow Advance Order (AO) to be set manually* checkbox is unticked so that all DCs have AO = 1.

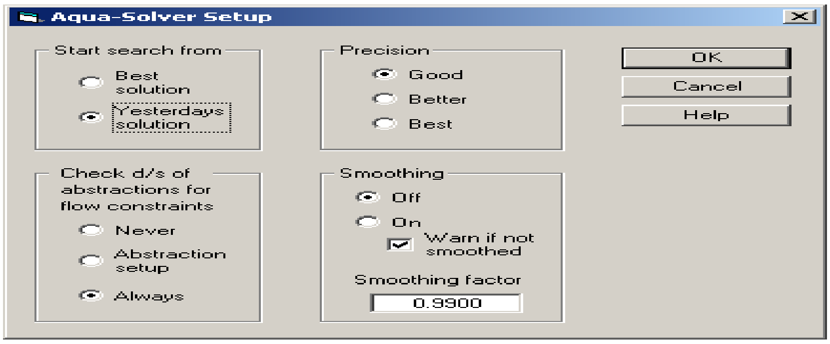
The states of the *Model passes* group of checkboxes can be independently set for the *Local* and *Global* options. In both cases *Satisfy leakage* (Pass 1) checkbox may be manually ticked, and will then execute in the usual way.

Customised demands (Pass 6) checkbox is also left enabled but in nearly every case the *Aqua Solver* solution should result in demand being satisfied (if water is available) and thus Pass 6 is superfluous. It is left enabled as a precaution: should the *Aqua Solver* solution fail to deliver water as predicted then Pass 6 could be used.

Two checkboxes in the *Model passes* group (*Satisfy minimum flows* and *Satisfy minimum demands*) are typically left checked. For the *Global* option their effect is to include the corresponding constraints in the *Aqua Solver* solution.

The final two checkboxes (*Demand excess water* and *Demand maximum water*) are also typically left checked. Their effect is to cause *Aqua Solver* to calculate the water movements that will meet demand and all constraints, while minimising cost and, if necessary, minimising the impact on resources.

The dropdown list of available global optimisers will currently only show “*Aqua Solver”*. Clicking the adjacent button will bring up a simple setup dialog for *Aqua Solver*, shown here.



The recommended settings are:

Check d/s of abstractions...:	Abstraction setup
Smoothing	Off

Then experiment with:

Smoothing:	Enable up to 3 smoothing factors
Smoothing factors:	0.99 / 0.90 / 0.50

Interaction with Advance Order and Regulation Order

The so-called *Advance Order* (AO) is the major order in which calculations are advanced and water moved to meet demand. AO = 1 is normally all demand centres, AO = 2 is the reservoirs closest to DCs refilling, etc. AO can be modified for each DC and Reservoir – see the *Model Setup dialog*.

The so-called *Regulation Order* (RO) is the major order in which regulators release water into rivers. RO can be set independently for each regulator – see the *Model Setup dialog*.

Note that, when enabled, the *Aqua Solver* runs once for each AO and once for each RO. It is strongly recommended that all DCs be set to AO=1 (this is the automatically determined default setting). In addition note that all variables associated with the *Aqua Solver* relate to AO = 1 only.

Interaction with VBA

The sole requirement for enabling *Aqua Solver* global optimiser and using VBA customisation at the same time is that all VBA code must execute before the *Aqua Solver* computes a solution.

Aqua Solver runs once for each *Advance Order* and once for each *Regulation Order*.

There are various events fired by the model which can be used to either execute VBA code or to check that VBA code in other event handlers has executed and is thus compatible with the *Aqua Solver*.

```
RiverRegulationStart(ByVal Timestamp As Date, _
                     ByVal Step As Long)
```

This is fired just before the start of moving water for the purposes of river regulation.

```
DemandsStart(ByVal Timestamp As Date, _
             ByVal Step As Long)
```

This isfired just before the start of moving water for the purposes of satisfying demand.

In general it is also completely safe to use component event handlers to alter component parameters such as:

```
BeforeInitializeDay(…)
AfterInitializeDay(…)
BeforeStartDay(…)
AfterStartDay(…)
```

These events are guaranteed to run before *Aqua Solver* first executes each day.

User-defined constraints

Using VBA it is possible to add arbitrary constraints to each optimiser solution. These constraints take one of the following forms.

$$\sum_n C_n Q_n = V$$

$$\sum_n C_n Q_n \leq V$$

$$\sum_n C_n Q_n \geq V$$

where Q_n are the quantities of water moved along each route, C_n are the user-defined coefficients (set once in VBA at the start of a model run), and V is the constraint value (set every day using VBA).

It is very important to note that constraints can be added independently for each RO (*Regulation Order*) and each AO

(*Advance Order*). To facilitate this, two separate events may be trapped to define the coefficients:

```
AddRegulationConstraint(ByVal RegulationOrder As Long,
                        ByVal ConstraintIndex As Long,
                        ConstraintName As String,
                        ConstraintType As IAquator.aqtASConstraint,
                        Coefficients As IAquator.IOptimiserCoefficients)

AddDemandConstraint(ByVal AdvanceOrder As Long,
                    ByVal ConstraintIndex As Long,
                    ConstraintName As String,
                    ConstraintType As IAquator.aqtASConstraint,
                    Coefficients As IAquator.IOptimiserCoefficients)
```

The *RegulationOrde*r and *AdvanceOrder* arguments indicate which RO or AO is being set up. The *ConstraintIndex* argument passes a value 1, 2... which increments on each call as an aid to setting up more than one constraint.

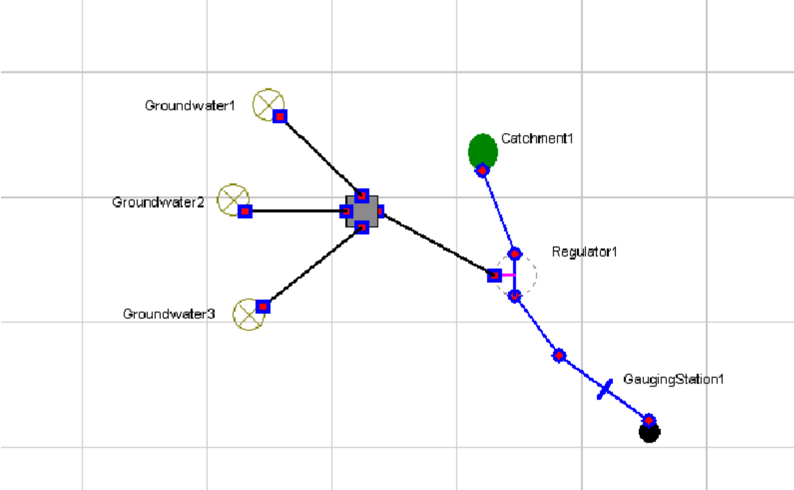
Leave *ConstraintName* blank if all constraints have been added and then your event handler will not be called again. Otherwise set to an arbitrary but globally unique (unique across all constraints for all RO's and all AO's) string. Similarly only set *ConstraintType* to one of *aqtASConstraintEQ* (equality constraint), *aqtASConstraintLE* (less than or equal constraint), or *aqtASConstraintGE* (greater than or equal constraint) if adding another constraint.

In summary: these event handlers will be called at the start of the model run for each RO and each AO. They will continue to be called, with *ConstraintIndex* being incremented each time, until you do <u>not</u> set *ConstraintName* and *ConstraintType*.

Finally set the coefficients. The *Coefficients* argument is a collection of numerical values indexed either by an integer or by a string which identifies each route i.e. each quantity of water Q_n. The string makes route indentification easy and it may be useful to temporarily show each route in the VBA IDE Immediate Pane and then copy-and-paste into code. For example in your event handler:

```
Dim oCoeff As IAquator.IOptimiserCoefficient
For Each oCoeff in Coefficients
  Debug.Print oCoeff.Route
Next
```

A simple example follows.



We want to force GW1, GW2, and GW3 to support the river in the ratio 1:1:2. Then using VBA we add two constraints.

```
Private Sub IModel_AddRegulationConstraint(ByVal RegulationOrder As Long, _
                           ByVal ConstraintIndex As Long, ConstraintName As String, _
                                 ConstraintType As IAquator.aqtASConstraint, _
                                 Coefficients As IAquator.IOptimiserCoefficients)
  If ConstraintIndex = 1 Then
    Coefficients("RL1-LK1-SR1-LK4-GW1").Value = 0
    Coefficients("RL1-LK1-SR1-LK3-GW2").Value = 1
    Coefficients("RL1-LK1-SR1-LK2-GW3").Value = -0.5
    ConstraintName = "GW3-GW2"
    ConstraintType = aqtASConstraintEQ
  ElseIf ConstraintIndex = 2 Then
    Coefficients("RL1-LK1-SR1-LK4-GW1").Value = 1
    Coefficients("RL1-LK1-SR1-LK3-GW2").Value = 0
    Coefficients("RL1-LK1-SR1-LK2-GW3").Value = -0.5
    ConstraintName = "GW3-GW1"
    ConstraintType = aqtASConstraintEQ
  End If
End Sub
```

In this simple example we want the constraint value V to be zero in each case. This is the default but for completeness we add an event handler to specifically set the value each day. Follow this pattern if you need to set non-zero values.

```
Private Sub IModel_QueryUserDefinedConstraintValue(ByVal ConstraintName As String, _
                                                   ConstraintValue As Single)
  Select Case ConstraintName
    Case "GW3-GW2"
      ConstraintValue = 0
    Case "GW3-GW1"
      ConstraintValue = 0
    Case Else
      Debug.Print "Constrint name error"
  End Select
End Sub
```

Restrictions

Certain restrictions are placed on a model which is to use the *Aqua Solver* global optimiser.

- A component which is disabled when the model run starts will never be used, even if subsequently enabled using VBA. This restriction can be overcome:

1. Enable the component before the model run starts. You will probably want to save the model in this state.
2. If the component is to be disabled on day 1 then trap the StartDayAllDone event of the Model object in VBA and disable it.
3. Subsequently, if the state of the component is to be changed from enabled to disabled or vice versa then trap the OnInitializeDay event of the Model object in VBA and toggle the enabled flag appropriately.
4. Note that you cannot trap component events if the component is disabled

- A minimum flow or maximum flow requirement which is not enforced when the model run starts will never be enforced, even if the enforce flag is subsequently set to True using VBA. This restriction can be overcome:

1. Set the enforce flag to True before the model run starts. You will probably want to save the model in this state.
2. If the flow requirement is not to be enforced on day 1 then set the enforce flag to False on day 1 using the *StartDayAllDone* event of the Model object in VBA.
3. Subsequently, if the enforce flag is to be toggled, then you can use either the *StartDayAllDone* event of the Model or the *BeforeStartDay* or *AfterStartDay* events of the component itself.

- *Diversions* with upstream *Abstraction* components and/or upstream *Regulator* components must work linearly i.e. divert a fixed percentage of the inflow. *Diversions* with no upstream components that can alter river flow except *Catchments* and R*eservoirs* (through spilling) can obey any rule.
- The *Blender* component cannot currently be used at all.
- Fixed losses e.g. in a *Reach* component may or may not cause a problem and need to be evaluated on a case-by-case basis. If the river flow always exceeds the *Reach* fixed loss then *Aqua Solver* can be used without problems.

- *Reaches* should not use an abstraction profile. Instead use an *Abstraction* component and some source of demand to get the same effect.
- If adjacent *Abstractions* on the same river are set up to allow both direct and indirect abstraction then it may be necessary to alter route priorities so that all direct abstraction is done after all indirect abstraction. This restriction is caused by *Aqua Solver* being capable of manipulating indirect abstraction to meet river flow constraints that would otherwise limit direct abstraction.
- If any Abstraction does not have a licence then it will also be necessary to give it a non-zero control flow so that the resource state is finite, otherwise route priority will not have the desired effect on Pass 5 (maximum water).
- Be aware of the timing restrictions placed on custom VBA code, as described in the previous section.

Issues to be aware of

- Removing unnecessary constraints in a model speeds up the execution time. Only have *Enforce max/min* on links etc. when really needed. Using *Aqua Solver* can reduce model execution time if only important constraints are set (by 40% in one model that has been tested).
- *Aqua Solver* tries to find the best solution it can each day in a series of attempts. The first attempt tries for a solution with no failures, resources in a good state, cheap in cost, minimum flows enforced and smooth with respect to the system flows yesterday. If this not possible the next best solution is found. This could mean dropping the minimum flow enforcement to ensure demands are met. If this is the case all minimum flow constraints are ignored.

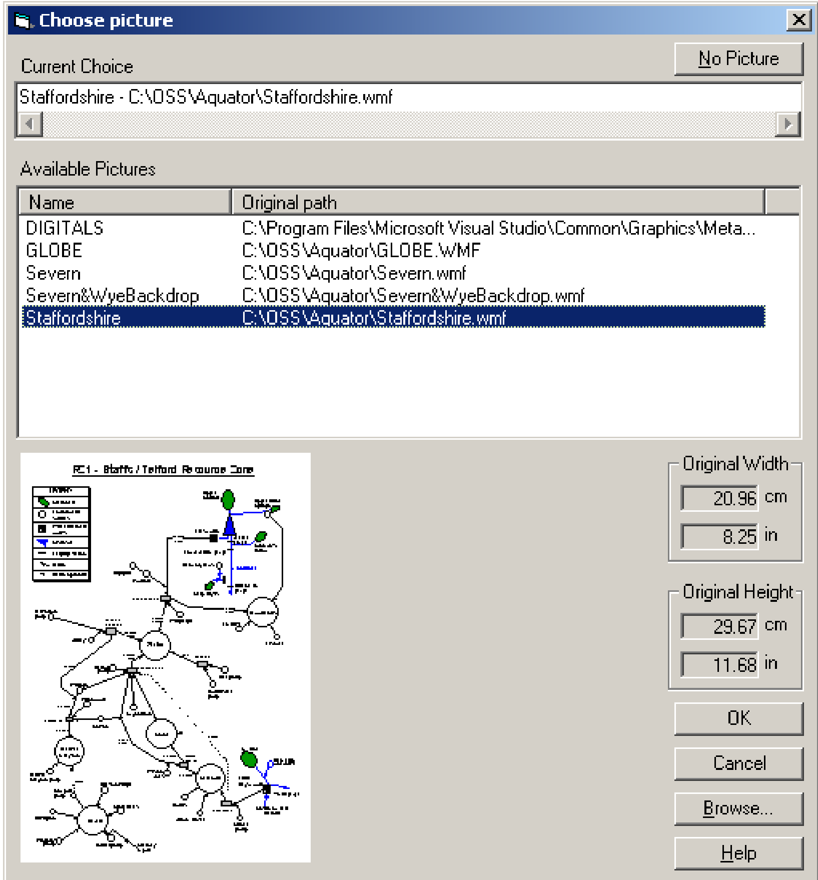
Aquator Dialogs

Aqua Solver Setup

To access this, click on the button to the left of the dropdown list in *Optimisation* of the *Parameters* tab of the *Model Setup* dialog.

See *Setup and use of Aqua Solver global optimisation*.

Choose Picture Dialog



This dialog allows you to choose a picture from those stored in the database, and optionally to add new pictures to the database. Typically this dialog appears when you want to add or remove a picture from the schematic.

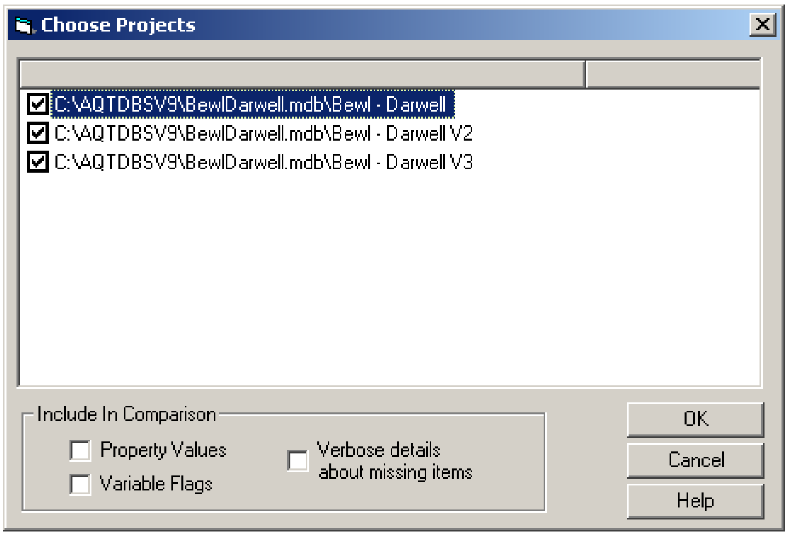
On this dialog the following controls appear:

- The *No Picture* button allows you to remove any previously selected picture
- The *Current Choice* box shows the name of the currently selected picture and the full pathname of the file where the picture was originally located (Aquator uses pictures stored in the database, it does not use the original file once the picture has been stored).
- The *Available Pictures* list show the pictures currently stored in the database
- The *Browse* button can be used to browse to new picture files which will be added to the database

- the lower portion of the dialog shows a preview of the currently selected picture in its correct aspect ratio, together with the picture's nominal width and height

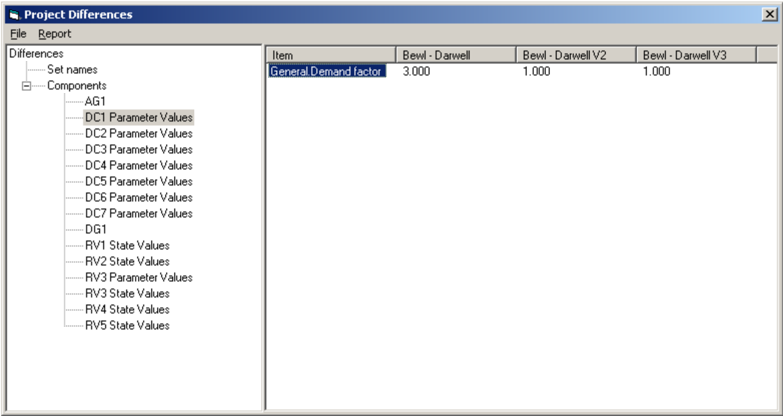
Best appearance will be achieved if the pictures are metafiles (WMF = Windows MetaFile, EMF = Enhanced MetaFile) because these formats are scaleable. Bitmap pictures (BMP files) do not reproduce well when stretched or shrunk and should be avoided except possibly for logos reproduced at their natural size.

Choose Projects To Show Differences Dialog



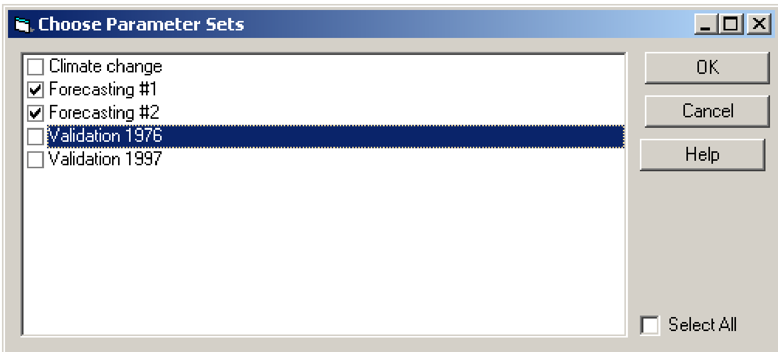
After selecting the menu item *File..Project Differences* this dialog allows you to choose up a number of projects which are then scanned for differences. Up to four projects, or the number of open projects, whichever is less, can be compared at the same time.

After choosing the projects and clicking OK a dialog similar to the following will appear, where only the differences between projects are displayed.



These differences are shown for the currently selected parameter set, state set, etc. The *File* menu contains a *Sets and Options* item which can be used to change sets using the *Choose Sets For Project Differences Dialog*.

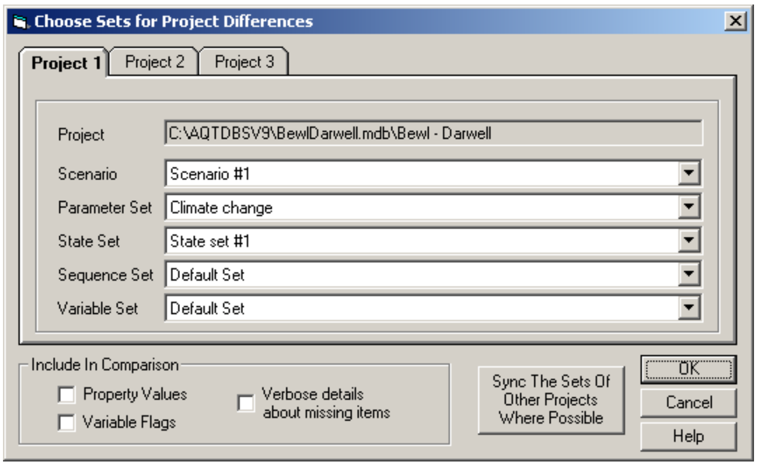
Choose Sets Dialog



The *Compare Sets* dialog includes buttons for copying values from one set to another or, if the CTRL key is held down at the same time, from one set to multiple other sets.

Then this dialog appears where you choose the sets whch are to be the target of the copy operation.

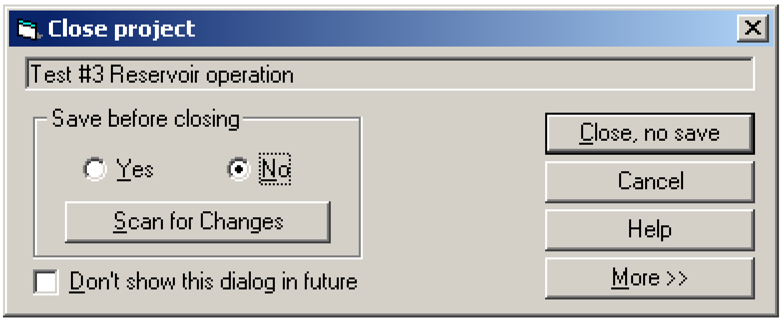
Choose Sets For Project Differences Dialog



For each project that is being compared (see the *Choose Projects To Show Differences* dialog) the scenario and/or the parameter set, state set, etc in use can be changed.

The *Sync The Sets Of Other Projects Where Possible* button will try to set the scenario and sets of the projects in the other tabs to those of the same name as the project in the current tab. In this way similar projects with similar sets can be rapidly scanned for differences.

Close Project Dialog



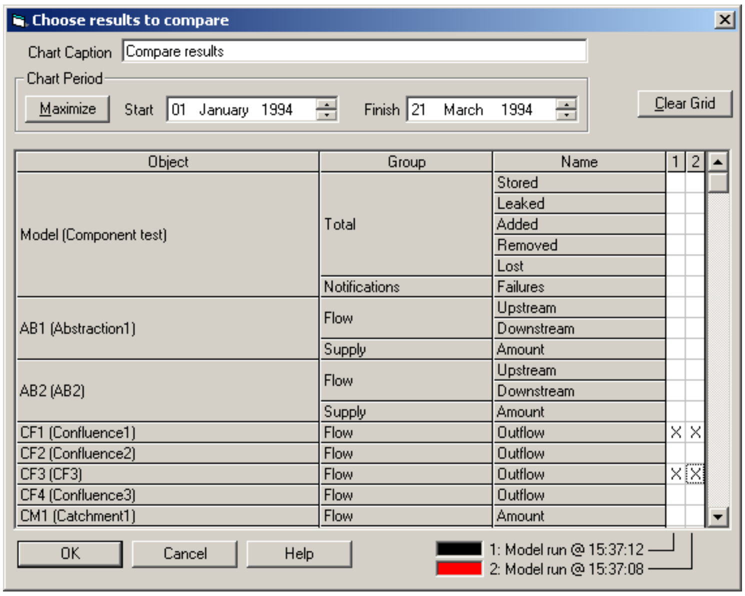
Optionally this dialog is displayed when a project is closed.

If you choose to *not* show this dialog then Aquator will always compare the project loaded into memory with the project in the database and, if there are any differences, will prompt you to save the modified project before it is finally closed. This automatic check can take several seconds with a large project.

If you choose to show the above dialog when a project is closed then there is no automatic check to see if the project has been modified. Instead you can choose to save or not save, regardless of whether there are any modifications, or you can click the *Scan* *for changes* button to initiate a comparison with the project as stored on the database.

Clicking the *More* button reveals an advanced facility where you can see what changes, if any, have been made.

Compare Results Dialog



This dialog appears when you select *Compare...* after right-clicking on the *Results* node of a *Model Run* node in the project tree view.

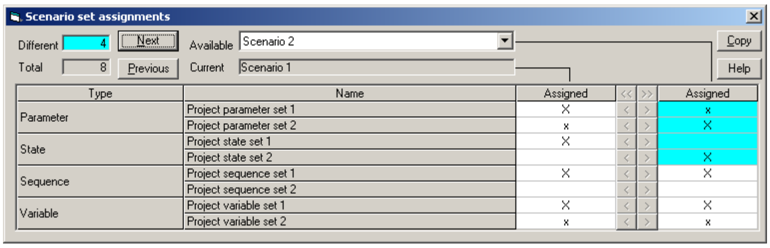
The dialog displays every variable captured for every selected model run. All model runs are selected unless check boxes are turned on in the tree view and one or more of these *Results* nodes have been checked, in which case only checked model run results are included.

By clicking in the columns headed 1, 2... (the number of such columns equals the number of model runs) you select results to compare by plotting. After making the selection click *OK* to generate the chart.

This can also be used to plot variables from a single model run that were not plotted by default when the model run finished.

There is no limit to the number of these charts that can be generated.

Compare Scenarios Dialog



The *Setup...Scenarios* menu command has 3 sub-items, the last of which (*Compare*) is used to compare two scenarios.

One scenario is always the active scenario and the other can be chosen from the dropdown list at the top.

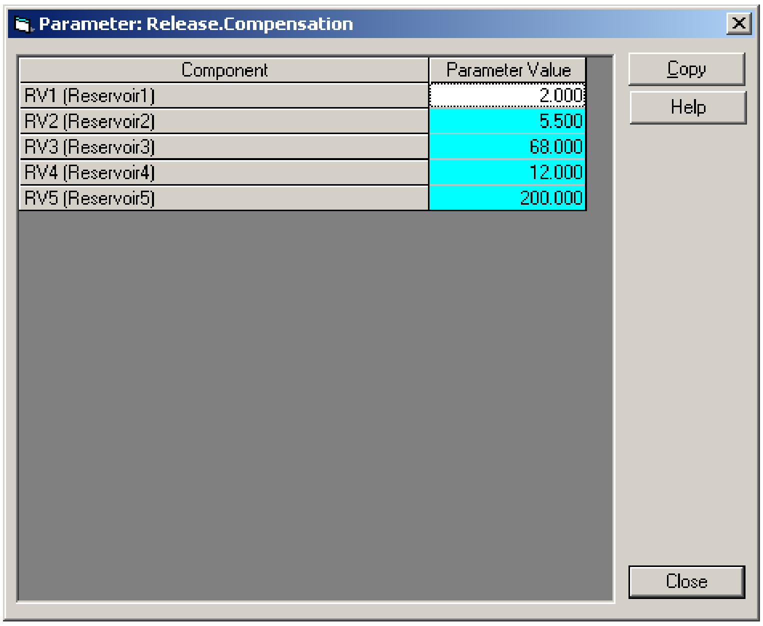
The two columns with the normally white background display the set assignments the two scenarios. Cells where the comparison scenario's assignments are different from those of the current scenario are highlighted in a different colour. The total count of differences is shown at top left.

The two narrow grid columns whose cells each contain either the < or > symbol are in fact columns of buttons. Clicking any button copies the assignment in the chosen direction.

Clicking either the << or >> buttons at the top of these columns copies every assignment in the chosen direction. In this way two scenarios can easily be synchronized.

See *Scenarios and Sets* for more information.

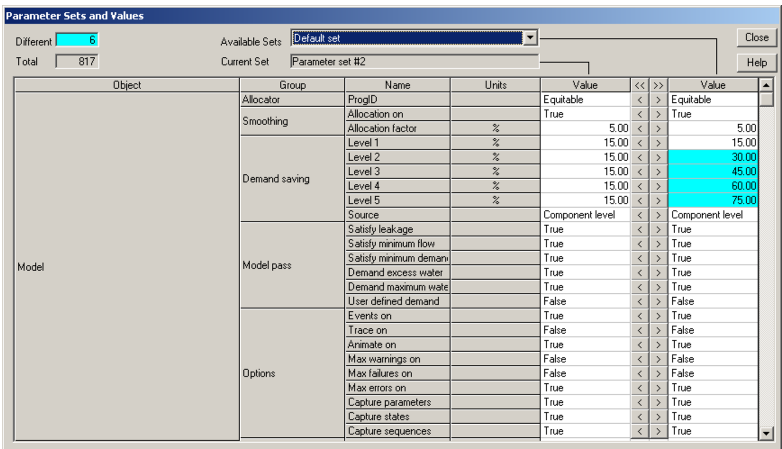
Compare Selected Components Dialog



This dialog appears as a child of the *Select Components Dialog* when the *Compare* button of the *Modify* tab is clicked.

The value of the selected item is compared across the selected components. In this example the *Release.Compensation* parameter value is compared for all reservoirs in a project with values that differ from the first highlighted. Click in a cell showing any value to choose which cell to compare the others with.

Compare Sets Dialog



The *Setup...Sets* menu command has 12 sub-items, the last four of which (*Compare Parameter, Compare State, Compare*

Sequence, and *Compare Variable*) are used to compare sets of parameters, states, sequences, and flags.

This screen shot shows the result of comparing two parameter sets. One set is always the currently active set. The second set may be chosen using the drop-down list at the top.

The two columns with the normally white background display the parameter values for the two sets. Cells where the comparison set's parameters have a different value from the current set's parameters are highlighted in a different colour. The total count of differences is shown at top left.

The two narrow grid columns whose cells each contain either the < or > symbol are in fact columns of buttons. Clicking any button copies the parameter value in the chosen direction.

Clicking either the << or >> buttons at the top of these columns copies every item in the chosen direction. In this way two sets can easily be synchronized.

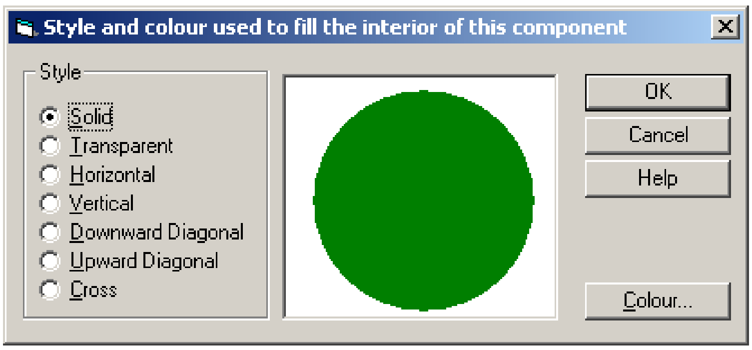
If the CTRL key is held down when clicking any of these buttons then another dialog appears which allows the value can be copied to multiple sets simultaneously. See the *Choose Sets* dialog.

Note that since every parameter in the project is included, there may be a noticeable delay while this dialog is being initalized.

The other commands that show this dialog compare state values (*Sets...Compare State* command), the time series and profiles linked to each sequence (*Sets...Compare Sequence*), and the flags controlling each variable (*Sets...Compare Variable*).

See *Scenarios and Sets* for more information

Component Fill Attribute Setup Dialog



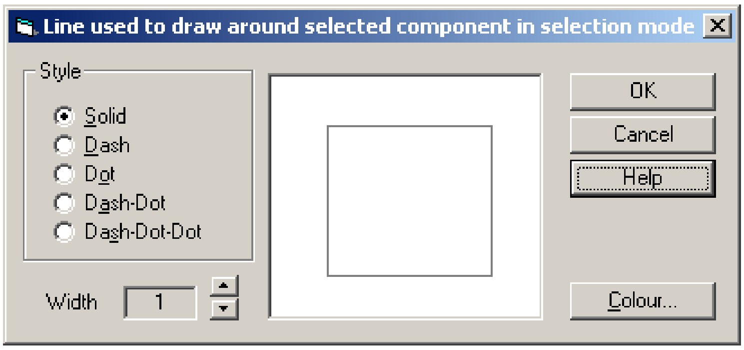
This dialog is used to choose the style and colour with which the interior of certain schematic elements is filled.

To show this dialog click on the cell in any properties grid which is displaying a preview of the fill attributes, and then click on the button which appears in the cell.

The style is chosen from the options on the left, and may be solid, transparent, or one of five different hatch patterns.

Clicking the *Colour* button will show the standard *Windows Choose Colour* dialog where the fill colour may be chosen.

Component Line Attribute Setup Dialog



This dialog is used to choose the style, width, and colour with which lines are drawn on the schematic.

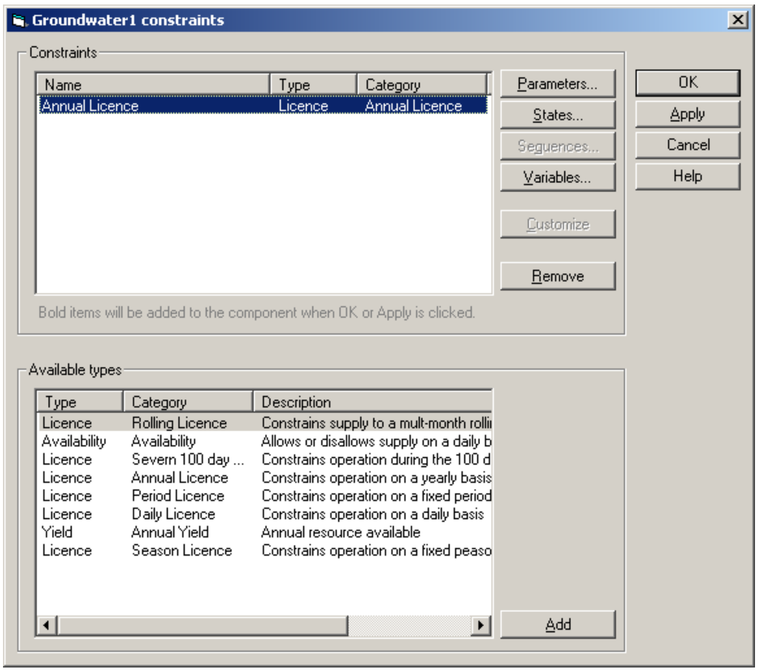
To show this dialog click on the cell in any properties grid which is displaying a preview of the line attributes, and then click on the button which appears in the cell.

The style is chosen from the options on the left, and may be solid or one of four different hatch patterns.

The width in pixels may be chosen using the up-down button at bottom-left. Widths greater than one pixel can only be shown in a solid style (this limitation of Windows improves drawing speed).

Clicking the *Colour* button will show the standard *Windows Choose Colour* dialog where the line colour may be chosen.

Constraints Dialog



This dialog appears when you right-click on a component and choose the *Constraints...* menu command. It is used to add constraints to a component.

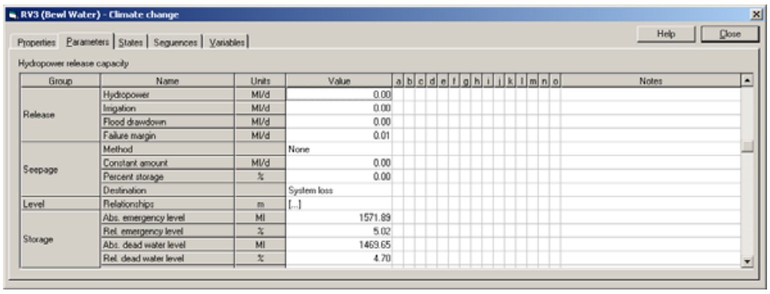
Selecting one of the available constraint types (either a licence or a yield) and clicking *Add* will add the new constraint.

Selecting one of active constraints and clicking *Remove* will remove the constraint.

Clicking in the *Name* column allows a suitable name to be assigned to any constraint e.g. "NRA licence #32567".

The buttons inside the *Constraints* group box allow you to edit the parameters, states, sequences, and variables of any constraint, where these exist, and to add the constraint to the VBA project by clicking *Customize*, but only of the component has been customized first.

Context Edit Dialog



This dialog appears when you right-click on a component in schematic view and choose the *Parameters...* menu command. Similar dialogs appear if you choose *Properties...*, *States...*, *Variables...*, or *Sequences*… In addition the tabs on this dialog enable you to navigate to any of these other editing grids without having to close the dialog first.

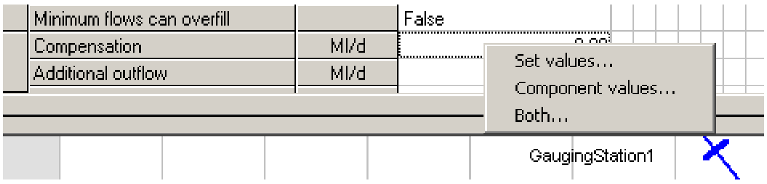
In each case you can edit the relevant values in exactly the same way as going to the corresponding nodes, but without losing the schematic view.

Using the mouse just click in any white cell to edit.Using the keyboard you move the focus rectangle to any white cell using the cursor arrow keys. Then hit F2 to edit a cell in the Value or Notes columns, or hit the space bar to toggle the setting any any cell a-o (this adds or removes the corresponding parameter to one of the user-defined editing dialogs).

When editing a cell in the Value column pressing the ESC key leaves the value unchanged and exits edit mode, otherwise the up-arrow or down-arrow keys accept the changed value and move to the next value above or below.

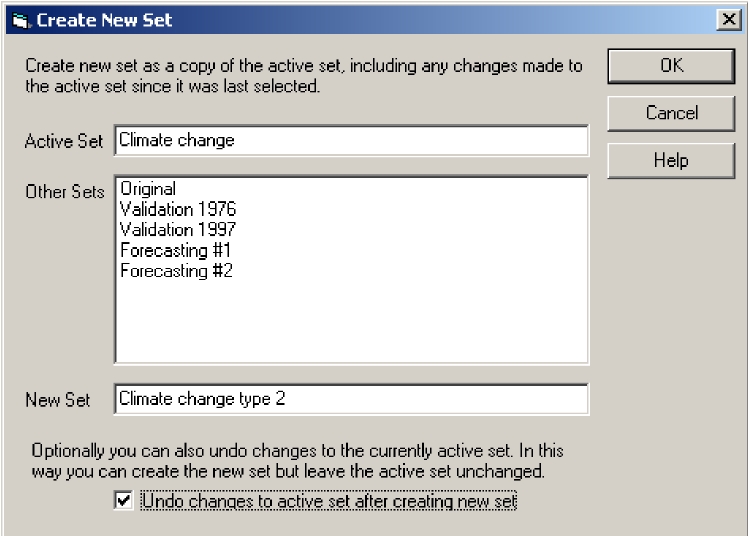
After editing the final value to be altered the key sequence RET...RET accepts the final edit and closes the dialog. Alternatively click 'Close' when finished.

Additionally right-clicking in a *Value* cell produces the following popup menu which enables editing a value across multiple sets and/or multiple components in a single operation.



Choosing one of these options shows an additional dialog, *Edit Set/Component Values Dialog*.

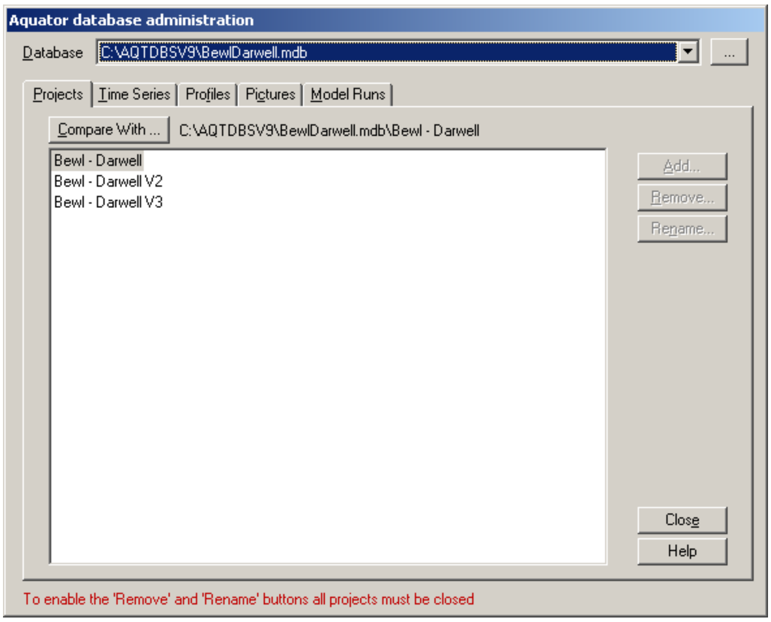
Create New Set Dialog



It can be useful to be able to save some parameter value changes without losing the original values. Use the *Create New Set* dialog for this purpose, accessed either from the *Setup...Sets...Create New...* menu items, or from the corresponding buttons visible in the topmost node of the project.

Enter the name of the new set (double-clicking existing set names pastes the corresponding string into the *New Set* textbox) and click OK. If the *Undo changes to active set after creating new set* checkbox is ticked then the active set is restored to the values it had when last selected.

Database Maintenance Dialog



This dialog appears when the *File...Database...* menu command is chosen. The *Projects* tab allows projects to be removed from the database and should be used with great caution. This option is only enabled if no projects are open.

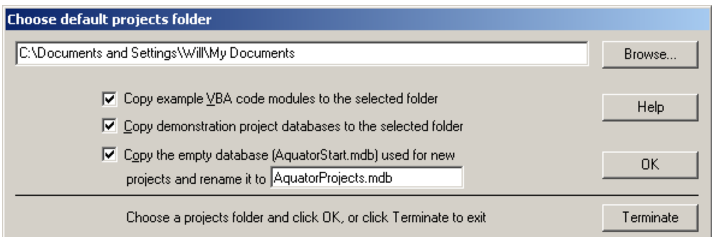
The *Compare With...* button allows any selected project to be compared with the currently active project.

The *Time Series* tab allows time series to be added *(see New Time Series or Profile Dialog)*, removed, and renamed. Similarly for the *Profiles* tab.

Pictures used to annotate the schematic are stored in the database. The *Pictures* tab allows you to add and remove pictures.

Model runs can optionally be stored in the database. The *Model Runs* tab allows you to remove previously-stored model runs.

Default Projects Folder Dialog



This dialog appears when Aquator is first run after installation, or if you choose the *File…Projects Folder* menu command, or if the previously-specified default folder is not found or you do not have permission to use the previously-specified default folder. The default folder is also used to store the error logging (ERRORLOG.*) files which are used to diagnose any problems detected while Aquator runs.

Aquator projects are stored in database (*.mdb*) files which are themselves stored on a local hard disk or network drive. Although these database files may be stored anywhere (you can browse to the required location when you open a project) it is convenient to have a default projects folder. This dialog allows you to specify the default folder to use.

Tip: if you select *Make New Folder* immediately after you have clicked *Browse…*, you may miss that you have created a new folder with name **New Folder** within **My Documents** because the standard Windows behaviour is not very clear when it does this. To rename it at this point simply right click with the mouse and select **Rename**.

On first execution you cannot proceed until you have specified the default folder. Aquator will check that you have write access permission to the selected folder.

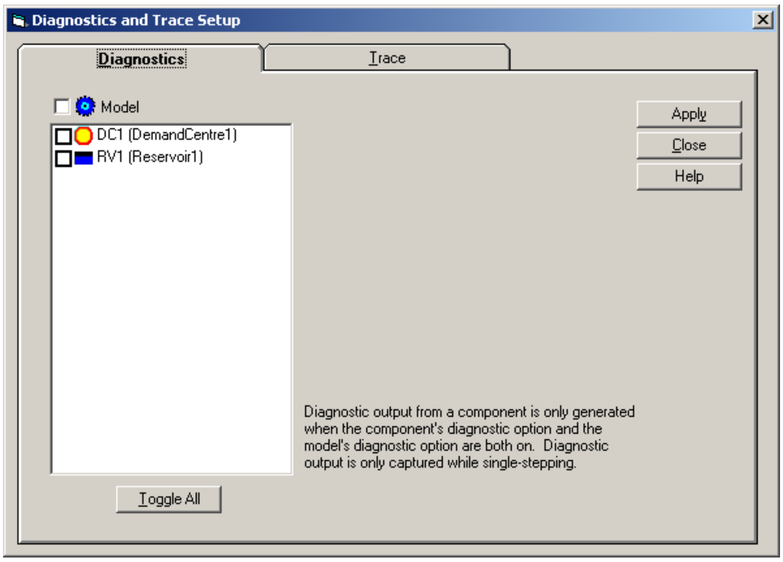
Optionally you can choose to:

- copy example *VBA* code to the chosen projects folder,
- copy any demonstration databases supplied with Aquator.
- copy an empty Aquator database file to use for your projects,

The empty database is called *AquatorStart.mdb* and is located in the Aquator installation directory. Although this dialog can copy and rename this database for you, other copies can be made manually at any time (but remove the *Read-only* attribute on the copied file).

Diagnostics and Trace Setup Dialog

This dialog appears when you choose the *Setup...Diagnostics and Trace...* menu command. It allows you to set up diagnostics and trace options globally i.e. across more than one component or type of component.

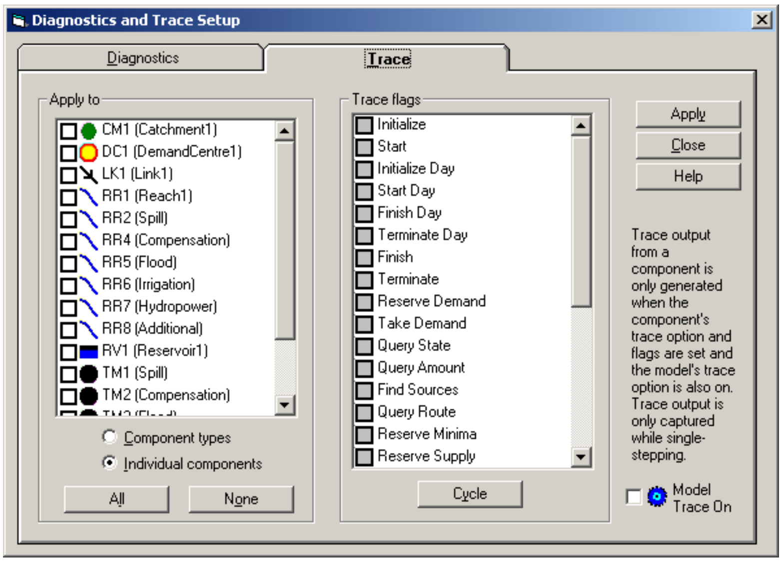


On the *Diagnostics* tab you can

- tick or untick components to enable or disable the generation of diagnostic output during a model run from those components
- tick or untick the Model check box to globally enable or suppress diagnostic output

The global Model option controls whether diagnostic output is generated as the model runs. Diagnostic output is only produced if this option is on *and* when single stepping the model, and only then from components which have individually been ticked.

Diagnostic output appears in a separate *Diagnostics* node under the model run node. It is terse but readable and is intended to log only the important stages in the calculations.



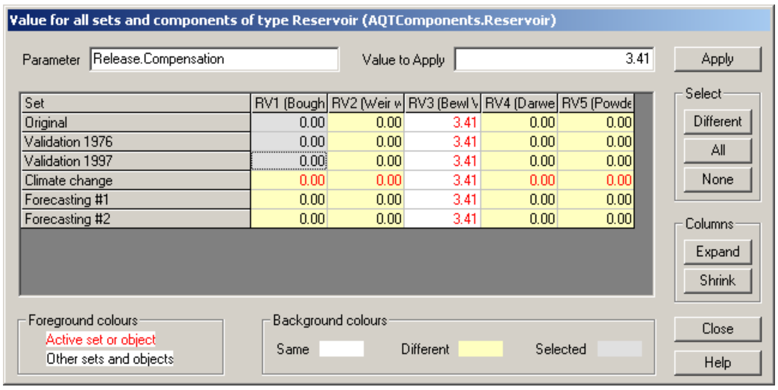
On the *Trace* tab you can

- tick the items in the Options group to choose which actions to trace,
- choose to apply the changes to components or to component types,
- tick the components or component types to apply the changes to.
- tick the Model Trace On check box to globally enable trace output.

The global option controls whether trace output is generated as the model runs. Trace output is only produced if this option is on *and* when single stepping the model, and only then from components whose individual *Options.Trace on* parameter is also on and whose *Options.Trace flags* parameter is non-zero.

Trace output appears in a separate *Trace* node under the model run node. It is verbose and barely readable and is intended to log every stages of the calculations no matter how trivial in order to assist in solving technical problems.

Edit Set/Component Values Dialog



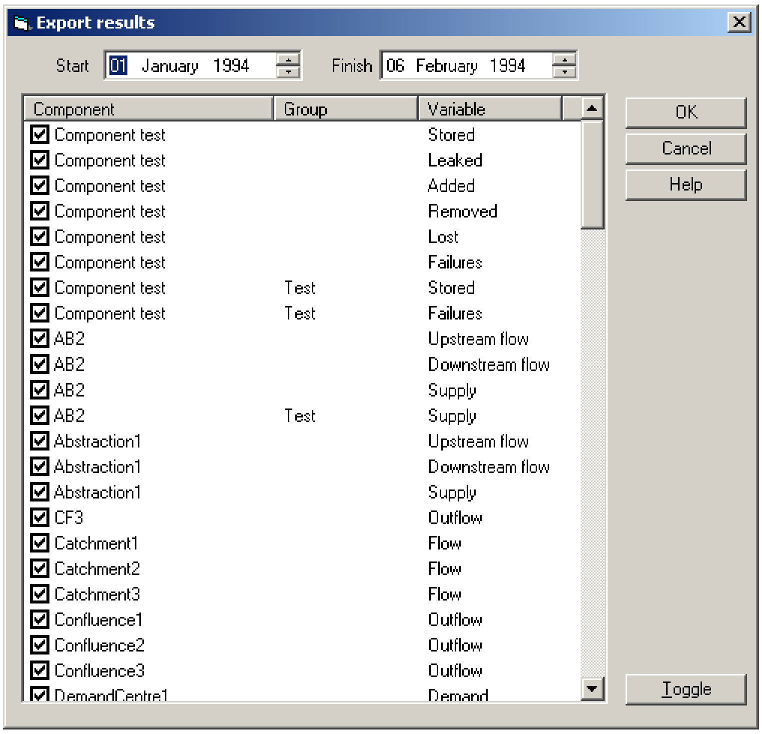
This dialog allows a parameter value to be changed in multiple sets and/or across multiple components. Access this dialog by right-clicking instead of left-clicking in an edit cell on any of the editing grids.

The columns represent components and the rows represent sets. On first appearance the *Value to Apply* textbox contains the value in the cell that you initially right-clicked on, while the white cells show which components or sets have the same value and the yellow cells show where the value differs.

By clicking the buttons in the *Select* group box, or by clicking individual cells in the grid, cells can be selected, as shown by their grey background.

Then clicking *Apply* will set the value in all selected cells to the value in the textbox, which can be altered first if necessary.

Export Results Dialog

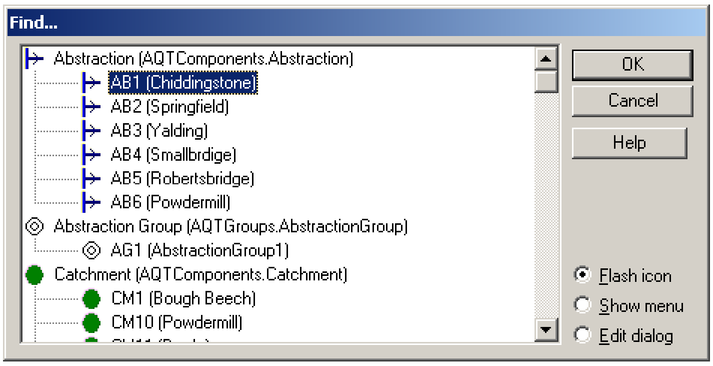


This dialog allows export of model run results in the same format as exporting time series and profile data. There are three ways to raise this dialog:

- left-click on the *Results node* to select it, then choose one of the tools available under the *Tools....Export* menu (this selects the format in which the data will be exported)
- left-click on the *Results node* to select it and click the *Export* button on the toolbar (the results will be exported in the format of the export tool which was last used)
- right-click on the *Results node* of the model run and choose *Export* from the popup menu which appears (the results will be exported in the format of the export tool which was last used)

Use the check box on each item in the list to select which results to export before clicking *OK*. A further dialog e.g. to choose a file in which to store the results may appear (this depends on the chosen format for the exported data).

Find Component Dialog

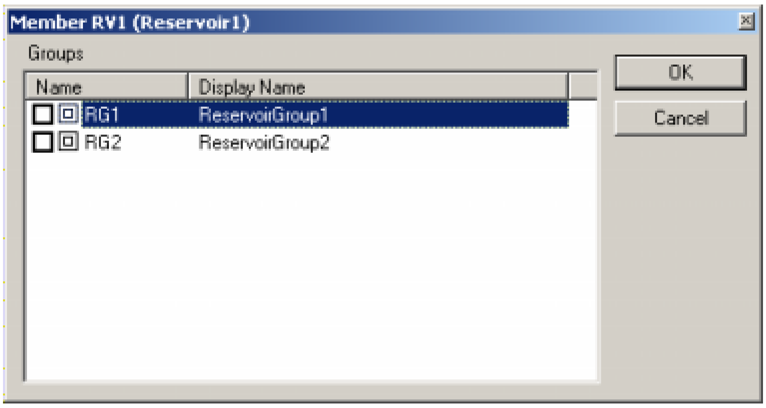


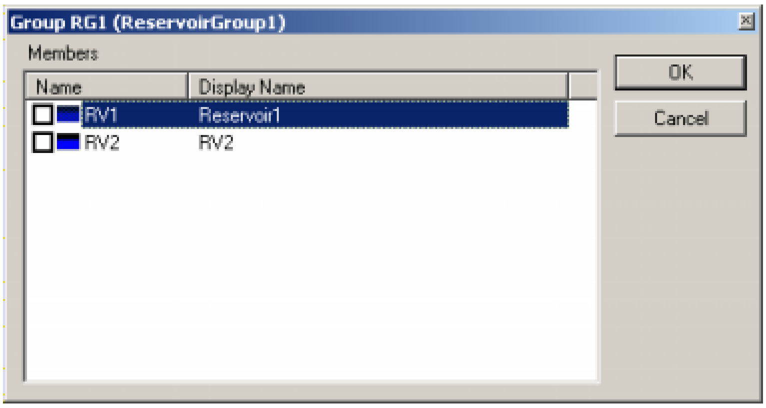
This dialog appears when you press Ctrl + F or select the *Edit...Find* menu item. To find a component or components on the schematic ensure that the *Flash icon* option is selected and

1. Either select a component such as *LK13 (L5)* in the above example and click OK. The dialog will close and the image of the chosen component will be flashed on the schematic. If necessary the schematic window will be scrolled to bring the component into view.
2. Or select a component type such as *Link(AQTComponents.Link)* and click OK. The dialog will close and the image of the first component of the chosen type in the list will be flashed on the schematic. If necessary the schematic window will be scrolled to bring the component into view. Subsequently pressing F3 or choosing menu item *Edit...Find Next* will find the next component of the chosen type. Each component so located will be added to the current selection.

Alternatively this dialog can be used to execute a popup context menu command for the component, or show the edit dialog for the component, using the keyboard only. Select either the *Show menu* or *Edit dialog* option first (the selection sticks until next changed), use the cursor keys to select a component, and press the RET key.

Groups And Member Dialog





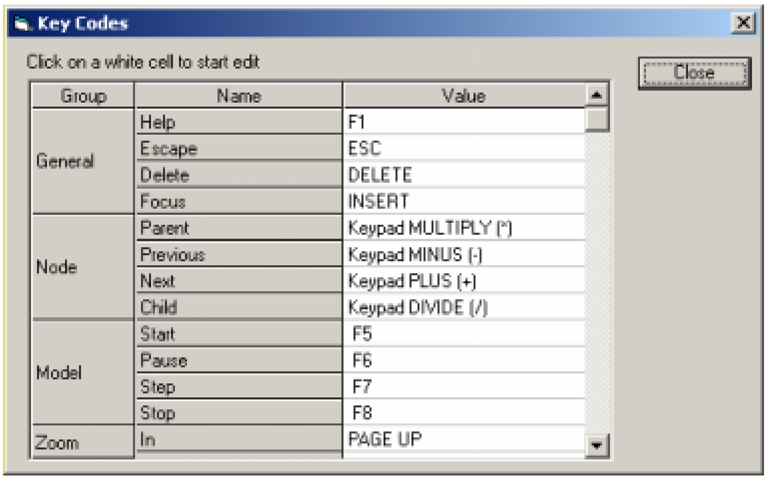
These dialogs are raised by clicking on a component or group in schematic view and selecting "Groups" or "Members" respectively. They perform the same function: allow components to join and leave groups.

Groundwater, reservoir and abstraction components can join abstraction groups. Reservoir components can join reservoir (storage) groups. And reservoirs and demand centres can joing demand saving groups. In all cases component operation is modified by being a member of one or more groups. Group licence constraints can be enforced and reservoir group operating rule curves applied.

Check or uncheck the members or groups in the above dialogs before clicking *OK*.

If group membership is ambiguous e.g. when a reservoir can join more than one group of different type, then the following dialog appears first.

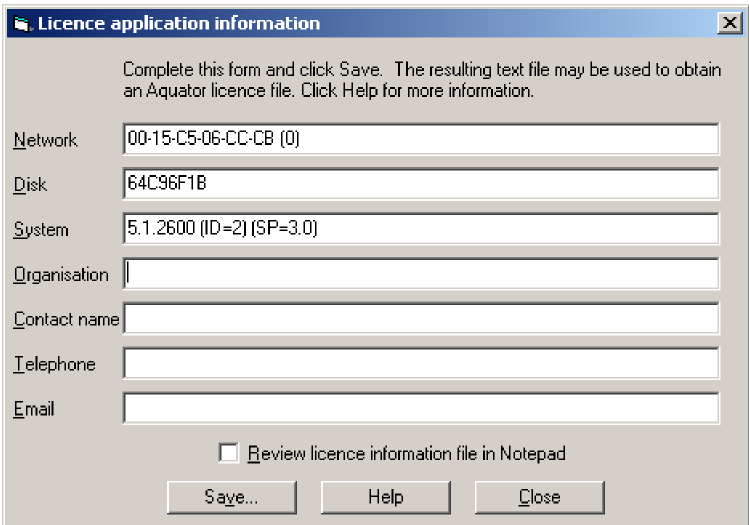
Keyboard Setup Dialog



This dialog appears when you select the *Setup...Keyboard* menu command. It allows you to choose various accelerator keys.

To alter a key assignment click in the cell and click the button which appears to select a different key from the drop-down list.

Licence Info Dialog



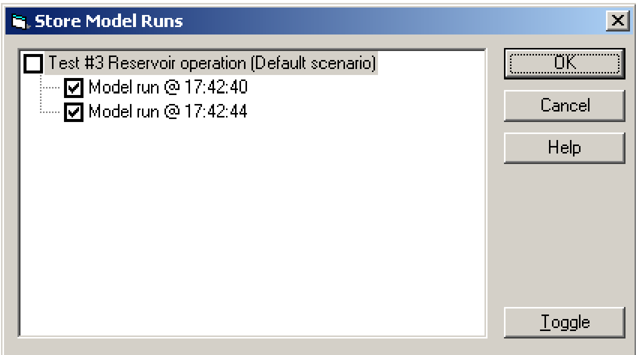
The use of Aquator is controlled by a licence file. Aquator licencees obtain a licence file by sending *Oxford Scientific Software* a small text file containing system and contact details. The licence file which will be returned to you is simply placed in the Aquator installation directory and thereafter is transparent in use.

To generate the required information use this dialog. Run Aquator but do not load a project (click *Cancel* on the *Startup dialog* if necessary). Then choose the *File…Licence Info* menu command.

The first three items on this dialog are automatically filled in. Complete the next four items and click *Save* to save the licence information in a text file. To proceed further you will need to receive instructions from Oxford Scientific Software, contact support@oxscisoft.com for further information.

If you wish to verify that no information other than that shown in the above dialog is transmitted then the licence information text file can be viewed in any text editor, or simply tick the *Review licence information in Notepad* check box before clicking *Save*.

Model Runs Dialog



The File menu commands *Model Runs...Delete, Model Runs...Store,* and *Model Runs...Retrieve*, can be used to delete, store, and retrieve model run results. This dialog appears when any of these commands is selected.

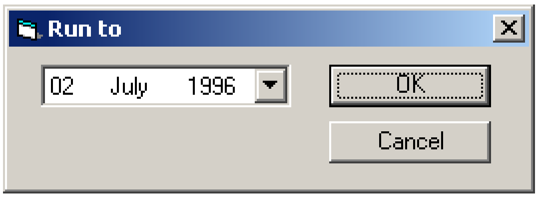
In the case of *Store* (above screen shot) a list of all model runs that have been executed in the current session appears. After checking those to be stored and clicking OK, each selected model run is saved to a disk file.

These model runs may later be reloaded, by choosing *Retrieve*. The above dialog appears with the caption *Retrieve Model Runs* and a list of all files known to contain model run results. After selecting the desired file or files, clicking *OK* results in the selected model runs being reloaded, exactly as if the runs had just been executed.

It is perfectly possibly to store a model run from one project and retrieve it into another project. But if the two projects do not share a common database any or all of the sequence links stored in the *Sequences node* of the retrieved model run may be broken, since typically the two databases will not contain the same time series and profile data. This causes no harm except that the sequences node will not be restored correctly.

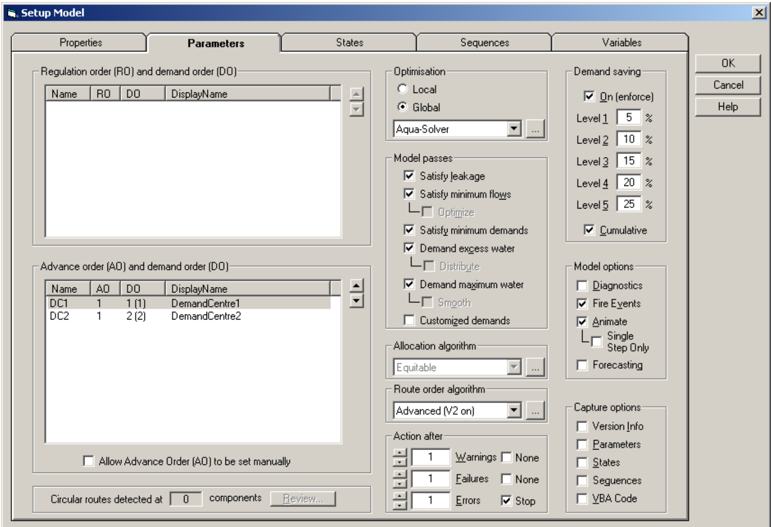
Retrieving model run results also re-plots all results on the charts that appear automatically at the end of a run.

Model Run To Dialog



This dialog appears when the *Run To* button on the Aquator Toolbar is clicked. It allows you to enter the date at which the model run is automatically paused. Clicking OK will start the model run.

Model Setup Dialog



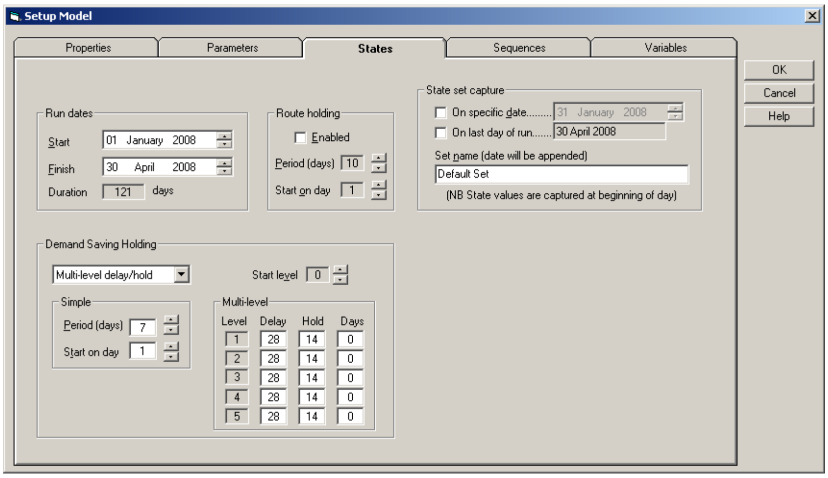
This dialog appears when you choose the *Setup...Model...* menu command or when you click on the Aquator Toolbar button that corresponds to this command.

Most of the information on each tab of this dialog can be found in five nodes in the tree view window under the Schematic node. These are the tree view nodes Properties, Parameters, States, Variables, and Sequences. This dialog, however is easier to use.

On the Parameters tab you can:

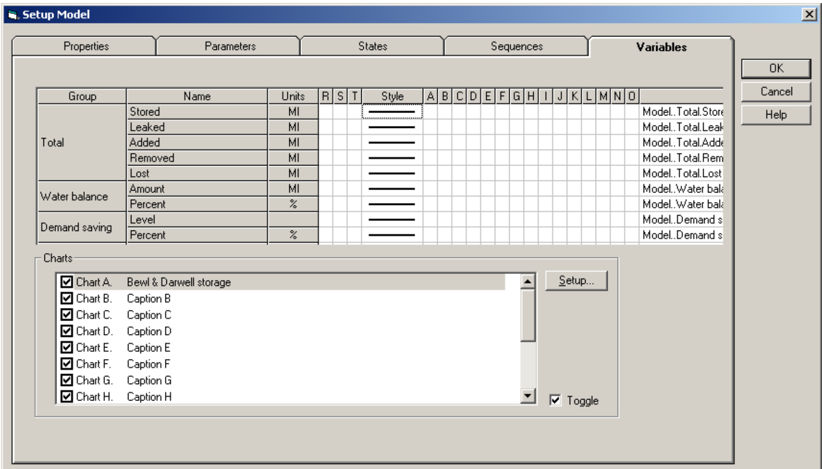
- Turn *Aqua Solver* on or off (see *Setup and use of Aqua Solver global optimisation*).

- use the upper up-arrow and down-arrow buttons to alter the Regulation Order (RO) and Demand Order (DO) of all regulators (see *Model Calculations*, p. 40)
- use the lower up-arrow and down-arrow buttons to alter the Demand Order (DO) of all demand-type components within a given Advance Order (AO), which is fixed by Aquator (see *Model Calculations*) unless you tick the *Allow Advance Order (AO) to be set manually* check box. If ticked then the lower up-arrow and down-arrow buttons can be used to alter Advance Order as well as Demand Order
- choose which model passes will be executed; see *How it all works – Demanding water* for a description of these passes.
- select from one of available allocation algorithms that determine how water is shared by demands; click on the "**...**" button if any of these tools require customization (see *Allocation Algorithm – local optimisation*).
- select from one of available route ordering algorithms that determine the order in which routes are used; click on the "..." button if any of these tools require customization (see *Route ordering*).
- choose whether to abort a model run on certain events; *Warnings* are events that should not happen, but probably do not cause failure to supply such as a minimum link flow not being met. *Failures* are events like inability to meet demand; *Errors* should not happen and signal either a bug or that a component cannot operate as intended because of bad input data
- set the demand saving levels (see *Demand Saving.*)
- turn on or off the capture of parameters, states, and sequence links at the start of the model run; turn these options on to provide an audit trail of the data that went into a model run
- set some global options for a model run which will slow execution if they are enabled; *Diagnostics* and *Trace* options generate diagnostic and trace information respectively when single-stepping; *Fire Events* is normally on to allow any VBA custom code to trap events and thus execute; *Animate* turns schematic animation on or off and should be off for lengthy runs; *Forecasting* enables or disables regulators to make forecast releases (see *Options.Diagnostics on, Options.Trace on, Options.Events on, Options.Animate on, and Options.Forecasting on*)



On the States tab you can:

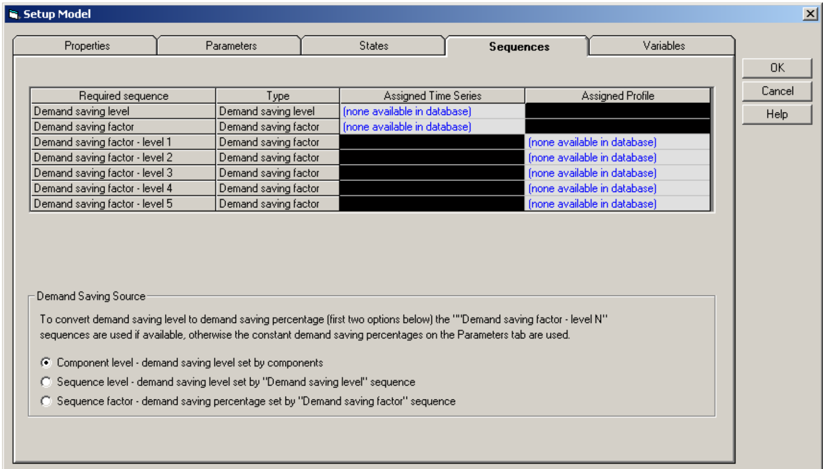
- set the start and end date of the model run
- set up the route holding algorithm which smooths model results by not allowing the route ordering to change every day
- set when and if the entire state of the model is captured on one day during the run and at the end; this will automatically add another project state set that can be used as the starting point for another model run
- choose the *Demand saving hold* method from *None*, *Simple period*, or *Multi-level delay/hold*. See the description of *Demand saving hold*.



On the Variables tab you can

- Setup the project variables in the usual way, by clicking on any of the white cells; see *Variables Node*, *p. 112*,

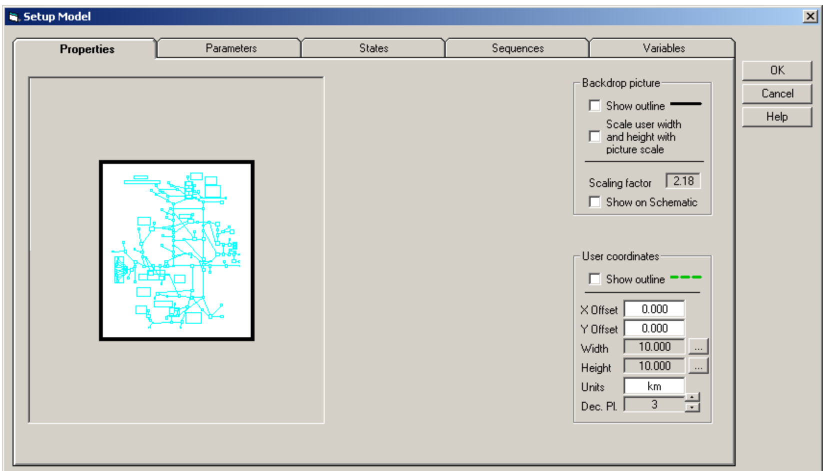
- Setup each of the automatic charts that is plotted at the end of a model run; see the *Chart Setup Dialog*, *p. 192*.
- By ticking or unticking the checkboxes in the *Charts* list the corresponding charts will either be automatically plotted at the end of a model run or not.



On the Sequences tab you can

- assign the time series and profiles which control demand saving
- select from the options that control whether demand saving is imposed by components such as reservoirs or by a sequence, and in the latter case whether to use the demand saving level sequence or the demand saving factor sequence

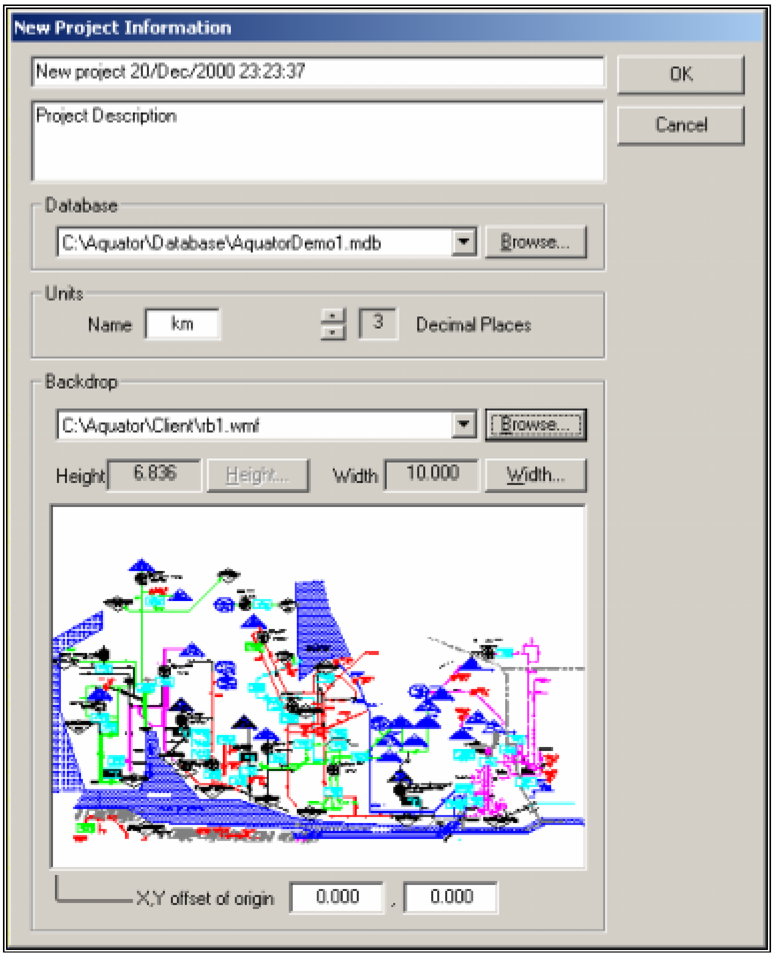
See *Demand Saving* for more information.



On the Properties tab you can

- drag the boundaries of the thumbnail image to alter the schematic extent e.g. to enable adding more components around the boundary
- if a schematic backdrop picture has been selected then dragging the top-right corner of the backdrop outline (not shown in the above screen shot) scales the backdrop size
- alter the user-defined coordinates shown in the status bar at the bottom of the schematic as the mouse is moved

New Project Dialog



This dialog appears when you select the *File...New...* menu command. It allows you to create a completely new project.

Enter a new project name, or accept the default (a typical example is shown above), and enter a description. Both may be changed later.

Browse to an Aquator database file which will hold the project. Preferably, do not use the AquatorStart.mdb empty database file placed in the installation directory. Rather, copy this to a new location and rename before selecting.

The Aquator status bar shows co-ordinates in user-selectable units. Enter the units used and choose the number of decimal places used for display. These have no hydrological implications and are merely a display aid.

Similarly the size of the schematic working area can be chosen in one of two ways:

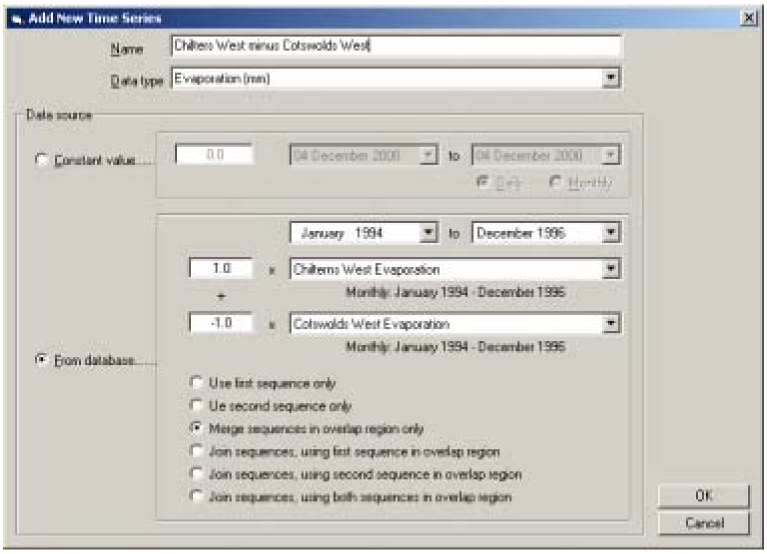
- click on the *Height* and *Width* buttons to set the height and width independently
- browse to a backdrop file, preferably a metafile, which is used to set the aspect ratio; then set either the width in your arbitrary units

A backdrop file is drawn as a background to the schematic. The above image shows an example but its use is somewhat problematic for the following reasons:

- it slows redraw operations and/or uses large amounts of memory
- bitmap backdrops will not scale well
- changing the backdrop to a different picture will typically not be successful unless the size information encoded in the bitmap or metafile is identical to that used previously
- on the more recent versions of the Windows platform it is possible to fade the backdrop, and Aquator provides this facility; on older versions of Windows this ability is missing and backdrops therefore tend to be overpowering and make it very difficult to see the components. If this is the case it may be worth fading the image in a drawing package before using in Aquator.

Nevertheless, since backdrops can be removed later, no permanent problems will ensue if a backdrop is chosen.

New Time Series or Profile Dialog



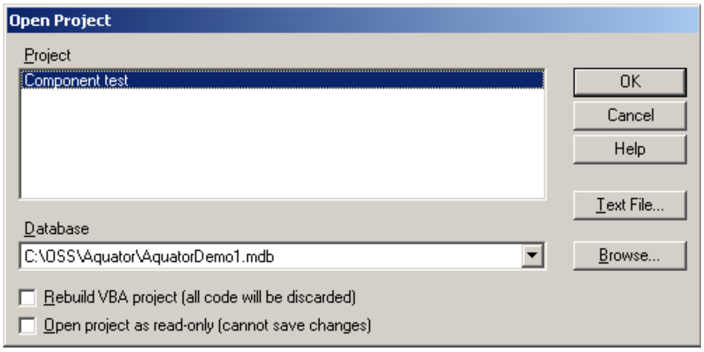
This dialog appears when the *Add* button on the *Database Maintenance Dialog* is clicked. It allows you to create a new time series or profile from existing time series or profiles.

The *Constant value* option provides a quick and simple way to get new data into the database.

The *From database* option allows the new series to be created by:

- *Use first sequence only* and *Use second sequence only*: scaling a single existing time series or profile
- *Merge sequences in overlap region only*: scaling and adding two existing time series or profiles; this can only be done over the region of overlap; a negative scaling factor allows one sequence to be subtracted from the other
- *Join sequences, using first sequence in overlap region*: stitch together two time series (option does not apply to profiles) with scaling; in the overlap region use the values from the first time series
- *Join sequences, using second sequence in overlap region*: stitch together two time series (option does not apply to profiles) with scaling; in the overlap region use the values from the second time series
- *Join sequences, using both sequences in overlap region*: stitch together two time series (option does not apply to profiles) with scaling; in the overlap region use the values from both time series i.e. merge by scaling and adding

Open Project Dialog



This dialog appears when the *File...Open...* command is chosen. After choosing a database any existing project in that database may be opened.

Rebuild VBA project (all code will be discarded)

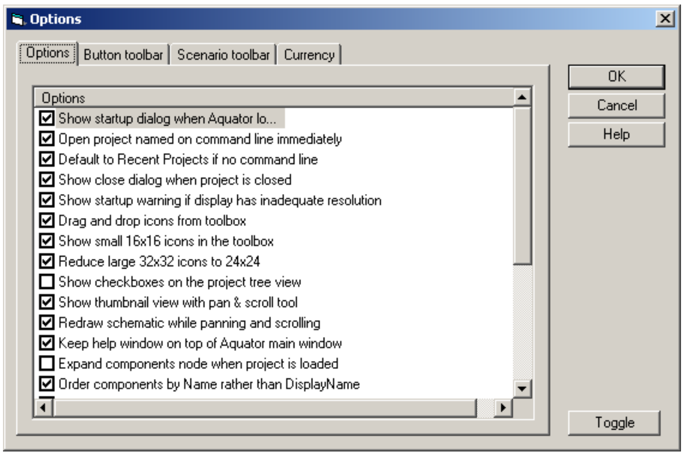
If checked then any VBA customization code in the project will be discarded when the project is loaded into memory. To delete VBA customization code entirely the project must then be saved i.e. this option does not alter the copy of the project in the database.

Open project as read-only (cannot save changes)

If checked then the Save command on the File menu and the Save button on the toolbar will be disabled. This option serves only to guard against accidentally saving a project which has been modified. This read-only attribute can be added or removed at any time using the *File...Read-Only* command

Options Dialog

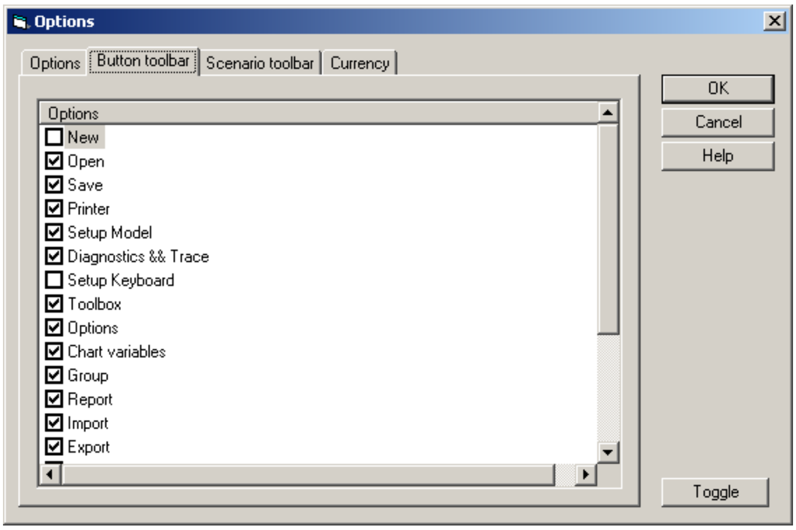
These dialogs appear when the *Setup...Options...* menu command is chosen.



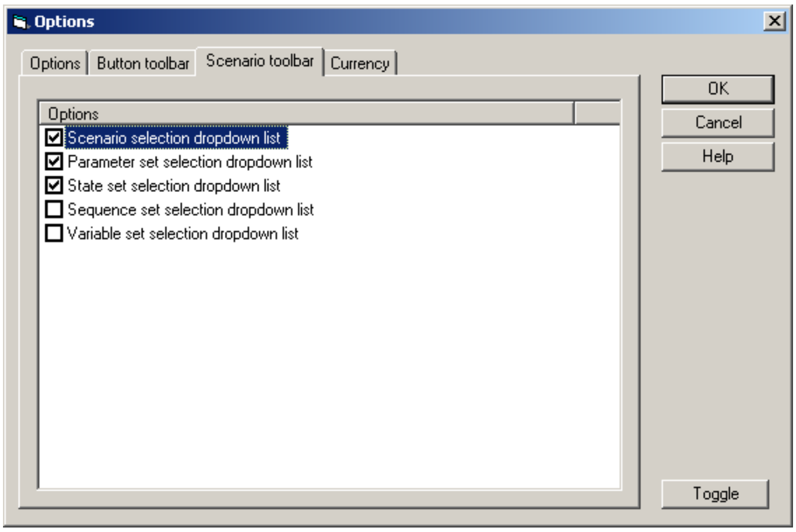
The Options tab allows you to choose:

- whether the startup dialog is shown when Aquator loads
- if so, whether a named project is opened immediately or Aquator waits for you to click OK
- which tab to default to when the startup dialog is shown
- whether or not a dialog is shown when a project is closed
- whether to display a warning at startup if the display resolution is below Aquator's minimum requirements
- whether to enable component placement by drag and drop from the toolbar (the alternative action is to click on a toolbar component and then click on the schematic)
- whether to show small or large icons in the toolbox
- whether to reduce the size of large icons
- whether check boxes are shown on the project tree view (this allows you to choose multiple nodes in the project tree view on which some commands can act)
- whether to display the thumbnail view of the schematic (this contains a rectangle which can be dragged to change the portion of the schematic displayed at full size)
- whether to redraw the schematic continuously during pan and scroll operations (on a slow machine it may be better to turn this option off)
- whether the Aquator help window always stays in front of the Aquator window, otherwise it can be pushed behind
- whether the components node in the project tree view is automatically expanded when the project loads
- the ordering of components on the project tree view and various dialogs (the Name property is a unique fixed abbreviation like "RV1" while the DisplayName property is an editable user-friendly string like "Main Reservoir")
- how the Paste context menu command works (the choices are to paste immediately at the mouse cursor location or to display a movable ghost image of the components to be pasted which are placed only when a mouse button left click is detected)
- whether to compress projects saved to the database; this should normally be on to reduce database size significantly
- whether to compress model runs saved to the database; this should normally be on to reduce database size significantly

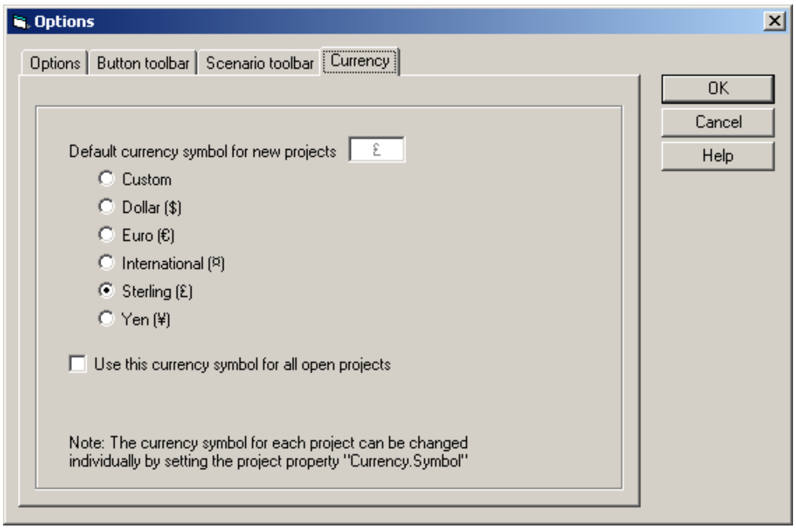
- whether to compress pictures saved to the database; this should normally be on to reduce database size significantly
- whether to check if any variables will be captured during a model run; this slightly slows execution; it is legal to run a model that produces no output
- whether to check a project when it is loaded for VBA code that will never execute; if a component is customized but the component's *Options.Events on* parameter is False then any custom code for that component will never execute; this may be intentional but without this check it may not be apparent that custom code is not executing



The *Button Toolbar* tab allows you to choose which buttons appear on the Aquator Toolbar. It is not possible to hide the model run/run to/step/pause/stop buttons. This dialog tab can be quickly accessed by double-clicking in an empty region of the toolbar.



The *Scenario toolbar* tab allows you to choose which dropdown lists appear on the scenario toolbar, if any. This dialog tab can be quickly accessed by double-clicking in an empty region of the scenario toolbar, if visible.



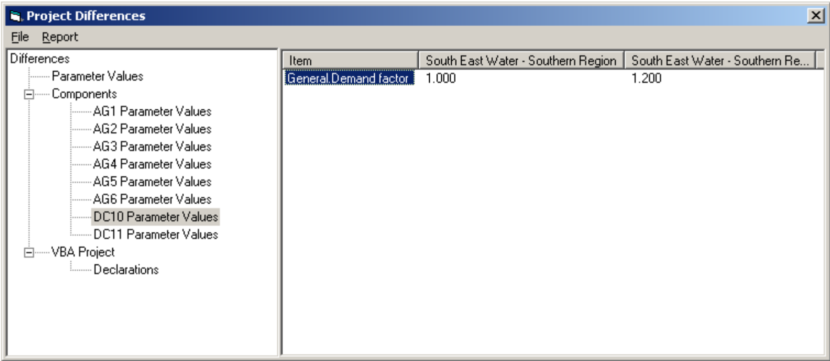
The *Currency* tab allows you to

- choose a default currency symbol for new projects
- apply the chosen symbol to all open projects

Note that the project property *Currency.Symbol* allows any individual project to use a different symbol.

Project Differences Dialog

When two or more projects are open then any two can be selected and any differences displayed using the *File...Project differences* command. Generally this is only useful if the projects are closely similar, for example if the *File...Save As* command has been used.

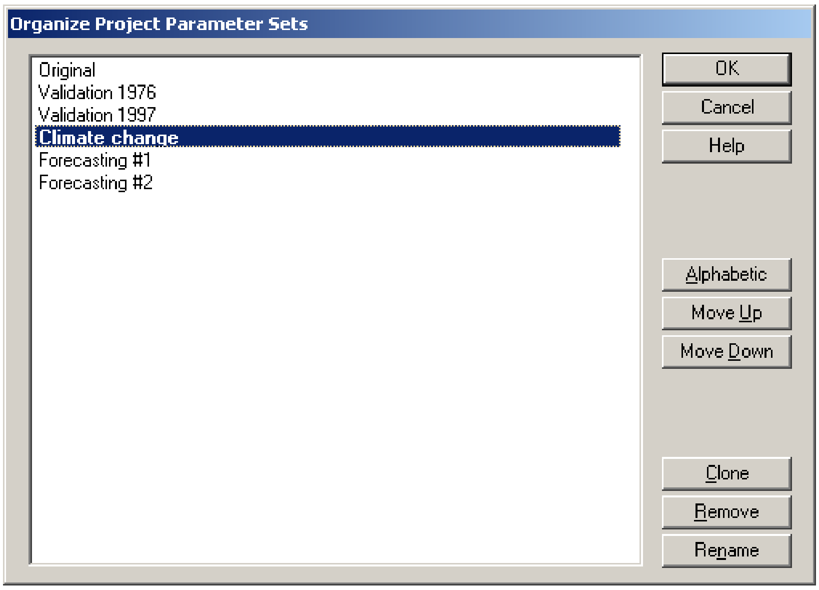


The two chosen projects are scanned and only those parameters etc. that differ in value are shown. The TreeView control on the left lists the components where differences can be found and the ListView control on the right shows the two different values.

Any VBA customization code is also scanned for differences.

As an alternative the *File...Database* command shows a dialog with a *Compare With …* button. This exhaustively scans the two projects, listing every item whether identical in value or not, and highlighting the differences. The *Project differences* dialog is a much more efficient of way of scanning for differences however.

Project Sets Dialog



This dialog appears when one of the four *Project Sets* buttons on the *Project Node* information window is clicked. These are the four buttons immediately adjacent to the four drop-down lists which show, respectively, the parameter, state, variable, and sequence set currently active.

A parameter set is a set of values, one value for each and every parameter in the model and in every component. This powerful facility allows you to switch between parameter set values with a single click. It avoids the necessity to regularly edit large numbers of individual parameter values e.g. when switching from prediction modelling to climate change modelling.

Similarly, a state set is a set of values for every state in the model and every component. A variable set is a set of flags for every variable in the model and every component. And a sequence set is a set of time series and profile links for every sequence in the model and every component.

Each of these categories has one default set. This dialog allows you to add, remove, and rename sets. The example above shows the dialog raised to organise parameter sets. The behaviour and use is identical for state, variable, and sequence sets.

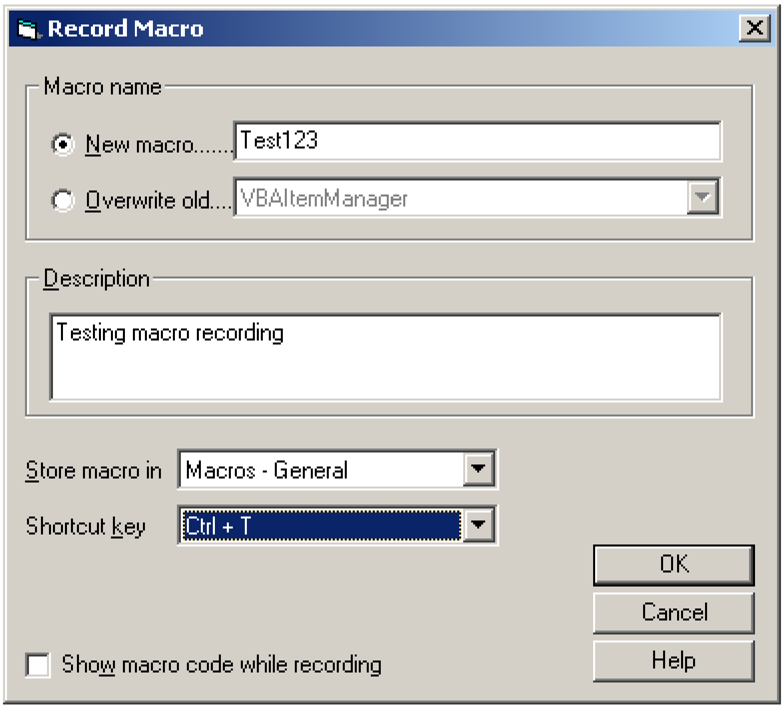
- Click *Clone* to add a new set. A new set is **always** a copy of an existing set.
- Click *Edit* to enter edit mode and rename a set, or just click on the name itself.
- Click *Remove* to remove a set. This will delete every value, flag, or link associated with the set.

- Use the *Alphabetic*, *Up*, and *Down* buttons to set the order in which these sets will appear.

The currently active set will be shown in bold. This set cannot be removed.The original (default) set also cannot be removed, even if it has been renamed.

See *Scenarios and Sets* for more information.

Record Macro Dialog



Selecting the menu item *Tools...Macro...Record New Macro* or pressing Ctrl + F9 begins the process of recording a macro by showing this dialog.

A macro is a VBA subroutine with no arguments that is stored in a module in a VBA project. Macros so stored can be executed from the Aquator user interface by selecting the macro name from the VBA toolbar button dropdown list:

or using the menu item *Tools...Macro...Macros*. See *Recording Macros* for more information on recording macros.

On this dialog you can

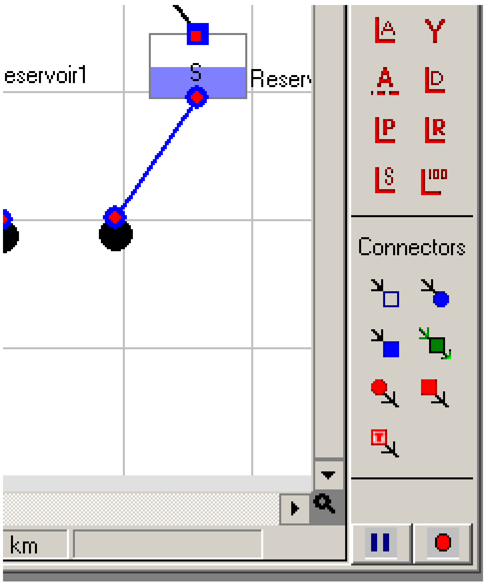
- Choose to create a new macro or overwrite an old one; the name of any new macro must be a valid VBA identifier made up of the letters A-Z and a-z, the digits 0-9, and the underscore character only (but the first character must be a letter)

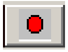
- Enter an optional description; this will appear as a comment in the macro body
- Choose where to store the macro; the choices will consist of any existing module in either the General VBA project or the VBA project that corresponds to the active Aquator project
- Choose a shortcut key; pressing this key after the macro has been recorded will execute the macro
- Optionally choose to open a modeless dialog so you can watch the macro code being created (you can also flip to the VBA IDE to examine the maco at any time)

After clicking OK your actions may result in VBA code being recorded into the macro i.e. inserted into the subroutine whose name is the macro name. For example, if you edit a parameter value a line of code that will change the parameter value will be created.

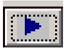
Not every action results in code generation. Some actions have no counterpart in VBA i.e. there is no VBA code that will have the same effect.

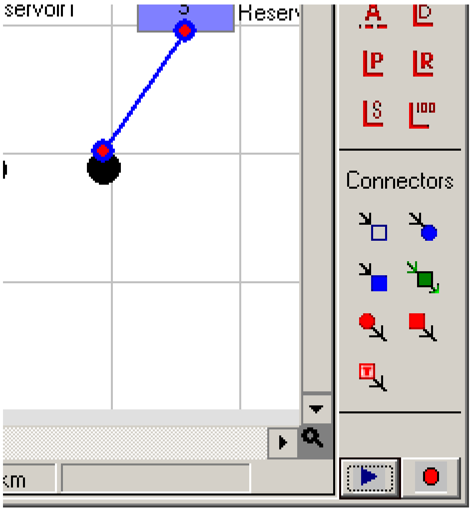
While recording two buttons will appear at the extreme lower-right of the main Aquator window as shown below or on the macro code window if the *Show macro code while recording* check box is ticked on the above form.



 The *Stop* button terminates macro recording and these two buttons will then disappear.

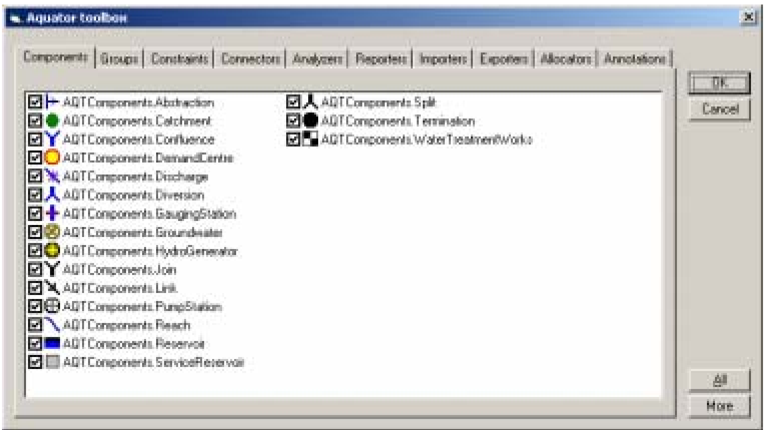
 The *Pause* button prevents any more code being inserted into the macro until the *Resume* button is clicked.

 Clicking on the *Resume* button will resume the macro. When paused the *Resume* button will replace the *Pause* button.



The *Resume* and *Stop* buttons appear while macro recording is paused.

Toolbox Setup Dialog



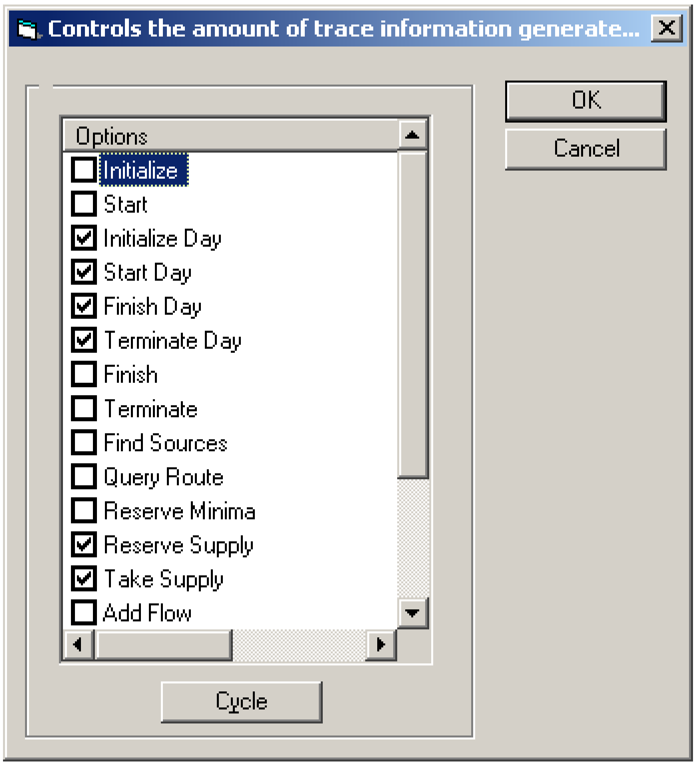
This dialog appears when you choose the *Setup...Toolbox...* menu command. In normal use there should be no need to access this dialog.

It allows you to control which tools appear in the Aquator Toolbox and in the Tools menu. Check the tools that are to be used.

There are two possible uses for this dialog:

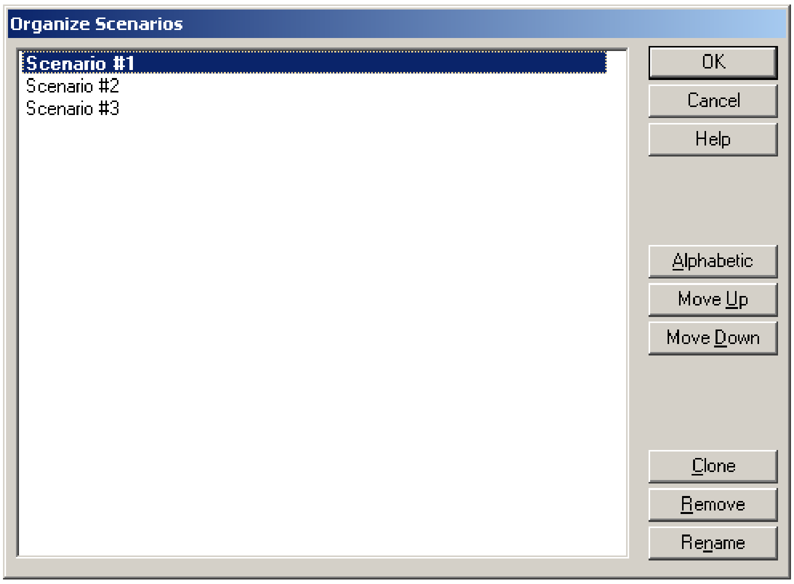
- to remove a tool that you will never use, or which depends on a third-party application that is to be uninstalled
- to add a tool which has been recently installed or enabled by the installation of a third-party application

Trace flags dialog



This dialog appears when editing the *Options.Trace flags* parameter of a component. Here the actions that will be traced for debugging purposes during a model run can be selected.

Scenarios Dialog



This dialog appears when the *Organize project scenarios* button on the *Project Node* information window is clicked. This is the button adjacent to the dropdown list of available scenarios. The *Project Node* is the topmost node of the project tree.

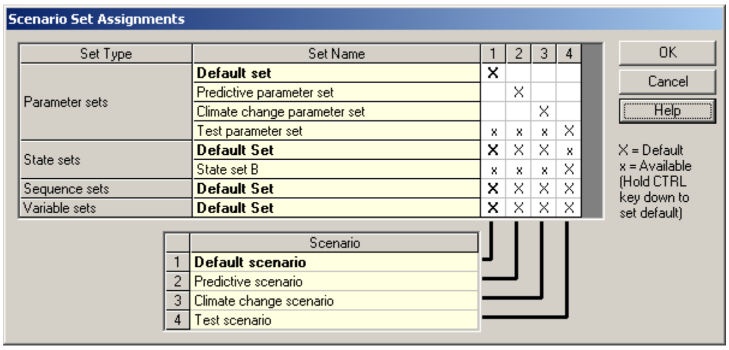
A scenario implicitly selects a parameter set, a state set, a sequence set, and a variable set. See *Scenario Sets Dialog* for how to assign sets to scenarios.

- Click *Clone* to add a new scenario. A new scenario is **always** a copy of an existing one.
- Click *Edit* to enter edit mode and rename a scenario, or just click on the name itself.
- Click *Remove* to remove a scenario.
- Use the *Alphabetic*, *Up*, and *Down* buttons to set the order in which these scenarios will appear.

The currently active scenario will be shown in bold. This scenario cannot be removed. The original (default) scenario also cannot be removed, even if it has been renamed.

See *Scenarios and Sets* for more information.

Scenario Sets Dialog



This dialog appears when *Assign sets to scenarios* button on the *Project Node* information window is clicked. This is the button adjacent but one to the dropdown list of available scenarios. The *Project Node* is the topmost node of the project tree.

A scenario implicitly selects a parameter set, a state set, a sequence set, and a variable set. This dialog assigns sets to scenarios.

An uppercase 'X' symbol in a white cell shows which set will become active when the corresponding scenario is selected. A lowercase 'x' symbol shows an alternative set that may be selected instead. An empty cell indicates a set that cannot be selected when the corresponding scenario is chosen.

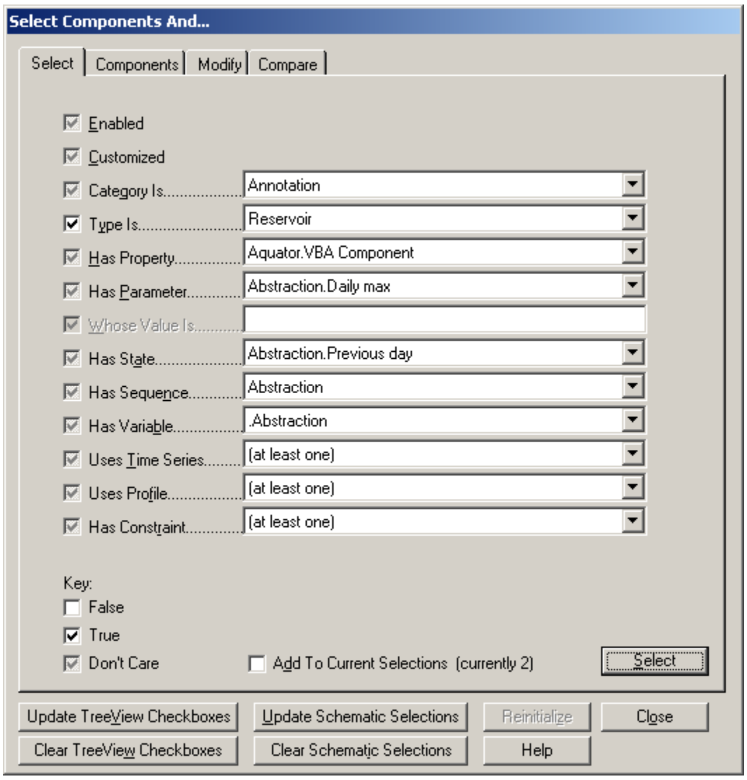
Thus selecting a particular scenario from the dropdown list referred to in the first paragraph will fill the set dropdown lists with those sets whose corresponding cells were not blank.

The intention is to restrict the selection of sets to valid combinations. As you create scenarios and sets this dialog provides a single place where the assignments are made. Thereafter the chosen assignments are enforced and it is not possible to choose an invalid combination.

To switch between a blank cell and a lowercase 'x' simply click on the cell. To move the uppercase 'X' to a different cell hold down the Ctrl key while clicking.

See *Scenarios and Sets* for more information.

Select Components Dialog

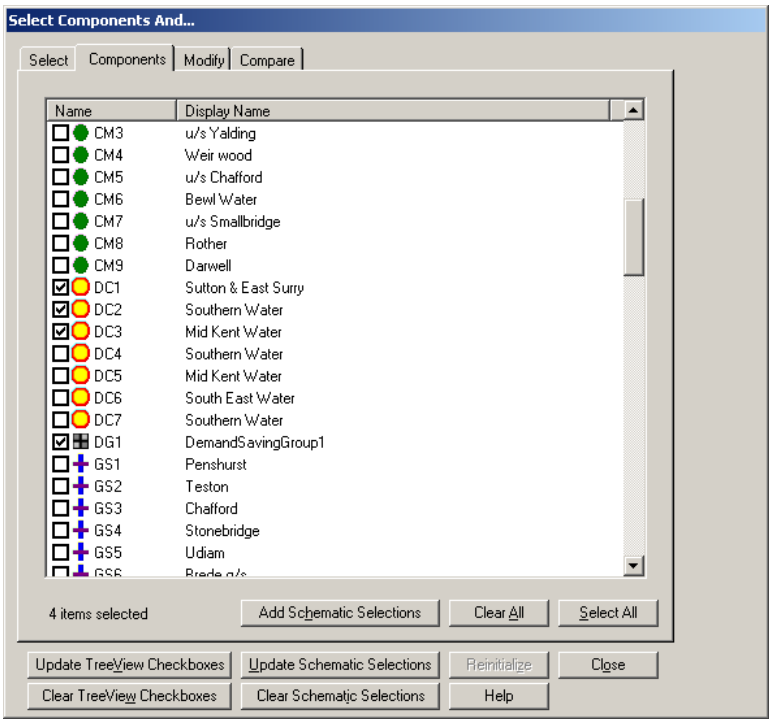


This advanced dialog enables modification of multiple components simultaneously and comparison of the parameters, stets, etc of any pair of components. It is reached by using the *Setup...Components* menu command.

The first tab on this dialog (shown above) is used to select components with particular characteristics. Each check box can be set to *False, True,* or *Don't Care* (see key at bottom-left of screen shot) and various characteristics, such as component type or whether the component has a particular parameter, can be selected in the drop-down lists. Then clicking the Select button searches the project and selects all components that satisfy the specified criteria.

Selections are preserved across multiple invocations of this dialog but it may be useful sometimes to click the Reinitialize button (if enabled) in order to reset the dialog to the initial state e.g. if the project has changed.

The next tab, *Components*, is used to review and refine the selection if necessary.



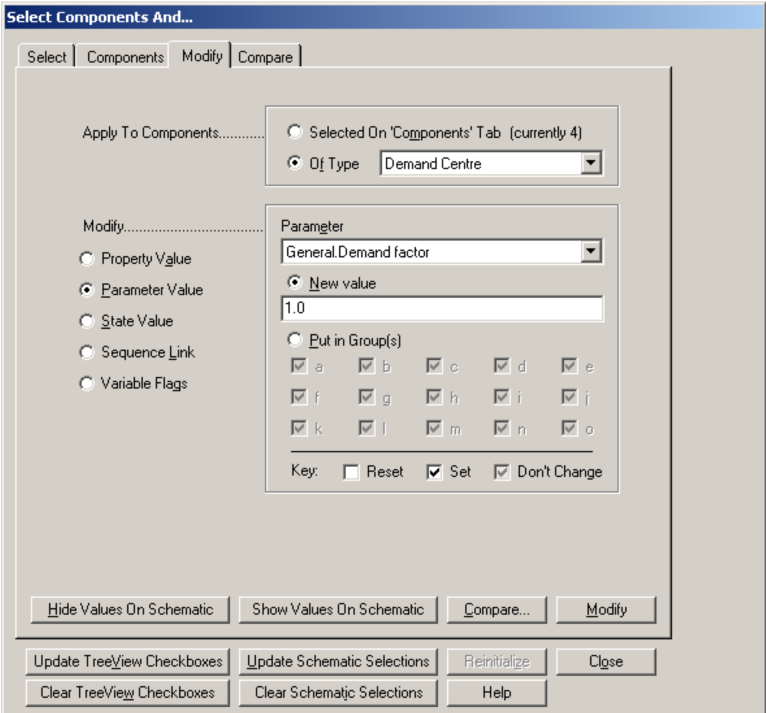
Here each checkbox can be simply ticked or cleared, or click the *Clear All* or *Select All* buttons to deselect or select all components.

In addition the *Add Schematic Selections* button will tick the checkbox of any component currently selected on the schematic.

Conversely the selections can be shown in the project tree view and on the schematic. Clicking the button *Update TreeView Checkboxes* will show check boxes on each node in the project tree (if not already shown) and then check those nodes that correspond to selected components. Use the *Clear TreeView Checkboxes* button to remove all checked items in the project tree.

Similarly the *Update Schematic Selections* button will place a selection box around each selected component on the schematic. Use the *Clear TreeView Checkboxes* button to clear all selections on the schematic.

Having selected components as required on this and the previous tab, the *Modify* tab of the dialog can be used to make changes across multiple components.



In fact this tab can be used immediately, without selecting any components first, if the intention is to modify components of a particular type. In that case click the *Of Type* option button and select the required type in the drop-down list in *the Apply To Components* group box above.

Otherwise the *Selected On 'Components' Tab* option button can be chosen to apply modifications to all selected components.

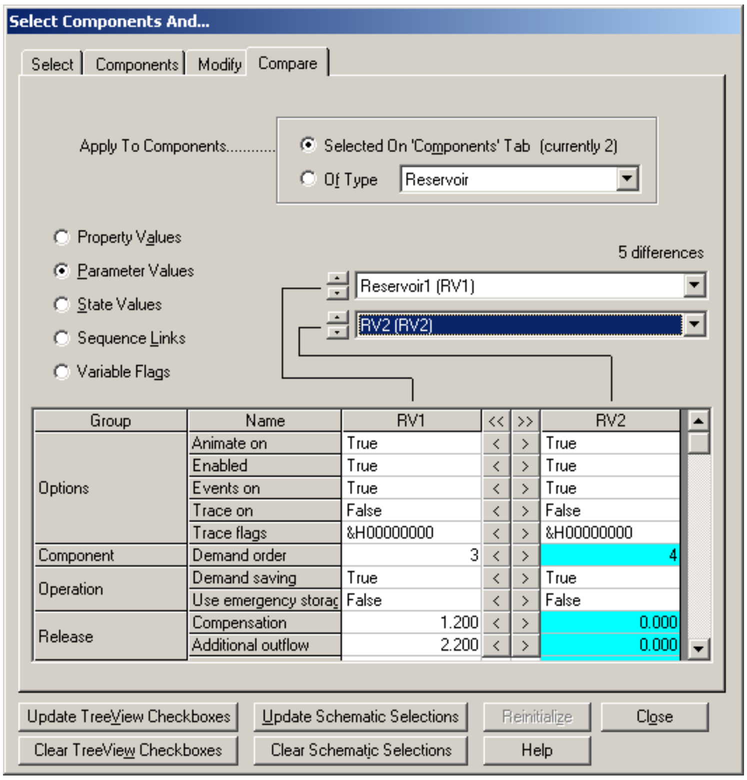
Choose one set of attributes to modify. The value of a property, parameter, or state, the link to a sequence, or the flags of a variable, can be modified. An alternative action is to put a selected parameter or state into a custom edit dialog by selecting the *Put in Group(s)* option and toggling the checkboxes.

Before or after any modification it may be useful to compare one component with another by selecting the *Compare* tab, which also allows copying from one component to another.

Alternatively, clicking the *Compare* button on this tab shows the *Compare Selected Components* dialog where the chosen parameter, state, etc is compared across all selected components.

For each component a single parameter value or state value can be shown or hidden on the schematic using the *Show Values On Schematic* and *Hide Values On Schematic* buttons.

Finally, click the *Modify* button to actually make the changes.

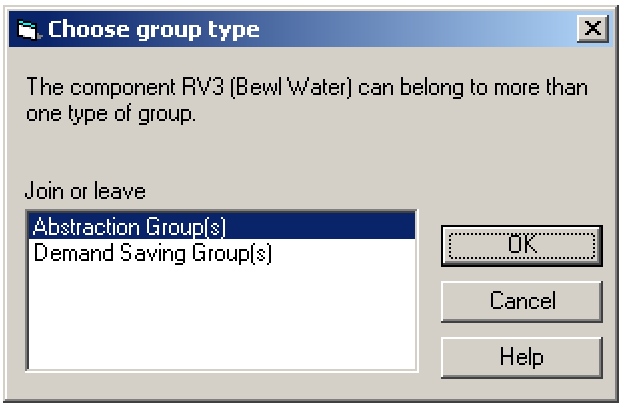


On the *Compare* tab any pair of components from those selected may be chosen from the drop down lists. Then the columns of the grid compare either the properties, parameters, states, sequences, or variables of the two components. Just above the first drop-down list the count of differences between the two components is shown. Differences are also highlighted in a different colour.

The two narrow grid columns whose cells each contain either the < or > symbol are in fact columns of buttons. Clicking any button copies the property, parameter, etc. in the chosen direction.

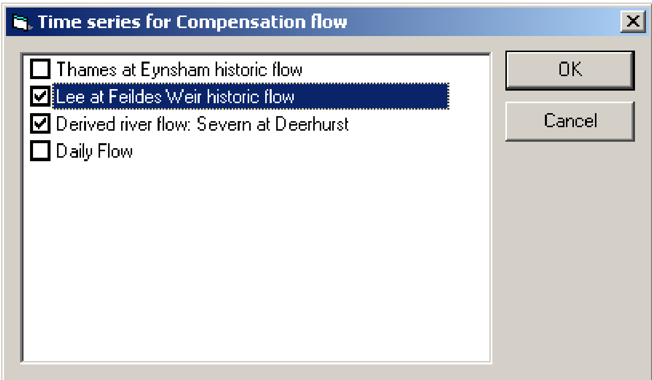
Clicking either the << or >> buttons at the top of these columns copies every item in the chosen direction. In this way components of the same type can easily be synchronized.

Select Group Type Dialog



This dialog appears after clicking on a component in schematic view and selecting "Groups". Because this component can be a member of more than one type of group you choose the group type first. Then clicking OK shows the *Groups and Members dialog* for the selected group type.

Sequence Links Dialog

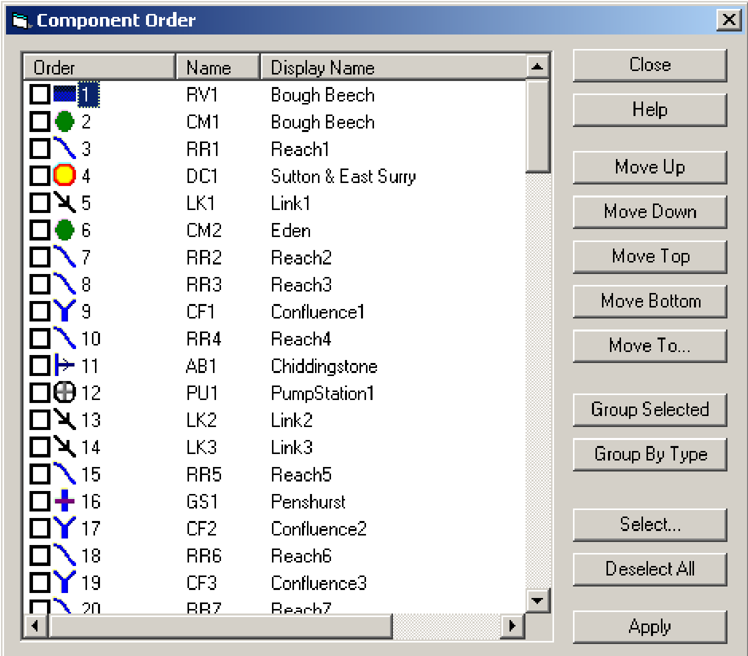


This dialog appears when you right-click on a time series link or a profile link in the *Sequences Node* grid cell.

Choose one or more of the time series or profiles shown in the list. These will become the subset of available time series or profiles that can be assigned to the link by subsequently *left-*clicking in the cell.

This two-stage process of assigning a time series or profile to a sequence allows you to filter what can be assigned from the potentially very large collection of time series and profiles available in the database.

Setup Component Order Dialog



The order in which certain calculations are done can be important. For example, the order in which DemandCentres satisfy demand and Reservoirs refill. This type of ordering is controlled elsewhere (see the *Model Setup dialog* and the description of *How it all works*).

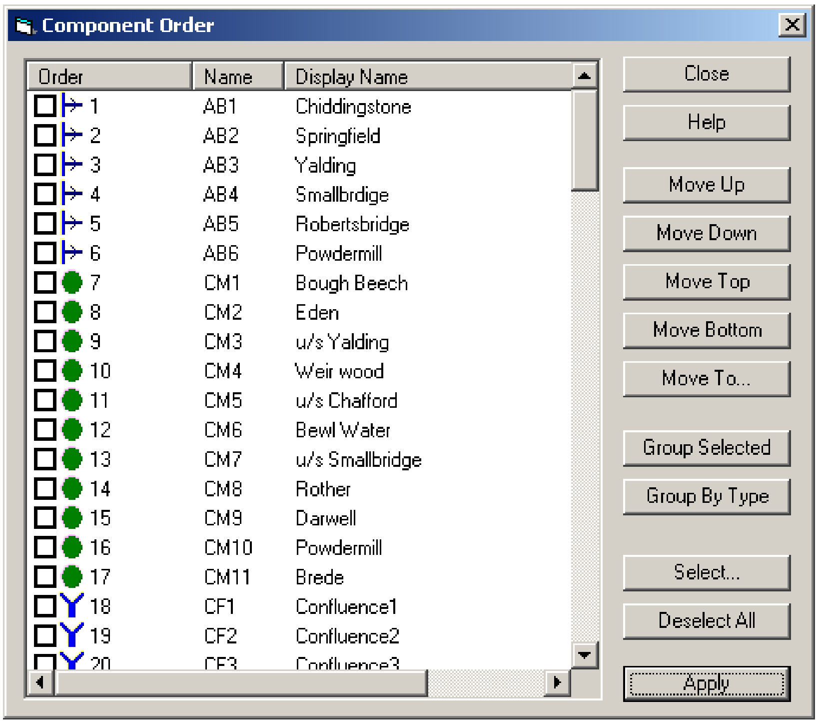
Otherwise an Aquator model is normally insensitive to most kinds of component ordering. For example, when the calculations for one day of a model run are started, each component's InitializeDay and StartDay methods are called. The order in which this happens is usually not important. Similarly for FinishDay and TerrminateDay, and most of the other methods that are called for each component in each day's calculations.

In the exceptional case where the order in which these methods are used is important this dialog can be used. It is shown through the *Setup...Component Order* menu item.

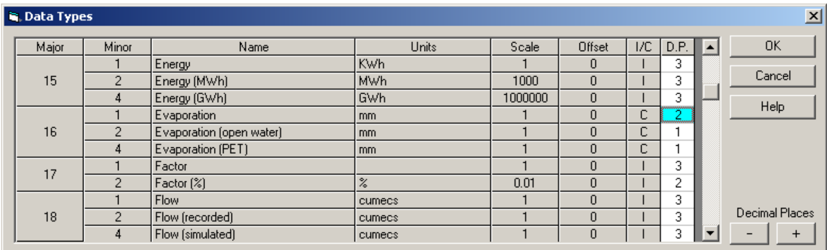
To change ordering, first select (tick) the components whose oreder is to be changed, then use one or more of the *Move...Up/Down/Top/Bottom/To* buttons, as required.

The *Group Selected* button moves selected components together. The *Grup By Type* button moves components of the same type together. The *Select* button selects via a string search.

As an example, the initial order shown in the screen shot above is changed as shown in the following screen shot after clicking *Group By Type* followed by *Apply*.



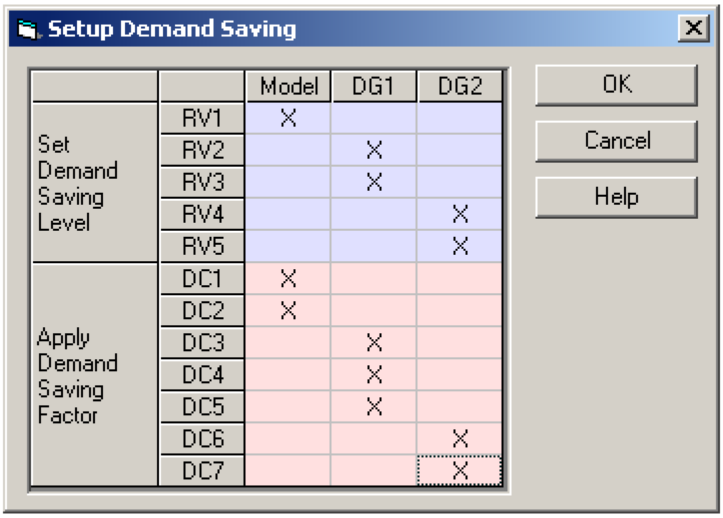
Setup Data Types Dialog



This dialog (menu item *Setup...Data Types*) allows the number of decimal places used to show values to be changed. Click in a cell in the 'D.P.' column to select a data type, use the '+' and '-' buttons to increase or decrease the number of decimal places, and click OK.

The settings are global across all databases and projects.

Setup Demand Saving Dialog



The *Setup Demand Saving* dialog is shown from the *Setup...Demand Saving* menu item.

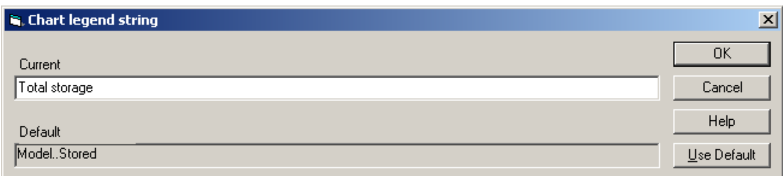
Demand saving can be enforced model wide or in groups. The model and each group can contain one or more reservoirs which determine the demand saving level to apply.

The level is converted into a demand saving factor and this is applied to all the DemandCentres in the model or corresponding group.

The level applied is the maximum level of any reservoir. The factor applied is the maximum value of any factor. These rules allow multiple reservoirs in a single group, and DemandCentres can be in multiple groups.

To use this dialog simply click in the cells to add or remove the 'X' symbol and finally click the OK button to apply the changes.

Setup Legend Dialog

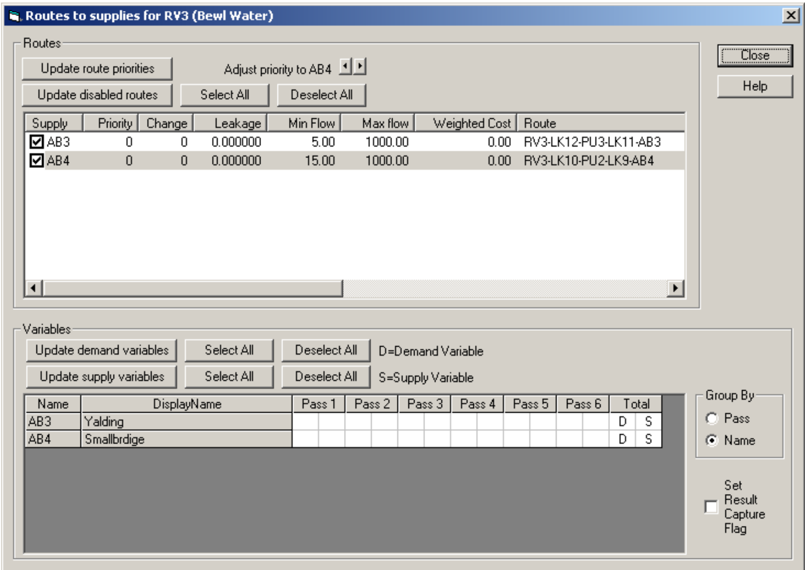


Use this dialog to customize the legend for a variable. This dialog is shown by clicking on any cell in the *Legend* column of a grid where variables can be set up. This includes any Variables grid in the *project tree* and the dialog shown by the *Setup...Chart Variables* menu command.

The default legend is constructed from the name of the variable and the name of the object or component to which the variable belongs. These names can become very long e.g. *Reservoir1.Annual Licence.Amout Left* and the appearance of the charts plotted at the end of a model run can be improved by choosing a more suitable legend string.

Click on the *Use Default* button to replace any custom legend with the default legend.

Setup Routes Dialog



This *Setup Routes* dialog is displayed either via the *Show Routes dialog* or by right-clicking on either a demand-type or supply-type component on the schematic and selecting *Routes* from the popup menu.

Most often the choice of route by which water will be moved can be made automatically at runtime by Aquator and the components. To alter the order in which routes are used, or whether they are used at all, is done using custom VBA code.

This dialog achieves its results by auto-generating VBA custom code for you to (a) alter route priority (b) disable one or more routes (c) create additional variables to calculate how much water is moved from a particular supply to a particular demand.

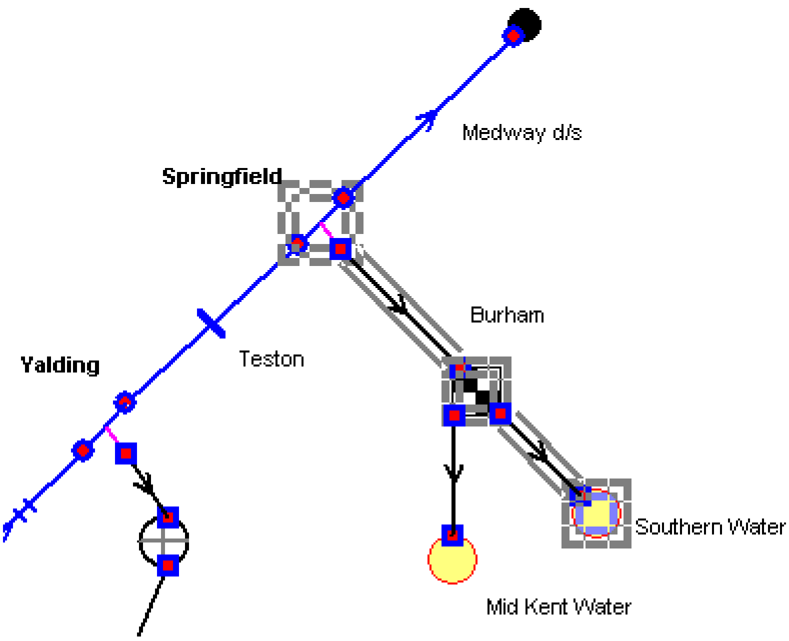
To alter a route priority highlight the route in the upper list, use the *Adjust priority to...* buttons to increase or decrease priority, and then click the *Update route priorities* button.

To disable selected routes untick the corresponding checkboxes in the upper list and then click the *Update disabled routes* button.

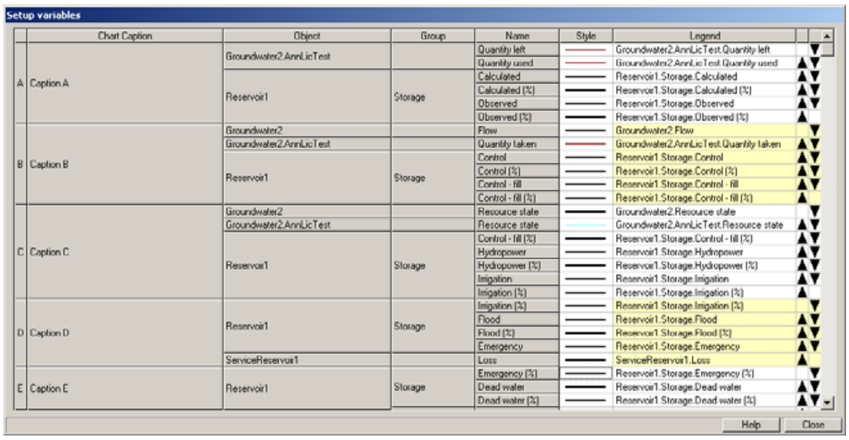
To add new variables to the demand click in the left-hand cell of the pair under "Pass 1", "Pass 2" ... "Total" until a 'D' is displayed and then click the *Update demand variables* button. To remove the variables just repeat these actions, this time making sure the 'D' is not displayed.

To add new variables to the supply click in the right-hand cell of the pair under "Pass 1", "Pass 2" ... "Total" until a 'S' is displayed and then click the *Update supply variables* button. To remove the variables just repeat these actions, this time making sure the 'S' is not displayed.

As each route is selected the components along that route are highlighted on the schematic, as shown in the following screen shot.



Setup Variables Dialog



This dialog is used to customize the overall appearance of each chart which is generated at the end of a model run, including the order in which each variable is plotted. Use the *Setup...Chart Variables* menu command or the *Chart Variables* toolbar button to show this dialog.

Before describing this dialog it may be useful to note that what variables are plotted, and on what charts, are all determined elsewhere, not by this dialog but in the *Variables nodes* in the *Project Tree View*. The intention is that the *Variables node* of each component is used to set each variable up, including which chart or charts to plot the variable on, and then this dialog customizes the chart, including the plot order for each variable.

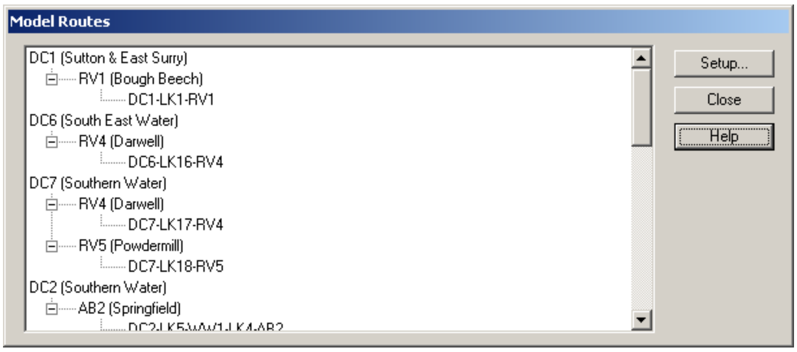
This dialog may be used to set:

- the chart caption
- the plot style
- the legend for each plotted variable

Simply click on the appropriate cell to set the corresponding quantity.

The arrows in the last two columns allow the order in which variables are plotted to be changed. Clicking on any arrow moves the row in the chosen direction, thereby changing the plot order.

Show Routes Dialog

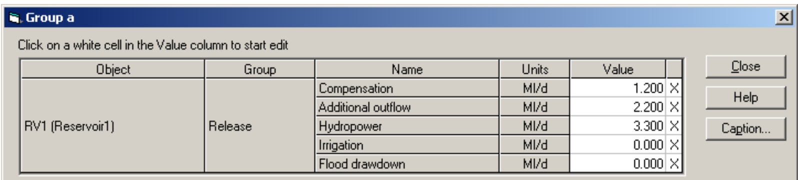


The menu item *Setup...Routes* brings up this dialog which shows for each demand-type component all the routes that lead to any of the supply-type components.

As each route is selected the components along that route are highlighted on the schematic.

When the *Setup* button is clicked the *Setup Routes dialog* is displayed for the selected demand-type component.

User-defined Editing Group Dialog



It is possible to put any parameter or state into a user-defined editing group, thus enabling the editing of diverse parameters or states in one place. See *User-defined parameter and state value editing groups*.

This dialog appears when one of these user-defined groups is being displayed. To edit a value click in a cell in the *Value* column in the usual way.

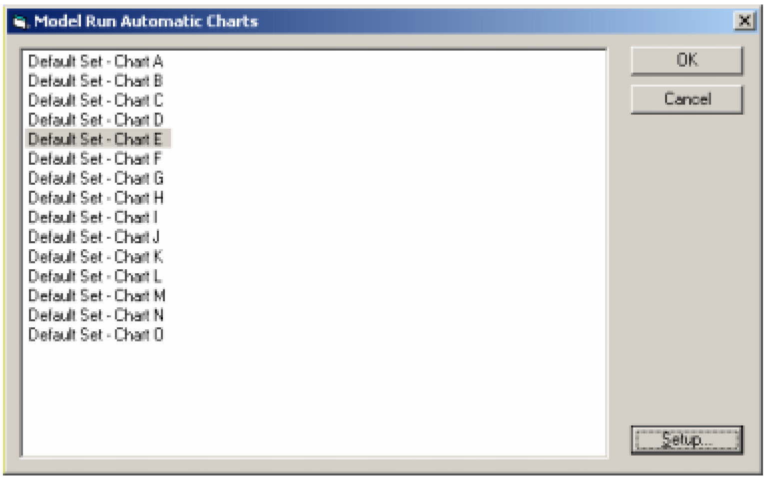
You can also click in the small cells in the last column to quickly remove any parameter or state from this group.

Click on the *Caption* button to change the dialog caption, which implicitly changes the name of the group.

Charts

Aquator provides a large number of options for customizing the graphical display of Results.

Charts Setup Dialog



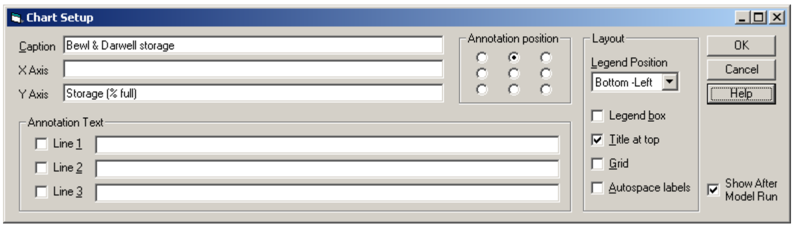
This dialog appears when you click on the *Charts* button in the *Project Sets Dialog*, which is accessed by clicking the project *variable* sets button on the *Project Node* page. The *Charts* button does not appear on the project sets dialog for parameters, states, or sequences because charts are only associated with variables.

At the end of a model run up to 15 charts can be automatically generated. A chart is only generated if at least one variable has been set to be plotted on that chart.

Charts are identified by an upper-case letter A to O. Each variable set may have an entirely different chart setup. In the above screen shot the *Default Set* is being customized.

Clicking on the Setup button brings up the *Chart Setup Dialog* which allows you to customize some details of each chart before it is plotted.

Chart Setup Dialog



This dialog is used to make permanent changes (if the project is saved) to any chart subsequently plotted at the end of a model run.

To show this dialog:

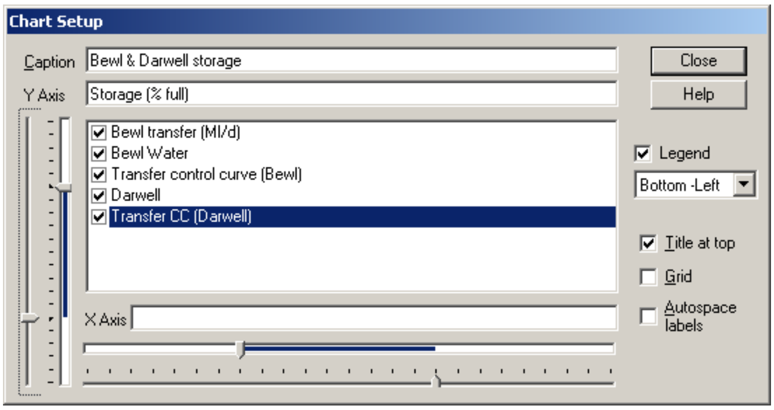
- either click on the *Setup* button on the *Variables* tab of the *Model Setup* dialog

- or click in the *Chart caption* column of the *Setup variables* dialog, shown by menu item *Edit...Chart variables* or by clicking on the corresponding toolbar button 

On this dialog you can:

- set a caption which will appear at the top
- add up to three lines of annotation which can be positioned on one of nine locations
- choose whether and where to place a legend, with or without a surrounding box
- optionally draw a grid instead of axis tick marks
- space out x-axis labels if they are becoming crowded
- suppress showing this chart after running the model

Chart Customize Dialog



This dialog appears when you click on the *Setup* button  in a chart node. Changes made in this modeless dialog will be immediately reflected by the chart being redrawn.

You can customize the chart in the following ways:

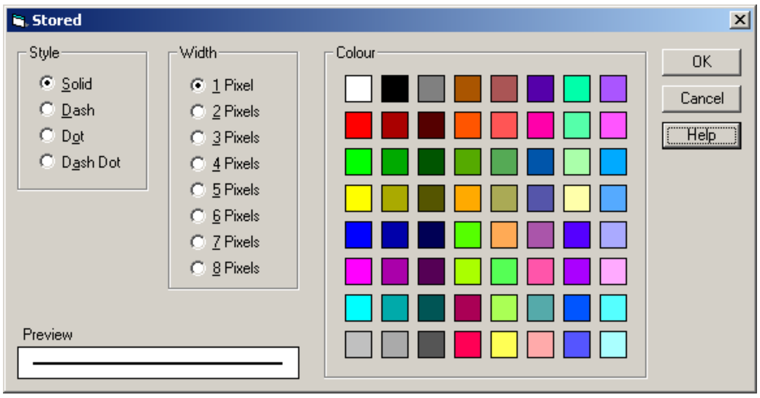
- set a caption which will appear at the top
- enter a string for the y-axis
- choose whether and where to place a legend, with or without a surrounding box
- optionally draw a grid instead of axis tick marks
- space out x-axis labels if they are becoming crowded
- suppress one or more lines by unchecking items in the plot list
- adjust the horizontal axis

- adjust the vertical axis

To adjust the axes

- drag the cursor of the lower horizontal (right-most vertical) slider to change the range of the x (y) axis
- drag the cursor of the upper horizontal (left-most vertical) slider to change the starting coordinate of the x (y) axis

Line Attributes Setup Dialog



This dialog is used to choose the style, width, and colour of lines used to plot results at the end of a model run.

Every variable in every component has its own style, width, and colour. In the grid where variables are set up (select any Variables node in the project tree) a column headed *Style* will show a preview of how the line will appear. Clicking on any cell in this column will show this dialog.

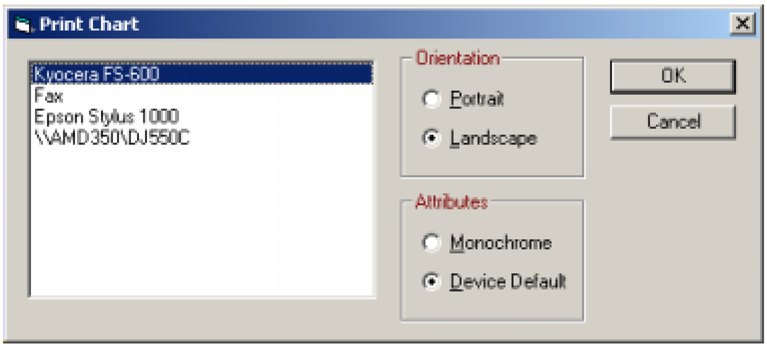
The style can be solid or one of three hatch patterns shown on the left.

The *Width* can be chosen from 1 to 8 pixels. Note that lines thicker than 1 pixel which are not solid are drawn very slowly. This is a very significant limitation for long model runs where plotting thousands of points in a hatched thick line can take many seconds.

The *Colour* of the line can be chosen by clicking one of the 64 coloured squares.

A preview of the how the line will appear is shown at bottom-left.

Chart Print Dialog



This dialog appears when you click on the *Print* button in a chart window. Choose the printer on which the chart will be printed and then click OK.

By choosing *Monochrome* attributes you can force a printout in black and white on a colour printer.

Chart Save Dialog

This dialog appears when you click on the *Save* button in a chart window. You can save the chart in a metafile for later importing into another application, but results vary.

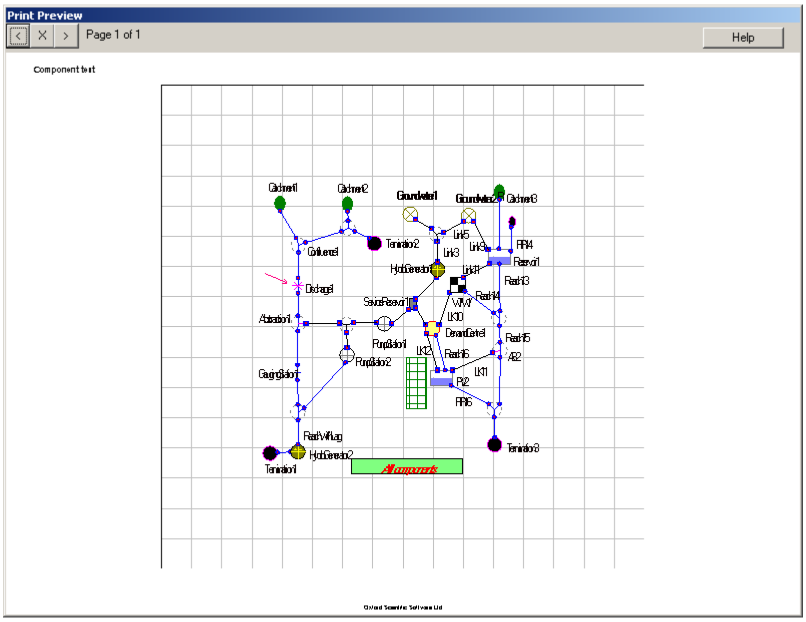
You can choose to save colour or monochrome attribute information. Choosing no attribute information will result in the import utility rendering colour information which can sometimes be useful.

CGM (*Computer Graphics Metafile*) format is highly portable but rarely gives very accurate results since many CGM import utilities are badly written. This includes those supplied with Microsoft Office. Setting the CGM compliance version to a low value usually improves results.

EMF (*Enhanced Metafile Format*) is the modern 32-bit Windows standard and will give accurate results when imported into another Windows application, but many applications still do not accept EMF files.

WMF (*Windows Metafile Format*) is the old 16-bit Windows format understood by almost all Windows graphics programs.

Chart Print Preview Dialog



This dialog appears when the *Print Preview* command is chosen from the File menu. It provides an *approximate* view of how each printed page will appear.

Aquator prints (or print previews) either the contents of the currently selected node on the project tree view, or, if tree view check boxes are visible and checked, then every checked node will be printed (or print previewed).

Aquator remembers from session to session which printer was last used and how the printer was set up. To alter this use the *File…Print Setup* command.

Previewing printed output consists simply of clicking the < and > buttons on the above dialog to scroll through each page. Click the *X* button to close the dialog.

See *Printing From Aquator* for more information about printing.

Aquator Project Properties

There are two ways to access and change project properties:

- on the *Properties* tab of the *Model Setup Dialog* which appears when you chose the *Setup...Model...* menu command,

- on the topmost *Properties Node* of the tree view of the project

Backdrop.Blend

This factor determines how much fading is applied to the backdrop i.e. how much the backdrop colours are mixed with white. This value will have no effect on older versions of the *Windows* operating system, which do not support blend.

Backdrop.Decimals

The number of decimal places to use when formatting user co-ordinates. The only use of this value is to display the *X, Y* mouse co-ordinate values shown on the status bar.

Backdrop.Height

The schematic height in user co-ordinates. The only use of this value is to compute the *X, Y* mouse co-ordinate values shown on the status bar.

Backdrop.Picture

The name of an image, preferably originally a metafile, used as a backdrop on the schematic. Click on the cell where the backdrop image name appears and use the resulting dialog to add, change or remove the picture.

Backdrop.Scale

Scaling factor to apply to backdrop picture.

Backdrop.Show

The schematic backdrop, if any, will be drawn if this flag is enabled.

Backdrop.User defined units

The string used to display user co-ordinates. The only use of this value is to display the *X, Y* mouse co-ordinate values shown on the status bar.

Backdrop.Width

The schematic width in user co-ordinates. The only use of this value is to compute the *X, Y* mouse co-ordinate values shown on the status bar.

Backdrop.X

The arbitrary X co-ordinate of the origin in user co-ordinates. The only use of this value is to compute the *X, Y* mouse co-ordinate values shown on the status bar.

Backdrop.Y

The arbitrary Y co-ordinate of the origin in user co-ordinates. The only use of this value is to compute the *X, Y* mouse co-ordinate values shown on the status bar.

Cell colour.Different value

On a grid comparing two lists of values this colour is used as the cell background colour to indicate where values differ.

Cell colour.Hidden item

On a grid comparing two lists of values this colour is used as the cell background colour to indicate values that are normally hidden (non-editable).

Cell colour.Missing item

On a grid comparing two lists of values this colour is used as the cell background colour when values in one list have no comparison value in the other list.

Component.Grab handle size

The size of the grab handles that appear around the box that highlights a selected component and which, using the mouse, can be dragged to change the component size

Component.Initial size

The initial default size of a component when it is added to the schematic.

Component Name.Bold if licenced

If this option is set then the label for any component which has one or more licences will be shown in a bold version of the font used for labels. In addition the text in the project tree view for the node which corresponds to a component with a licence will also be bold.

Component Name.Colour if customized

The label for any component which has VBA custom code will be shown in this colour, thus distinguishing customized and uncustomized components. In addition the text in the project tree view for the node which corresponds to a component with custom code will also be coloured.

Component Name.Show

This option can be used to globally turn on and off the display of component labels on the schematic. Note that this flag is ANDed with the individual *Options.Show Name* property of each component i.e. both the global flag and the individual flag must be true for a label to appear. The global flag does not affect annotations placed on the schematic.

Component.Selection line

The line drawing attributes used to draw the lines used for selection boxes.

Component.Selection mode

The logical drawing mode used to draw the component selection box. One option may give a better appearance than the other, dependent on the video display and colour setup.

Connector.Bidirectional colour 1

The colour used to draw around a bidrectional connector which is connected.

Connector.Bidirectional colour 2

The colour used to draw a bidrectional connector which is not connected.

Connector.Input colour

The colour used to draw input connectors, by default blue.

Connector.Output colour

The colour used to draw output connectors, by default red.

Connector.Radius

The radius of each connector on the schematic. Smaller values are neater but require greater precision when moving and joining.

Connector.Show

If enabled, connectors are drawn on the schematic, else connectors are not drawn.

Currency.Symbol

The currency symbol used for all costs and revenue. This can also be changed on the *Setup...Options* dialog *Currency* tab where a default symbol for new projects can be chosen.

Grid.Show

When enabled a schematic grid is drawn.

Grid.Line

The plotting attributes used to draw the schematic grid.

Model run.Log file name

This property holds the path to the file that optionally can be used to log water movements. If the *Model run.Log to file* property is set to *Every day* then the log file can grow to tens or even hundreds of megabytes for a large model or a long run, and will seriously slow execution.

Model run.Log to file

This enumerated property controls whether water movements are logged to an external file. By default it is *Off* but this can be changed to *Single step* or *Every day*. In the latter case the log file can grow to tens or even hundreds of megabytes for a large model or a long run, and will seriously slow execution.

Model run.Progress update interval

Can be set to Day, Month, or Year, and controls how often the progress indicators are updated during a model run.

Model run.Remove results at end

Can be set to Never, Ask, or Always, and controls whether the nodes generated during a model run are automatically deleted when the model run terminates. Useful during debugging and experimentation to prevent the treeview filling with large amounts of unwanted results.

Print.Preview scale

The scaling factor used for print preview i.e. the size of the print preview window page on the screen as a percentage of the page size. A value around 50-60% is usually optimum and allows the whole page to be previewed.

Print.Scale

The scaling factor applied to the printed page. A scaling factor less than 100% can squeeze more on a page. Use the *File...Print preview...* command to preview printed output.

Schematic.Font

The font used for the annotation of the schematic. Before printing or reporting it can be highly advantageous to experiment with scalable fonts such as Arial rather than use the default font.

Schematic.Hit test margin

This value determines the accuracy with which hit testing is performed. Small values require you to position the mouse more accurately.

Schematic.Left, Schematic.Top, Schematic.Width, Schematic.Height

The position and extent of the schematic working area. These values can only be altered by dragging the outlines shown on the *Properties* tab of the *Model Setup Dialog*.

Schematic.Locked

If set to *True* the schematic is protected against change. Components cannot be changed or moved. Connector symbols are drawn at half size when the schematic is locked.

Schematic.Max. zoom factor

The maximum zoom factor allowed when zooming in schematic view. By default zooming in corresponds to the *Page Up* key.

Schematic.Min. zoom factor

The minimum zoom factor allowed when zooming out in schematic view. By default zooming out corresponds to the *Page Down* key.

Schematic.Pan steps

The number of key presses of the horizontal or vertical cursor keys required to pan the schematic from left to right or top to bottom.

Schematic.Print orientation

The printer orientation used to print the schematic only. The printer orientation for other pages is determined by the printer setup.

Schematic.Print/Report extent

This option allows you to choose whether to print the visible portion of the schematic or the entire schematic. In the latter case this includes the white space around the components (see *Schematic.Schematic margin*). As the name implies, this property also determines how much of the schematic is reported by a Report tool e.g. pasted into Word when generating a Word report.

Schematic.Print/Report margin

If the printout or report scaling option (see *Schematic.Print/Report scaling* property) is to fit to the page then this property sets the margin which will not be used. The units twips or one-twentieth of a point i.e. 1440 twips = 1 inch

Schematic.Print/Report scale

If the schematic is to be printed at a fixed user-specified scale factor (see *Schematic.Print/Report scaling* property for alternatives) then this property provides the scale factor to use.

Schematic.Print/Report scaling

This provides three options: print at a specified scale factor (see also the property *Schematic.Print/Report scale*); print to fit page; or print at full scale (100%). If the option to fit to the page is chosen then the *Schematic.Print/Report margin* property also applies.

As the name implies, this property also determines the scaling used when a report is generated. If the option to fit to the page is chosen then the current Aquator printer page setting is used to determine the scaling. This may not be the same printer page setting in the application used to view the report.

Schematic.Schematic margin

The margin to leave around the base schematic area. Expressed as a percentage it can be used to expand the schematic area by placing components in the margin.

Schematic.Zoom intervals

The number of steps between minimum zoom and maximum zoom.

User-defined.Group N name

It is possible to put any parameter or state into one of 15 user-defined editing groups, thus enabling the editing of diverse parameters or states in one place. These properties set the names of the 15 groups See *User-defined parameter and state value editing groups*.

Aquator Project Parameters

There are two ways to access and change project parameters:

- on the *Parameters* tab of the *Model Setup Dialog* which appears when you chose the *Setup...Model...* menu command,
- on the topmost *Parameters Node* of the tree view of the project

Action.After max errors

This can be set to *None*, *Pause*, or *Stop* to take no action, to pause model run, or terminate model run, respectively, once the errors limit (*Limits.Max errors* parameter) has been exceeded.

Action.After max failures

This can be set to *None*, *Pause*, or *Stop* to take no action, to pause model run, or terminate model run, respectively, once the failures limit (*Limits.Max failures* parameter) has been exceeded.

Action.After max warnings

This can be set to *None*, *Pause*, or *Stop* to take no action, to pause model run, or terminate model run, respectively, once the warnings limit (*Limits.Max warnings* parameter) has been exceeded.

Additional pass.Distribute excess flows

if *True* then before pass 4 an additional algorithm is executed which distributes demand across multiple supplies simultaneously, in order of cost. This tends to use resources more uniformly. Although the overall numerical totals and averages are hardly affected the daily flows look more realistc when plotted.

Additional pass.Optimize minimum flows

If *True* then before pass 2 an additional algorithm is executed which attempts to (a) minimize how much water is moved to satisfy the minimum flow requirements, and (b) use all available supplies in proportion to their resource state

Additional pass.Smooth maximum flows

if *True* then before pass 5 an additional algorithm is executed which attempts to reduce unwanted oscillations in daily flows. Such oscillations usually occur when two or more supplies take it in turns to supply (most of) the required water. Although the overall numerical totals and averages are hardly affected the daily flows look more realistc when plotted, at the expense of extra execution time.

Advance order.Set manually

Allow manual setting of the advance order parameter of all demand-type components. See *Demanding water*.

Allocator.Minimum quantity

The smallest amount of water that will be allocated (moved). Quantities below this are truncated to zero and ignored.

Allocator.ProgID

This is the name (programmatic identifier) of the allocator tool, or algorithm, that decides how to share water among the competing demands. The allocator in use can be chosen from the drop down list of available allocators.

Aqua Solver.Check downstream

This enumerated value can be set to *Never*, *Abstraction setup*, or *Always*. It controls whether the *Aqua Solver* searches downstream of an abstraction for flow constraints. Normally leave set to *Always* since usually you will want flow constraints on the river to be included.

It is possible to configure an Abstraction component to ignore downstream flow constraints by setting the *Check downstream.Maximum flow constraints* parameter of the Abstraction to zero. In this case the *Aqua Solver.Check downstream* parameter may need to be set to *Abstraction setup* which means only check downstream of those abstractions whose *Check downstream.Maximum flow constraints* parameter is non-zero.

Note that choosing any setting other than *Always* can lead to failing to meet demand because the Aqua Solver will not have full knowledge of all constraints. In this situation some VBA may be needed to control one or more Abstraction components.

Aqua Solver.Demand excess water

This logical (Boolean) parameter controls whether excess water is used in the *Aqua Solver* solution in such a way as to minimise cost. Set to True to use excess water while minimising cost.

Excess water is that deemed to be over and above the normal or desired operating level of the supply, for example if a reservoir's storage lies above its control curve. Excess water is used in such a way as to minimise the cost of supply.

Either this parameter or the *Aqua Solver.Demand maximum water parameter* or both must be set to True. If both are set to True then maximum water is only used if excess water fails to meet demand.

Aqua Solver.Demand maximum water

This logical (Boolean) parameter controls whether the *Aqua Solver* solution can demand maximum water from supplies in such a way as to minimise the impact on resources. Set to True to use maximum water while minimising the impact on resources.

The impact on resources is measured by the *resource state* of each component.

Either this parameter or the *Aqua Solver.Demand excess water parameter* or both must be set to True. If both are set to True then maximum water is only used if excess water fails to meet demand.

Aqua Solver.Library

For very large models it may be appropriate to use a third-party solver library, an option labelled *FastLib* in the AquaSolver Setup dialog. Contact OSS Ltd if you would like to discuss taking out a licence to enable this option.

Aqua Solver.Logging

If this option is set to *Log water movements to file* then AquaSolver stores each days solution in the specified file.

The option *Pause when water movements differ from log file* may be useful on second or subsequent runs when investigating the effect of changes to the model parameters.

Aqua Solver.Max P4 demand if P5 used

When the system is under stress AquaSolver may be unable to find a solution which moves as much water as possible while minimising cost (pass 4), and then moving the remainder while maximising resource state (pass 5). In this situation a typical behaviour is to use pass 5 only and cost no longer influences the solution.

By setting this parameter to some value less than 100% AquaSolver will search for a solution that moves as much water as possible but less than the specified percentage while minimising cost (pass 4), and then moving the remainder while maximising resource state (pass 5). If successful, this allows the solution to be influenced by cost even when the system is under stress.

Aqua Solver.Satisfy leakage

This logical (Boolean) parameter can be set to False (leakage pass not executed before the *Aqua Solver* solution is calculated) or True (leakage pass is executed first).

Aqua Solver.Satisfy minimum demand

This logical (Boolean) parameter controls whether minimum demands (demands at an input connector of a demand centre that cannot be satisfied by water arriving on a different connector) are ignored or enforced in the *Aqua Solver* solution. Set to True to enforce these minimum demands.

Aqua Solver.Satisfy minimum flow

This logical (Boolean) parameter controls whether minimum flow constraints are ignored or enforced in the *Aqua Solver* solution. Set to True to enforce minimum flow constraints.

If enforcing minimum flow constraints causes demand to be unsatisfied, while relaxing these constraints means that demand can be met, then the minimum flow constraints are automatically relaxed, on a day-by-day basis, by *Aqua Solver*. This may still cause a model failure if a component is set up to fail if its minimum flow requirement is not met.

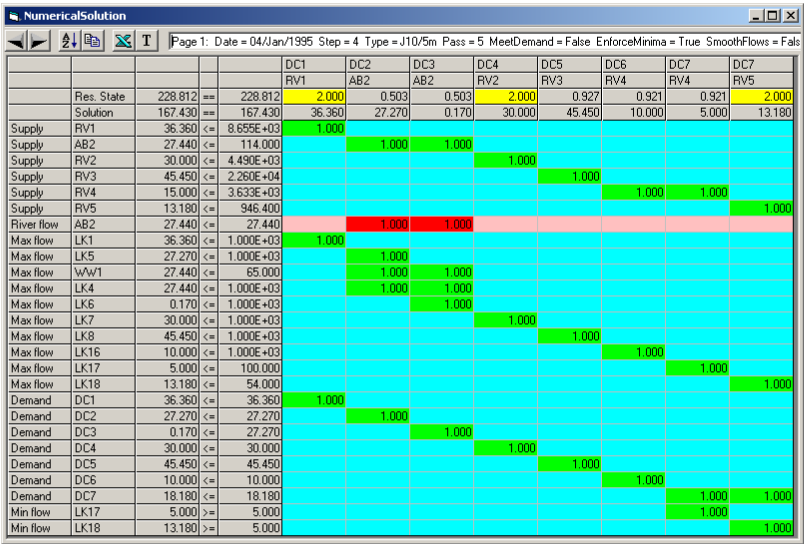
Aqua Solver.Show solution matrix, Aqua Solver.Show solution on, Aqua Solver.Identify components by

This advanced option is for users who wish to inspect the details of the numerical problem that AquaSolver is tackling.

To enable this facility, first check the *Display numerical solution for* checkbox and choose a date for which the solution matrix will be displayed. Optionally add a report to the model run outputs, showing the critical constraints on the system, by checking the

Add Aqua-Solver report to model run results for this day checkbox.

On the specified day a grid-based view of the numerical problem and its solution is presented.



					DC1	DC2	DC3	DC4	DC5	DC6	DC7	DC7
					RV1	AB2	AB2	RV2	RV3	RV4	RV4	RV5
	Res. State	228.812	==	228.812	2.000	0.503	0.503	2.000	0.927	0.921	0.921	2.000
	Solution	167.430	==	167.430	36.360	27.270	0.170	30.000	45.450	10.000	5.000	13.180
Supply	RV1	36.360	<=	8.655E+03	1.000							
Supply	AB2	27.440	<=	114.000		1.000	1.000					
Supply	RV2	30.000	<=	4.490E+03				1.000				
Supply	RV3	45.450	<=	2.260E+04					1.000			
Supply	RV4	15.000	<=	3.633E+03						1.000	1.000	
Supply	RV5	13.180	<=	946.400								1.000
River flow	AB2	27.440	<=	27.440		1.000	1.000					
Max flow	LK1	36.360	<=	1.000E+03	1.000							
Max flow	LK5	27.270	<=	1.000E+03		1.000						
Max flow	WW1	27.440	<=	65.000		1.000	1.000					
Max flow	LK4	27.440	<=	1.000E+03		1.000	1.000					
Max flow	LK6	0.170	<=	1.000E+03			1.000					
Max flow	LK7	30.000	<=	1.000E+03				1.000				
Max flow	LK8	45.450	<=	1.000E+03					1.000			
Max flow	LK16	10.000	<=	1.000E+03						1.000		
Max flow	LK17	5.000	<=	100.000							1.000	
Max flow	LK18	13.180	<=	54.000								1.000
Demand	DC1	36.360	<=	36.360	1.000							
Demand	DC2	27.270	<=	27.270		1.000						
Demand	DC3	0.170	<=	27.270			1.000					
Demand	DC4	30.000	<=	30.000				1.000				
Demand	DC5	45.450	<=	45.450					1.000			
Demand	DC6	10.000	<=	10.000						1.000		
Demand	DC7	18.180	<=	18.180							1.000	1.000
Min flow	LK17	5.000	>=	5.000							1.000	
Min flow	LK18	13.180	>=	5.000								1.000

Each colour-coded column represents a water movement to the demand whose name appears in the first row from the supply whose name appears in the second row.

Each colour-coded row represents a constraint on the system. Column 1 indicates the type of constraint and column 2 the component responsible. Column 5 holds the constraint value with column 3 showing how much water was actually moved. Column 4 shows whether the constraint is an equality, a minimum or a maximum type.

Cyan rows and columns indicate non-zero water movemments. A pink row indicates a constraint that is strictly limiting how much can be moved. And a red cell shows where a particular water movement (the column) is absolutely limited by a particular constraint (the row).

In the above example the demand at DC2 and DC3 cannot be met from AB2 because of a flow constraint at AB2.

Aqua Solver.Smooth flows
Aqua Solver.Smooth flows 2
Aqua Solver.Smooth flows 3

These logical (Boolean) values control whether up to three flow smoothing constraints are applied. A flow smoothing constraint requires that the flows from each supply for each day differ by less than a settable amount from the previous day – see the *Aqua Solver.Smoothing factor* parameter.

Up to three flow smoothing constraints can be enabled, which we number 1, 2 and 3 (the VBA name for the 1st omits the '1' character). Flow smoothing factor 1 is the largest while 3 is the smallest i.e. 1 gives the smoothest flows but is the most difficult to satisfy, while 3 smoothes flows the least but is the easiest to satisfy. *Aqua Solver* attempts to find a solution by applying each enabled flow smoothing factor in turn and choosing the best smoothing factor that still allows demand to be met.

The Aqua Solver may remove the flow smoothing constraints completely if it cannot find a smoothed solution that meets demand. It is worth experimenting with flow smoothing since some models are dramatically improved but in others it can have little, or even an adverse, effect.

Aqua Solver.Smooth warning

The logical (Boolean) parameter controls whether a flow smoothing failure generates a warning in the model run log.

Aqua Solver.Smoothing factor
Aqua Solver.Smoothing factor 2
Aqua Solver.Smoothing factor 3

These floating point values control the amount of flow smoothing applied. Each is a value between 0 and 1, where 0 is equivalent to flow smoothing off and 1 implies flows are constant day-to-day.

A value of 0.99 can provide very good results in some models. In others even much lower values do not seem effective and so experimentation can be useful.

Up to three flow smoothing constraints can be enabled, which we number 1, 2 and 3 (the VBA name for the 1st omits the '1' character). Flow smoothing factor 1 is the largest while 3 is the smallest i.e. 1 gives the smoothest flows but is the most difficult to satisfy, while 3 smoothes flows the least but is the easiest to satisfy. *Aqua Solver* attempts to find a solution by applying each enabled flow smoothing factor in turn and choosing the best smoothing factor that still allows demand to be met.

Aqua Solver.User defined demand

This logical (Boolean) parameter should be left set to False in all but the most specialised circumstances, due to the following considerations.

Normally the *Aqua Solver.Demand excess water* and *Aqua Solver.Demand maximum water* parameters are set to True, *Aqua Solver* produces a solution that completely determines the all water movements, and VBA customisation plays no part in modifying the *Aqua Solver* solution.

The *Aqua Solver* solution is optimal in the sense that it minimises cost and the impact on resources while either meeting demand or, if not possible, while moving the maximum amount of water available.

Theoretically it is possible to use VBA to modify the *Aqua Solver* solution. In that case demand may not be met and then the *Aqua Solver.User defined demand* parameter can be set True to allow one final stage of water movement under the control of VBA custom code..

Demand saving.On

If True then demand saving is enforced on the model. Demand saving groups are controlled separately by a similarly named parameter.

Demand saving.Level 1

This is the percentage reduction in demand at demand saving level 1. The daily demand saving level is set by the greatest demand saving level requested by any component subject to a minimum period at which the demand saving level is held constant.

Demand saving.Level 2

This is the *difference* between the reduction in demand at demand saving level 2 and at demand saving level 1, expressed as a percentage of the unreduced demand. The daily demand saving level is set by the greatest demand saving level requested by any component subject to a minimum period at which the demand saving level is held constant.

Demand saving.Level 3

This is the *difference* between the reduction in demand at demand saving level 3 and at demand saving level 2, expressed as a percentage of the unreduced demand. The daily demand saving level is set by the greatest demand saving level requested by any component subject to a minimum period at which the demand saving level is held constant.

Demand saving.Level 4

This is the *difference* between the reduction in demand at demand saving level 4 and at demand saving level 3, expressed as a percentage of the unreduced demand. The daily demand saving level is set by the greatest demand saving level requested by any component subject to a minimum period at which the demand saving level is held constant.

Demand saving.Level 5

This is the *difference* between the reduction in demand at demand saving level 5 and at demand saving level 4, expressed as a percentage of the unreduced demand. The daily demand saving level is set by the greatest demand saving level requested by any component subject to a minimum period at which the demand saving level is held constant.

Demand saving.Source

This determines the source of the *Demand Saving* imposed during the model run. Possilbe choices are:

1. **Component level**: The maximum of the Demand saving levels imposed by all components.
2. **Sequence level**: the Demand saving level taken from a *Time series* or *Profile*.
3. **Sequence factor**: the multiplying factor by which Demand is scaled, taken from a *Time Series* or *Profile*.

Disabled items.Notify daily during run

If left at its default value of *True* then disabled components are notified of each day's Initialize, Start, Finish, and Terminate events. For a project with a large number of disabled components significant reductions in model runtimes may be possible by setting this parameter to *False*.

Global optimiser.Enabled

This logical (boolean) parameter determines whether or not global optimisation is enabled. If False then global optimisation is off and Aquator models will run as in releases prior to V4.

If this parameter is set to True then global optimisation is on. Then the *Global optimiser.Name* parameter controls which global optimiser is used (currently, only the *Aqua Solver* global optimiser).

Global optimiser.Name

This enumerated parameter determines which global optimiser is used if the associated *Global optimiser.Enabled* parameter is set to True.

Currently the only valid name is "**Aqua-Solver**".

Limits.Max errors

This is the maximum number of errors allowed during a model run which, if exceeded and enabled, terminates a model run. An error is an indicator that a component encountered an unexpected condition, which prevented correct operation. Review the *Model Run Log* to see any errors that were generated.

Limits.Max failures

This is the maximum number of failures allowed during a model run which, if exceeded and enabled, terminates a model run. A failure is a condition such as a demand not being met, although all components may be operating correctly. Review the *Model Run Log* to see any failures that were generated.

Limits.Max warnings

This is the maximum number of warnings allowed during a model run which, if exceeded and enabled, terminates a model run. A warning is an indicator that a component encountered an unexpected condition but continued to operate correctly. Review the *Model Run Log* to see any warnings that were generated.

Model pass.Satisfy leakage

If *True* then pass 1 of the model run is executed. (See *Overview of the Model Run*).

Model pass.Satisfy minimum flow

If *True* then pass 2 of the model run is executed. (See *Overview of the Model Run).*

Model pass.Satisfy minimum demand

If *True* then pass 3 of the model run is executed. (See *Overview of the Model Run).*

Model pass.Demand excess water

If *True* then pass 4 of the model run is executed. (See *Overview of the Model Run).*

Model pass.Demand maximum water

If *True* then pass 5 of the model run is executed. (See *Overview of the Model Run).*

Model pass.User defined demand

If *True* then pass 6 of the model run is executed. (See *Overview of the Model Run).*

Options.Animate on

This global option controls whether the schematic is animated as the model runs. Turning animation off will reduce execution time. This toggle does not affect the individual component animate options but both must be enabled for animation of a particular component to occur.

Options.Animate single-step only

If True then the schematic is only animated while single-stepping. This can be useful in a complex model where animation can slow execution significantly.

Options.Capture parameters

If *True* then each model *Parameter* is captured at the start of the model run and stored in a *Model Run Node* for that model run.

If *False* then memory requirements will be reduced for complex models.

Options.Capture sequences

If *True* then each model *Sequence* is captured at the start of the model run and stored in a *Model Run Node* for that model run.

If *False* then memory requirements will be reduced for complex models.

Options.Capture states

If *True* then each model *State* is captured at the start of the model run and stored in a *Model Run Node* for that model run.

If *False* then memory requirements will be reduced for complex models.

Options.Capture version info

if *True* then complete information about the exact version of each software component is captured and stored with the rest of the model run information. Use this option to provide a complete audit trail.

Options.Capture VBA code

if *True* then all VBA customization code is captured and stored with the rest of the model run information. Use this option to provide a complete audit trail.

Options.Diagnostics on

This global option controls whether diagnostic output is generated as the model runs. Diagnostic output is only produced if this option is on *and* when single stepping the model, and only then from components whose individual *Options.Diagnostics on* parameter is also on.

Diagnostic output appears in a separate *Diagnostics* node under the model run node. It is terse but readable and is intended to log only the important stages in the calculations.

Options.Events on

This global option controls whether events are fired during a model run. Custom VBA code executes by trapping these events, hence this option enables or disables execution of custom code. This toggle does not affect the individual component events option but both must be enabled for a particular component to respond to events. Execution is slowed if components enable events unnecessarily i.e. when not customized.

Options.Forecasting on

This global option controls whether river regulators make releases of water which will be used either to maintain river flow or to support abstractions at a later date i.e. after the day the release is made. When a regulator queries downstream components for the amount to release, the query does not propagate past river reaches with non-zero time delay if forecasting is off.

Options.Trace on

This global option controls whether trace output is generated as the model runs. Trace output is only produced if this option is on *and* when single stepping the model, and only then from components whose individual *Options.Trace on* parameter is also on and whose *Options.Trace flags* parameter is non-zero.

Trace output appears in a separate *Trace* node under the model run node. It is verbose and barely readable and is intended to log every stages of the calculations no matter how trivial in order to assist in solving technical problems.

Route order.Algorithm

The route order algorithm used during model runs. See *Route ordering*.

V21 emulation.Licence excess bug

This parameter enables or disables emulation of the behaviour of Aquator V2.1 when returning to a licence an excess amount that was previously taken. This parameter is only visible if the project was originally built with an earlier (pre V3.0) version of Aquator.

Consider the following example. A component attempts to remove from each of its 3 licences an amount Q. The first licence allows Q, the second allows $R < Q$, and the third allows $S < R$. Then.an excess of $Q - S$ must be returned to the first licence and an excess of $R - S$ must be returned to the second licence. In Aquator V2.1 the third licence would incorrectly be returned an amount $Q - S$.

This parameter can be used to study the effect of this error, but all models should ultimately set this parameter to "**Off (no emulation)**" to ensure correct behaviour.

V21 emulation.Group RS update

This parameter enables or disables emulation of the behaviour of Aquator V2.1 when the resource state of group components is calculated. This parameter is only visible if the project was originally built with an earlier (pre V3.0) version of Aquator.

In V2.1 the resource state computed on any day did not take effect until the following day.

This parameter can be used to study the effect of this behaviour, but all models should ultimately set this parameter to "**Off (no emulation)**" to ensure correct behaviour.

V21 emulation.Group share fairness

This parameter enables or disables emulation of the behaviour of Aquator V2.1 when water is shared between components in a group. This parameter is only visible if the project was originally built with an earlier (pre V3.0) version of Aquator.

Water is shared when the total demand cannot be met. For two components in a group water is shared three times: once in each component and once in the group.

In Aquator V2.1 the group sharing allows for any sharing already done in each component. This is not the fairest procedure since then the group can cut back even more on the one component that has already been cut back.

If this parameter is set to "**Off (no emulation)**" then a fairer procedure is adopted whereby, if the group share is sufficient to satisfy the component share, then the components in the group are treated equitably.

V21 emulation.Model FinishDay timing

This parameter enables or disables emulation of the behaviour of Aquator V2.1 as regards the timing of the **FinishDay** event fired by the Model object. This parameter is only visible if the project was originally built with an earlier (pre V3.0) version of Aquator.

In V2.1 the model's **FinishDay** event is fired *before* the **FinishDay** events of the components are fired. This is inconsistent with the **TerminateDay** event which the model fires after the components.

For better consistency set this parameter to "**Off (no emulation)**". Then the **FinishDay** event fits into the general pattern:

InitializeDay and **StartDay**	Model event *before* components
FinishDay and **TerminateDay**	Model event *after* components

V21 emulation.Parameter change timing

This parameter enables or disables emulation of the behaviour of Aquator V2.1 when certain parameter vales are changed by custom VBA code while the model is running. This parameter is only visible if the project was originally built with an earlier (pre V3.0) version of Aquator.

The issue is whether other internal quantities that depend on the parameter are recalculated immediately, possibly part way through the calculations for one day, or whether the dependent quantites are recalculated at the start of the *next* day.

This applies for example to parameters that control the maximum abstraction at an *Abstraction* component, and to losses in a *Reach* component.

In case of doubt as to when changes take effect the best procedure is to devise the simplest possible model that includes the parameter to be changed and to experiment by changing it using the various events that occur during the calculations for one day.

V21 emulation.Reach net flow bug

This parameter enables or disables emulation of the behaviour of Aquator V2.1 when predicting the net flow at a reach, for the purposes of calculating how much water to add to the river e.g. to satisfy flow constraints and abstractions. This parameter is only visible if the project was originally built with an earlier (pre V3.0) version of Aquator.

In V2.1 the *Reach* component computed the predicted net flow by adding.to the current flow the predicted releases required upstream to satisfy any flow constraints, but incorrectly omitting from the calculation any predicted upstream abstractions.

This parameter can be used to study the effect of this error, but all models should ultimately set this parameter to "**Off (no emulation)**" to ensure correct behaviour.

Aquator Project States

There are two ways to access and change project states:

- on the States tab of the *Model Setup Dialog* which appears when you chose the *Setup...Model...* menu command,
- on the topmost *States Node* of the tree view of the project.

Demand saving hold (DSH).Method

This enumerated state value can take the one of the values: *None*. *Simple period*, or *Multi-level delays*. See *Demand saving hold*.

Demand saving hold.Start level

If *Demand saving hold (DSH).Method* is *Simple period* or *Multi-level delays* then this is the demand saving level at the start of the model run. This enables restarting on any day and getting the same results.

Multi-level DSH: [1-5].Delay

If *Demand saving hold (DSH).Method* is *Multi-level delays* then this is the delay in days that must elapse before the increase to the corresponding demand saving level is allowed. See *Demand saving hold*.

Multi-level DSH: [1-5].Hold

If *Demand saving hold (DSH).Method* is *Multi-level delays* then this is the hold period in days that must elapse before the corresponding demand saving level can be decreased. See *Demand saving hold*.

Multi-level DSH: [1-5].Days

If *Demand saving hold (DSH).Method* is *Multi-level delays* then this is the time in days spent so far at each demand saving level. See *Demand saving hold*.

Route holding.Enabled

If enabled then the ordering of the routes between demand-type components and supply-type components will only be allowed to change after the *Route Holding Period* has elapsed. Normally the 'best' route is chosen, where 'best' may mean cheapest or the route to the most abundant supply, but on a daily basis this can introduce artificial fluctuations in the flow of water. If route holding is enabled then these fluctuations will be reduced or eliminated.

Route holding.Period

This is the minimum number of days that must elapse before the ordering of the routes between demand-type components and supply-type components will only be allowed to change. Normally the 'best' route is chosen, where 'best' may mean cheapest or the route to the most abundant supply, but on a daily basis this can introduce artificial fluctuations in the flow of water. If *Route Holding* is *Enabled* then these fluctuations will be reduced or eliminated.

Route holding.Start day

This value is used to determine how many days into the route holding period have elapsed on the first day of the model run. This enables restarting on any day and getting the same results.

Run.Finish

This is the **Finish Date** of the model run.

Run.Start

This is the **Start Date** of the model run.

Simple DSH.Delay/hold period

If *Demand saving hold (DSH).Method* is *Simple period* then this is the minimum number of days that must elapse between changes in demand saving level.

Simple DSH.Day in period

If *Demand saving hold (DSH).Method* is *Simple period* then this is the number of days into the holding period at the start of the model run. This enables restarting on any day and getting the same results.

State capture.At end of run

If enabled the entire state of the model will be saved at the end of the model run as a new state set. This facilitates continuing a model run and obtaining identical results that would have been generated if the first run had been extended.

State capture.At specific date

If enabled the entire state of the model will be saved at the *State Capture Date* as a new state set. This facilitates restarting a model run at a particular date and obtaining identical results.

State capture.Date

This is the date during a model run at which the state of the model will be captured as a new state set.

State capture.New set name

This is the name which will be used when the state of the model is captured as a new state set during or at the end of the model run. The date will be appended to the name.

Aquator Project Variables

There are two ways to access and change project variables:

- on the *Variables* tab of the *Model Setup Dialog* which appears when you chose the *Setup...Model...* menu command,
- on the topmost *Variables Node* of the tree view of the project.

Aqua Solver.Actual supply

Total actual supply.

This value only applies to Advance Order = 1, see *Interaction with Advance Order and Regulation Order.*

Aqua Solver.Demand met

1 if demand was met, otherwise 0.

This value only applies to Advance Order = 1, see *Interaction with Advance Order and Regulation Order.*

Aqua Solver.Flows smoothed

The best flow smoothing factor that was applied. Up to three flow smoothing factors can be applied and the best one that still allows demand to be met is used.

This value only applies to Advance Order = 1, see *Interaction with Advance Order and Regulation Order.*

Aqua Solver.Iterations

The number of iterations used. Equivalent to the number of solution types examined.

This value only applies to Advance Order = 1, see *Interaction with Advance Order and Regulation Order.*

Aqua Solver.Minima enforced

1 if minimum flows were enforced, otherwise 0.

This value only applies to Advance Order = 1, see *Interaction with Advance Order and Regulation Order.*

Aqua Solver.Numerical errors

Error count of internal algorithm, should always be zero. Contact Oxford Scientific Software Ltd if this is not the case.

This value only applies to Advance Order = 1, see *Interaction with Advance Order and Regulation Order.*

Aqua Solver.Pass 4 used

1 if pass 4 was used, otherwise 0.

This value only applies to Advance Order = 1, see *Interaction with Advance Order and Regulation Order.*

Aqua Solver.Pass 5 used

1 if pass 5 was used, otherwise 0.

This value only applies to Advance Order = 1, see *Interaction with Advance Order and Regulation Order.*

Aqua Solver.Predicted supply

Predicted supply from *Aqua Solver*.

This value only applies to Advance Order = 1, see *Interaction with Advance Order and Regulation Order.*

Aqua Solver.Result code

Result code from *Aqua Solver* algorithm: 0 = OK, 1 = unbounded, 2 = insoluble. Contact Oxford Scientific Software Ltd if any value other than 0 is observed.

This value only applies to Advance Order = 1, see *Interaction with Advance Order and Regulation Order.*

Aqua Solver.Solution rating

Internal solution rating (1 is best)

This value only applies to Advance Order = 1, see *Interaction with Advance Order and Regulation Order.*

Aqua Solver.Solution type

Internal solution type code 1, 2....

This value only applies to Advance Order = 1, see *Interaction with Advance Order and Regulation Order.*

Aqua Solver.Total demand

Total actual demand.

This value only applies to Advance Order = 1, see *Interaction with Advance Order and Regulation Order.*

Demand saving.Level

The daily value of the demand saving level, an integer between zero (no demand saving) and five (maximum demand saving).

Demand saving.Percent

The daily value of the demand saving as a percentage of total demand in the absence of any demand saving.

Demands.Requested

Total demand for day across the whole model.

Demands.Shortfall

Demand shortfall for day across the whole model.

Demands.Supplied

Total supply to demands for day across the whole model.

Notifications.Errors

The daily total of the number of errors detected. An error is a condition under which a component cannot operate as intended. See the *Model Run Log* for information about errors generated during a model run.

Notifications.Failures

The daily total of the number of failures detected. A failure is a condition under which a component cannot satisfy its demand. See the *Model Run Log* for information about failures generated during a model run.

Notifications.Warnings

The daily total of the number of warnings detected. A warning is an unexpected condition but one under which the component can continue to operate as intended. See the *Model Run Log* for information about warnings generated during a model run.

Releases.Requested

Total releases required for day across the whole model.

Releases.Shortfall

Release shortfall for day across the whole model.

Releases.Supplied

Total supply to meet releases for day across the whole model.

Total.Added

The daily value of the total quantity of water entering the system due to inflow and rainfall.

Total.Cost

Daily cost of operation of the entire model.

Total.Leaked

The daily value of the total leakage of the system.

Total.Lost

The daily value of the total quantity of water lost from the system.

Total.Removed

The daily value of the total quantity of water leaving the system due to outflow, leakage, and evaporation.

Total.Stored

The daily value of the difference between the total quantity of water entering the system due to inflow and rainfall, less the total quantity of water leaving the system due to outflow, leakage, and evaporation.

Water balance.Amount

The daily value of the water balance of the system in megalitres, ideally zero, affected only by floating point round-off error. Long runs and large models may result in non-zero values which are insignificant compared to the totals involved, hence it is preferable to monitor the *Water balance.Percent* parameter.

Water balance.Percent

The daily value of the water balance of the system as a percentage, ideally zero, affected only by floating point round-off error. Values significantly different from a fraction of a percent may indicate a bug.

Aquator Project Sequences

There are two ways to access and change project sequences:

- on the Sequences tab of the *Model Setup Dialog* which appears when you chose the *Setup...Model...* menu command,
- on the topmost *Sequences Node* of the tree view of the project.

Demand saving factor

The sequence of Demand saving factors (between 0 and 1) that determine the Demand saving imposed daily during a model run.

Demand saving factor level 1

This sequence determines the demand saving factor if the demand saving level is 1. See *Demand Saving*.

Demand saving factor level 2

This sequence determines the demand saving factor if the demand saving level is 2. See *Demand Saving*.

Demand saving factor level 3

This sequence determines the demand saving factor if the demand saving level is 3. See *Demand Saving*.

Demand saving factor level 4

This sequence determines the demand saving factor if the demand saving level is 4. See *Demand Saving*.

Demand saving factor level 5

This sequence determines the demand saving factor if the demand saving level is 5. See *Demand Saving*.

Demand saving level

The *sequence* of Demand saving levels that determine the Demand saving level imposed daily during a model run.

Troubleshooting and FAQ

This section is intended to help the User when Aquator does not appear to behave in the expected way. It is set out in the form of **Frequently Asked Questions** (FAQ).

Questions about Constructing the Model

The following questions relate to building the model.

Why can't I connect my components?

You may be trying to connect an input to an input or a man-made connection to a natural connection point, see *Rules for Linking Components*, *p. 100*.

How do I add another connection to a component?

Right click on the component and select the appropriate connection type from the pop-up menu, see *Adding … Connection Points*, *p. 101*.

How do I know if a Link is connected to a Component?

The connection point should be a Red Square or Circle with a Blue border.

Why does the text not reproduce well when the schematic is printed or inserted into a report?

Change the font to a scalable font such as Arial and experiment with the font size. The font is one of the Project Properties.

How can I arrange my schematic to look tidy?

You can do the following to tidy-up your schematic:

1. Move components to the best location. The most effective way of doing this can be to import an existing schematic/map as a backdrop to the Aquator schematic. Aquator components can then be placed in the same position as the original schematic or put in the correct geographic location on the map.
2. Zoom into one area of the schematic to be tidied up.
3. Move connection points around component until the linking components are drawn a sensible angles (e.g. 0, 45, 90 degrees etc.).
4. Go back to step 2 for another area until the whole schematic is tidy.
5. Zoom out to 100% zoom factor.
6. Use the *Lock* Toolbar command *(see p. 105)*.

Questions about Model Run Results

The following questions relate to running the model.

How do I get graphical results from a run?

You need to ensure that you are capturing the required *Variable* and have set the Charts column, see *Setup Variables Dialog, p. 190*.

How do I display Demand Failures during a run?

Set the *Variable: Status Level* to be captured and charted graphically, see *Setup Variables Dialog, p. 190*.

Why no Time Series data during a model run?

You need to link the appropriate Time Series Data to the Component, see *Sequence Links Dialog, p. 184*.

Why no Demand?

The default parameter value for *Demand* is zero. You must set this parameter to the required value before running the model, see *Parameters Node, p. 110*.

Why no water movement in response to a demand?

The default parameter value for *MaxFlow* is zero. You must set this parameter to the required value on all links etc before running the model, *Parameters Node, p. 110*.

Why no Catchment flow?

You need to assign a *Time Series* or *Profile* to the Catchment component, see see *Sequence Links Dialog, p. 184*.

How It all Works – before Aquator Version 4

Prior to the development of the *Aqua Solve* global optimiser in Version 4, Aquator only executed local optimisation such that each component behaved independently when allocating water during a daily time step. It is still possible to run models this way by selecting *Local Optimisation*.

In this appendix we describe how Aquator works for models with global optimisation disabled (i.e. Aquator Version 3 and earlier).

Algorithms, rules and local optimisation

At the highest level, and ignoring a lot of detail to be described later, an Aquator water resource system model is designed to:

- Meet demand
- Enforce all rules, constraints and licences
- Minimise cost of operation when water is plentiful
- Minimise impact on resources when water is scarce

There is no restriction on the rules (algorithms) that can modelled. Aquator is the only software package of its type to incorporate *Microsoft® Visual Basic for Applications®* (VBA) which the modeller can use to craft bespoke algorithms.

VBA is the software industry's premier solution for end-user customisation, providing:

- A visually rich development environment with syntax colour coding, automatic code generation, *Microsoft intellisense* code writing assistance, and hundreds of pages of online help
- Complete compatibility with VBA in Microsoft Office
- Wide range of courses offered by many training companies
- Large pool of external consultants and companies available that provide programming development services using VBA

Usually there is no penalty to pay in execution time for modelling the most arcane non-linear or discontinuous rules and constraints.

Model Calculations – local optimisation

Aquator models a water resource system by combining a daily multi-pass calculation of how water is to be distributed together with the operating rules built into each component in the model.

This approach is intended to model real-world systems. It optimizes the allocation of water by allowing demands to first reserve and then take water. The reservation of water by all demands before any one demand actually takes water allows a sharing algorithm located on the supply to make decisions based on some kind of rule. This rule can be replaced or modified and might for example apply a simple reduction formula in times of shortage. Or it might apply a more complex algorithm taking into account alternative sources of water and the state of these resources.

Aquator models water resource allocation on a daily basis. Sub-daily calculations are done by some components, for example for minimising pumping costs by taking advantage of different electricity tariffs in each 24 hour period.

When the *Aqua Solver* global optimiser is disabled Aquator does does not attempt a mathematically optimal solution, instead each demand-type component independently chooses the 'best' source of water, in a way to be described. The advantages of this configuration are

- can model any type of non-linear interaction,
- can use VBA to modify how the model works at any time during the calculations.

The disadvantage can be that a lot of effort is required by the user to avoid failures to meet demand because of conflicts between different demands, conflicts that are automatically resolved (if at all possible) by the *Aqua Solver* global optimiser.

Overview of the Model Run – local optimisation

During each day of a model run water is moved according to the input data, the rules built into each component, and the connectivity (the *network*) of the model. There are five phases to such water movements.

1. Any catchments in the model add to river flows at the start of the day. Such flows may be controlled by a simple time series of values or possibly by a full catchment model running at the same time.
2. Any river regulators in the system may then augment river flows in order to satisfy river flow constraints and today's expected abstractions. If the forecasting option is turned on then releases may also be made to satisfy flow constraints and predicted abstractions on future days.
3. Demand centres then try to satisfy their demands by drawing water from any or all available supplies, such as river abstractions, groundwater abstractions, reservoirs, etc. Demand is typically specified by a profile of daily

demand values repeated each year, but can be defined in other ways.

4. Reservoirs refill as necessary from their available supplies according to built-in rules governing refilling
5. At the end of the day any reservoir which has had excess water pushed into it will spill into its attached river spillway, possibly leading to further spills from any other downstream connected reservoirs.

In phases 3 and 4 water is *pulled* from supplies to demands through what is called the *supply* part of the network. These movements of water are termed *supply-type*. Components lying between the demand and the supply can control, disallow, decrease, or even increase, the amount requested by the demand.

In the other phases water is *pushed* into the *river* part of the network. These movements of water are termed *flow-type*. Downstream components have no control once water has been pushed into the river.

The *river* and *supply* paths in the network may partially overlap. For example demands for water in phases 2 or 3 may be routed upstream along a river, entering at an abstraction and exiting at a regulator, finally terminating at a supply connected to the regulator. In other words Aquator allows the use of river reaches to transport water to satisfy demands.

This very high level overview omits much detail. The next sections give more comprehensive descriptions. First we describe how the model attempts to satisfy demand (phases 3 and 4). Then we describe river regulation (phase 2). This is followed by a discussion of the finer detail omitted for clarity in the previous sections.

Demanding water – local optimisation

There are two types of component that demand water in phases 3 and 4. Demand Centres represent the demands that drive the system e.g. towns, industry, etc. These demands are made during phase 3 and will typically reduce reservoir storage levels. Reservoirs act as demands in phase 4 when they attempt to refill. Because Demand Centres and Reservoirs act as demands in near-identical fashion the description of this section applies to both except where noted.

When a model run is initiated, Aquator determines the connectivity (the network) of the components that make up the system. Demands for water will be passed up the network from component to component, starting with a demand-type component and terminating on a supply-type component. During this initiation phase the order in which these demands will be made each day is determined.

Demand-type components are ordered in two ways. The **Advance Order** is usually determined automatically by Aquator and is so named because the model calculations are advanced in this order. The first components to demand water are Demand Centres and these all have an advance order of 1. Reservoirs immediately up-stream of demand centres will have an advance order of 2. Reservoirs upstream of reservoirs with an advance order of **N** will have an advance order of **N+1**. Components that do not demand water have an advance order of 0 (zero).

In a multiply-connected network it is possible to get a situation where for example reservoir **A** is upstream of both reservoir **B**, whose advance order is **M**, and also of reservoir **C**, whose advance order is **N**. In such a case reservoir **A** is assigned an advance order which is the larger of **M+1** and **N+1**.

Within each advance order, demand-type components are ordered by **Demand Order**. This user-settable value increments from 1 such that each demand-type component with the same advance order has a unique demand order. The simplest way to set the demand order is to open (Setup...Model) the *Model Setup Dialog* and click the up and down arrows inside the *Advance and Demand Order* box on the *Parameters* tab. Note the abbreviations AO for Advance Order and DO for Demand Order on this dialog.

The advance order of individual components can also be manually set if desired by ticking the "Allow Advance Order (AO) to be set manually" check box. If ticked then clicking the up and down arrows inside the *Advance and Demand Order* box on the *Parameters* tab of the *Model Setup Dialog* also adjusts advance order as well as demand order.

Whether the calculations depend on demand order is user-selectable. If water is not shared among all demands then demand order is important. Essentially this is a first-come first-served algorithm. Choose *Demand Order* for the *Allocation Algorithm* on the *Parameters* tab of the *Model Setup Dialog*. Choosing one of the other allocation algorithms forces sharing and hence demand order has no effect on the calculations.

Each day all the calculations for one advance order are completed before the calculations for the next advance order are started. Within each advance order, demand-type components request water in order of their demand order value (but as described in the previous paragraph this may or may not have an effect).

Each day water is moved in response to demands as follows:

- Each advance order is taken in turn, starting with advance order 1 (demand centres), finishing when all demand-type components have been done

- For each advance order: take water in six **passes** (described below)
- For each pass: each demand-type component in this advance first reserves and then takes water, using each **route** in turn
- The term **Pass** is used to denote water moved to satisfy a particular requirement. The six passes currently supported by Aquator are:
- **Pass 1**. Leakage. Water is taken to satisfy any leakage along each route. On this pass demand-type components actually request a zero quantity water but this request is increased to a non-zero value by any component on the route that has a non-zero leakage that has not yet been satisfied.
- **Pass 2**. Route minimum. Water is taken to satisfy any minimum flow requirement along the route. The minimum may not be satisfied e.g. if the maximum flow has been set lower than the minimum.
- **Pass 3**. Minimum demand. Demand centres can optionally have more than one input and each input can optionally specify that a certain part of the total demand can only be met by water entering on the specified input. This models situations where water cannot be freely transported across a city and pass 3 attempts to satisfy these requirements. This pass does not apply to reservoirs. See *Pass 3 Minimum Demands and Route Order*.
- **Pass 4**. Excess demand. Ideally the bulk of the required water is moved in this pass. Supply type components are allowed to release water that is in excess of their normal operating rules and constraints. Excess water is water that can be distributed while staying within normal limits.
- **Pass 5**. Maximum demand. If the previous passes fail to satisfy demand then on this pass, demand-type components request supply-type components to release additional water, up to the maximum available. This is typically more than their excess water i.e. it may break normal operating rules but it will typically still obey safety and any other overriding rule.
- **Pass 6**. User Defined. On this pass water is taken as determined by a user-defined algorithm coded using *VBA*.

Within each pass and advance order, water is first reserved by all demand-type components and then taken. This reserve-take operation is repeated in turn for each route that connects demands with supplies. The first route to be used is the optimum for each demand, where optimum depends on the pass. For pass

4 it typically means the cheapest route while for all other passes it typically means the route whose supply has the most excess water. The finer details of the sophisticated route ordering algorithm used by Aquator are described in *Route Ordering*. The amount of excess water that a source has available for Pass 4 depends on its **Resource State**. Sources and constraints report their Resource State to Aquator at the beginning of the day as a number that must be zero or greater. A Resource State of 1.0 means that the source is in its normal state for the current day. For example a reservoir reports a Resource State of 1.0 when the storage in the reservoir is exactly the same as its control line storage for that day. A Resource State of less than or equal to 1.0 indicates that there is no excess water available from this source for pass 4. A resource state of greater than 1.0 indicates that there is excess water available for pass 4.

The allocation of resources to meet demand is therefore driven by Aquator in a way that tends to equalise Resource State of all the sources. It is only when water is plentiful (i.e. Resource State>1.0) that cheapest water is used first.

All routes are used in turn, ordered in this way. All demands are allowed to use all routes if they wish. Once a demand has used all available routes on a particular pass it takes no further part until the next pass.

The multi-stage, multi-pass, multi-route organization of the demand portion of the model calculations may become clearer if it is expressed in pseudo-code as an algorithm. This is taken from the actual code used by Aquator but translated from the programming language to English.

```
For AdvanceOrder = 1 to the max. advance order of any demand
   For Pass = 1 to 6
      For RouteNumber = 1 to the max. no. routes to any demand
         For each demand-type component
            reserve water on route 'RouteNumber'
         Next component
         Optionally share water if reservations exceed availabilty
         For each demand-type component
            take water previously reserved
         Next component
      Next RouteNumber
   Next Pass
Next AdvanceOrder
```

A possible point of confusion is the interpretation of route number (RouteNumber in the above pseudo-code). On execution of the innermost loops, where each demand first reserves and then takes water, Aquator passes a route number to each demand.

This route number is an integer that starts at one and increments. Each demand then interprets this route number as an index into the collection of all routes that lead to itself. Effectively Aquator instructs each demand to use the next best route available to it, where 'best' means 'lowest cost' on pass 4 and 'highest resource state' on pass 5. A demand with fewer routes than other demands simply drops out of this process once it has utilised all its routes on any one pass.

Local Optimisation and Smoothing

The previous section described the optimisation algorithms employed by Aquator:

- Minimise cost of operation when water is plentiful
- Otherwise minimise impact on resources

and the multi-pass calculations performed each day:

- Pass 1. Satisfy leakage
- Pass 2. Meet minimum flows
- Pass 3. Meet minimum demands
- Pass 4. Meet demand while minimising cost
- Pass 5. Meet any remaining demand while minimising impact on resources

In addition some additional optimisation can be performed, as follows.

Pass 1 Additional Optimisation. Leakage is quickly satisfied by each demand requesting a supply of zero from each available supply. The requested amount of zero is increased by any component along the route which has non-zero leakage at zero flow.

However this can over-supply when the route from demand A to supply B intersects a route from demand C to supply D at a component with non-zero leakage. This is not usually a concern since leakage is usually small compared with the amounts of water moved on later passes. But at the expense of increased calculation time leakage can be satisfied while not moving more than the bare minimum required through multiple intersecting routes.

Pass 4 Additional Optimisation. If the cost of water from different supplies is the same then the order in which supplies are used does not depend on cost, which is the purpose of pass 4. In many cases this does not matter but it can lead to over-use of one supply compared to another.

The additional optimisation available on pass 4 uses all supplies of equal cost in the ratio of their resource state. The effect is to equalise the use of supplies with the same cost.

Pass 5 Additional Optimisation. On this pass supplies are used in order of their resource state. This can lead to a situation where on day N supply A is used in preference to supply B, which reduces the resource state of supply A on the next day, resulting in supply B being used on day N+1, supply A on day N+2, etc.

If this oscillatory 'hunting' behaviour is not acceptable an additional smoothing optimisation can be enforced which distributes demand across supplies in proportion to resource state. This has the dual effect of reducing hunting and equalising resource state as the model runs.

River Regulation – local optimisation

In phase 2 of the calculations water may be released from Regulator components into rivers in order to satisfy downstream flow constraints and/or downstream abstractions that will take place later, in phases 3 and 4, either on the same day or optionally on subsequent days.

Regulators demand water from the supplies to which they are connected. Therefore much of the previous section which describes how the model attempts to satisfy demands applies to regulators. For example, six passes are made, exactly as for demand centres and reservoirs. Regulators can use all routes to all supplies, ordered in the same way as routes to demand centres and reservoirs.

But regulators also differ significantly in two ways, which is why we describe them separately and why they are allocated a separate phase in the calculations.

Firstly, the order in which regulators make releases, the so-called **Regulation Order**, can be arbitrarily changed, whereas the **Advance Order** of demand centres and reservoirs is usually automatically determined.

Secondly, their demands are determined by *other* components whereas demand centres and reservoirs determine their demands internally.

The order in which regulators make releases can be controlled with complete flexibility via two parameters of each regulator: the **Regulation Order** and the **Demand Order**.

The simplest way to set these parameters is to open (Setup...Model) the *Model Setup Dialog* and click the up and down arrows inside the *Regulation and Demand Order* box on the *Parameters* tab. Note the abbreviations RO for Regulation Order and DO for Demand Order on this dialog. Some experimentation may be needed to understand how one pair of buttons control two parameters.

Regulators with different **Regulation Order** release water completely independently, at different times, lowest regulation order (1) first. Regulators with the same **Regulation Order** release water in order of **Demand Order**, which means they either compete for or share the available supply at the same time.

The discussion of **Demand Order** in the previous section therefore applies. If you choose an allocation algorithm that shares then **Demand Order** has no effect, otherwise its first-come first-served within the same **Regulation Order**.

Regulators query downstream components in order to determine how much water to release. A component such as a Gauging Station which has a flow constraint i.e. a constraint that dictates the minimum flow in the river, may request a release to augment the river flow. An Abstraction may also request a release in order to support any expected demands later in the day.

If forecasting is turned off then regulators only query downstream through river reaches with zero time delay. If forecasting is turned on this restriction does not apply and releases may be made in order to satisfy flow constraints and/or abstractions on days after the day on which the release was made.

Route ordering – local optimisation

The finer details of route ordering are necessarily complicated. The order in which supplies are used by demands can be critically important and so Aquator provides sophisticated algorithms and also allows you to customize the route ordering at several levels.

Versions of Aquator prior to V2.0 provided one route ordering algorithm that can be succinctly summarised as follows:

- on pass 4 (excess water) order routes by cost, lowest first
- on all other passes order routes by resource state, highest first

For a discussion of passes and resource state see *Demanding water*.

From Aquator V2.0 two route ordering algorithms are supported. The one to use is specified by opening (Setup...Model) the *Model Setup Dialog* and then selecting from the *Route order algorithm* dropdown list on the *Parameters* tab. The *Classic (V1.X)* entry enables the simple algorithm used by earlier versions of Aquator.

Selecting *Advanced (V2 on)* enables a more sophisticated algorithm to be used. This should be used for new projects and for old projects unless complete 100% backwards compatibility is required.

In this algorithm each pass has its own route ordering. Common to all is the new concept of **Priority** which overrides other parameters like cost and resource state.

By default **Priority** is an integer value equal to the number of rivers traversed by a route. The intention is to distinguish between routes that traverse a river and those that do not. Priority zero routes are used before priority one, one before two, etc.

You can override the default priority once, at the start of the model run, and the new priorities then enforce your own route ordering throughout the entire run. This requires some custom VBA code and is described in *How to use VBA with Aquator*.

Additionally you can override the choice of each route on each pass every day, on a per-component basis, again using custom VBA code. Therefore the following description of route ordering can be customized as much or as little as required.

There are eight parameters of a route that can affect route ordering. Not all are used on every pass.

1. Priority. Integer number. By default the number of rivers traversed by the route
2. AvailableVolume. Floating point number. Indication of maximum available amount of water. Might be further restricted by licences.
3. ResourceState. Floating point number. Excess water for pass 4 is available if greater than 1.0
4. ResourceStateValid. Boolean (True or False variable). False if the resource state cannot be calculated e.g. if the component's parameters or sequences that enter into the calculation of resource state have not been set
5. WeightedCost. Floating point number. Weighted cost per unit volume of moving water along the route
6. Leakage. Floating point number. Total leakage along route.
7. MinFlow. Floating point number. Largest minimum flow along route
8. DemandMinimaFraction. Float. See *Pass 3 Minimum Demands and Route Order*.

ResourceState and AvailableVolume interact according to whether ResourceStateValid is True or False. It is clearer to divide the routes into groups using the following criteria:

Group	**ResourceState/Available Volume**
Group 1	**ResourceStateValid** = False **AvailableVolume** > 0
Group 2	**ResourceStateValid** =True **ResourceState** > 1
Group 3	**ResourceStateValid** =True **ResourceState** > 0

Group 4	All other routes

The groups indicate how much water is available. Supplies whose routes are in group 1 can be drained empty on pass 4. For group 2 some water is available on pass 4, more on pass 5. For group 3 no water is available on pass 4 but some on pass 5. For group 4 there is no water available at all.

Then the route ordering for each pass is determined as follows.

- **Pass 1**. Leakage.

```
Order by Leakage <> 0 (True, False)
Within Leakage <> 0 subset
    Order by Priority
    Within each Priority
        Order by Group (Group 1 + Group 2, Group 3, Group 4)
        Within Group 1 + Group 2
            Order by WeightedCost
        Within Group 3
            Order by ResourceState
        Within Group 4
            Order is don’t care
Within Leakage = 0 subset
Order is don’t care
```

- **Pass 2**. Route minimum.

```
Order by MinFlow <> 0 (True, False)
Within MinFlow <> 0 subset
    Order by Priority
    Within each Priority
        Order by Group (Group 1 + Group 2, Group 3, Group 4)
        Within Group 1 + Group 2
            Order by WeightedCost
        Within Group 3
            Order by ResourceState
        Within Group 4
            Order is don’t care
Within MinFlow = 0 subset
    Order is don’t care
```

- **Pass 3**. Minimum demand.

```
Order by DemandMinimaFraction
```

See *Pass 3 Minimum Demands and Route Order*.

- **Pass 4**. Excess demand.

```
Order by Group (Group 1 + Group 2,  Group 3 + Group 4)
Within Group 1 + Group 2
    Order by Priority (0 = highest)
    Within each Priority
        Order by WeightedCost
    Within Group 3 + Group 4
        Order is don't care (no water available)
```

- **Pass 5**. Maximum demand.

```
Order by Group (Group 1, Group 2 + Group 3, Group 4)
Within Group 1
    Order by Priority (0 = highest)
    Within each Priority
        Order by AvailableVolume (largest first)
    Within Group 2 + Group 3
        Order by Priority (0 = highest)
        Within each Priority
            Order by ResourceState (largest first)
    Within Group 4
        Order is don't care
```

- **Pass 6**. User Defined.

```
Order should be user-defined as well
```

Pass 3 Minimum Demands and Route Order

Demand centres (DCs) can optionally have more than one input and each input can optionally specify that a certain part of the total demand can only be met by water entering on the specified input. This models situations where water cannot be freely transported across the geographic area covered by the DC.

These minimum demands are reserved ahead at the start of the model run by DCs requesting their supplies to make these look-ahead reservations. Supplies may allow none, some, or all of such requests, controlled by parameters of the supply.

The order of the routes used to make these requests, done once only at the start of the model run, is determined as follows:

```
Order by Priority
Within each Priority
    Order by MinFlow <> 0 (True, False)
    Within MinFlow <> 0 subset
        Order by WeightedCost
    Within MinFlow = 0 subset
        Order by WeightedCost
```

This ordering attempts to (partially) satisfy pass 3 minimum demands with pass 2 minimum flows, for efficiency.

Each route returns a **DemandMinimaFraction** parameter from these reservation requests, a number between zero and one which tells the demand how much of its demand minima the supply is willing to reserve. This **DemandMinimaFraction** parameter completely controls route order when water is moved on pass 3 each day.

Allocation Algorithm – local optimisation

Previous sections described how demands first reserve and then take water. The intention is to allow supplies to accumulate the reservations from all demands. Then when the first demand attempts to take water an optional sharing algorithm can be executed so that in times of shortage water is not simply taken on a first-come first-served basis.

Make a selection for the *Allocation Algorithm* on the *Parameters* tab of the *Model Setup Dialog* to choose which algorithm to execute.

At the time of writing three allocation algorithms were available.

1. **Demand order** First-come first served. In the absence of any customization the first demand to take water will have more available than others. Since the demand order of individual components can be set this algorithm is sometime useful in conjunction with some custom code that determines how much water to make available.
2. **Equitable** If the total amount reserved by all demands is not available then the amount taken by each demand is reduced by the same factor. The calculations no longer depend on the demand order of each demand.
3. **Equitable/Other sources** If all demands have other sources of supply not yet used then the behaviour is identical to the **Equitable** algorithm. But a demand with no other sources of supply is, if possible, satisfied completely before sharing the remaining water amongst the remaining demands. This takes place on Pass 5 only. The calculations do not depend on the demand order.
4. **Equitable/Other sources 2** This extends the capability of the **Equitable/Other sources** allocator described above. The enhanced “Other source” functionality here operates on both Pass 4 and Pass 5. This helps prevent failures that need not happen due to insufficient Pass 5 water being available. Unlike the **Equitable/Other sources** allocator, this allocator determines the amount of water other demands can get from other sources, not just the fact that they have other sources. Other sources that can supply the demands on the current allocation are also checked for available water and this may increase the amount that must come from the current source to avoid a failure. If there is not enough water to meet the demands, cut back occurs first on those demands which have some other water available. This cut back uses equitable

reduction. In this case sources with no other supply are satisfied in full first. Finally, if there is not enough water even to meet the demands that have no other alternative supply, equitable sharing is applied here also. This is the most sophisticated sharing algorithm and introduced in Aquator V3.

Demand Saving – local optimisation

In times of water shortage Aquator can optionally implement a multi-layered demand saving algorithm. These calculations are controlled by several model sequences and model parameters.

Here we provide a step-by-step explanation of how demand saving is calculated.

1. First we distinguish between demand saving level and demand saving factor. Level can go from 0 (no saving) to 5 (max saving) while factor can be 0% (no saving) to 100%. The original idea was to first determine level, then translate to factor. This gives the freedom to decouple the algorithm that determines level from the actual percentage. Demand saving can be applied globally, as a single number across the whole model, or through demand saving group components that allow different groups of demands to have different demand saving factors at any one time.
2. Under *Demand Saving Source* there are three options on the Sequences tab of the model setup and demand saving group setup dialogs. The third option (*Sequence factor*) is the simplest while the first option (*Component level*) is the most complex, so taking them in reverse order:
3. The project or demand saving group has one sequence named *Demand saving factor*. If the *Sequence factor* option (third option) is chosen and a time series is linked to this sequence then the demand saving factor for every day is taken from this time series, unless the value for the day happens to be missing. No other calculations are done i.e. all other demand saving factors and levels (sequences and parameters) are ignored. But if this sequence is not linked to a time series, or if the value is missing, then the demand saving is simply set to zero percent.
4. The project or demand saving group has a *Demand saving level* sequence. If the Sequence level option (second option) is chosen and a time series is linked to this sequence then the level for the day is taken from this series. If there is no time series, or the value for the day

is missing, then the demand saving level is set to zero, which implies a factor of zero percent.

5. If the *Component level* option (first option) is chosen then the demand saving level is the maximum demand saving level returned by any component in the model or demand saving group.This value is returned in the call to InitializeDay for each component (this call takes place before any water has been moved in the model for that day). In practice only reservoirs set a non-zero demand saving level at present.
6. To prevent any possibility of the demand saving level changing every day there is a *demand saving hold* option on the *States* tab. If enabled then after a demand saving change, the demand saving level cannot change again until the specified number of days have passed. Additionally the demand saving level is constrained to change by plus 1 or minus one only.
7. To translate level to factor, the appropriate project or demand saving group sequence *Demand saving factor - level N* (*N*=1 to 5) is examined. If the sequence is linked to a profile of factors then the value from the profile is used. If not, or if the value was marked as missing (unlikely for a profile), then the value of the project parameter *Demand saving.Level N* is used instead. In the Model dialog the relevant parameters can be seen on the *Parameters* tab, the sequences on the *Sequences* tab.
8. Finally, Aquator passes the now-calculated demand saving factor to every component in the model or demand saving group, as appropriate in the call to *StartDay*.

Demand saving hold – local optimisation

To constrain the demand saving level from changing too rapidly demand saving hold may be enforced. Essentially this puts a value greater than one on the minimum number of days that must elapse after a requested change in demand saving level before the demand saving level that is enforced is changed.

This is controlled by the *Demand saving hold (DSH).Method* state value that can take one of three enumerated values.

None No demand saving hold and the demand saving level may change as frequently as every day.

Simple period A single period is specified in days, which applies to all levels and to both increases and decreases in demand saving. The period is specified by the *Simple DSH.Delay/hold period* state value and the number of days so far at the current demand saving level by the *Simple DSH.Day in period* state value.

Multi-level delays Before a higher demand saving level is reached that level must have been requested for at least *D* days, where *D* is given by one of five state values *Multi-level DSH: 1.Delay* to *Multi-level DSH: 5.Delay*, one for each level. Similarly the demand saving level cannot decrease until at least *H* days that this level have passed, where *H* is given by one of five state values *Multi-level DSH: 1.Hold* to *Multi-level DSH: 5.Hold*, one for each level. Finally, the five state values *Multi-level DSH: 1.Days* to *Multi-level DSH: 5.Days*, one for each level, record how many days have passed at each level respectively.

Forecasting – local optimisation

The forecasting facility in Aquator enables the water moved on any one day to allow for river flows and abstractions on *future* days and not just the current day. In particular it enables Regulators (and Pump Stations acting as regulators) to add water to rivers to meet anticipated flow constraints and demands in the future.

By default the model parameter *Options.Forecasting on* is set to *False* and no forecasting is performed. This means each daily timestep is executed without regard for the state of the system in the future.

At the time of writing the only (non 3rd party) component with a time delay state is the Reach component, whose time-of-travel may be specified in days. If all reaches are set to have zero time-of-travel then forecasting will have no effect and should be left off.

Consider a Gauging Station with a maintained flow constraint, and a Regulator, on the same river but separated by one or more reaches with a total time-of-travel of T days where T > 0. To meet the maintained flow constraint on any day N, the regulator may need to add water to the river on day N – T.

To turn forecasting on take the following steps:

- set the model parameter *Options.Forecasting on* to *True*
- for a Regulator component, set its *Release.Support river* parameter to *True*; for a Pump Station to act as a regulator set its *Pumping.Amount* enumerated parameter to *Release (RO)* which is shorthand for "make a release in river regulation order"
- on the *Model Setup Dialog* (menu command *Setup...Model...*) set the order in which these regulator components are to add water to the river by using the buttons in the *Regulation and Demand Order* box on the *Parameters* tab

- for both Regulators and Pump Stations acting as regulators set the parameter *Release.Maximum delay* to the number of days ahead these components are to make forecasts
- also for both, set the parameter *Release.Maximum components* to an integer not smaller than the total number of downstream abstractions and gauging stations which are to be inquired as to whether they have forecast requirements for water; include abstractions and gauging stations that will respond negatively i.e. this parameter simply limits how many downstream components will be examined; setting this parameter and the one described in the previous bullet point to the smallest valid values may greatly improve execution speed
- for downstream Abstraction and Gauging Station components that are to participate in forecasting, set the parameters *Upstream releases.Minimum Delay* and *Upstream releases.Maximum delay* to values such that the time-of-travel from upstream regulators to these components lies between these limits; this enables you to fine-tune which regulators support which abstractions and gauging station flow constraints
- for downstream Abstractions whose predicted demands are to be supported by forecast releases by upstream regulators, set the remaining parameters in the *Upstream releases* group appropriately; the *Prediction method* enumerated parameter allows a choice of how to predict the demand in the future; you can use the *Prediction amount* parameter value, yesterdays abstraction, or yesterdays demand; finally the *Prediction scale factor* parameter is used to scale any prediction e.g. to allow for uncertainties, losses, etc
- both Regulator components and Pump Station components acting as regulators also have a parameter *Release.Scale factor* which can be used to scale up the sum of the releases requested by downstream components

This completes the actions needed to add forecasting to river releases. But this leaves open whether or not abstractions are to take forecasts into account.

Consider an Abstraction component and a GaugingStation component separated by reaches with a total time-of-travel of T days where T > 0. if the Gauging Station has a flow constraint then the Abstraction may optionally look at the forecast river flows T days ahead to decide if abstraction today will break a flow constraint in T days time.

There are two parameters on each Abstraction component which control this:

- the *Check downstream.Maximum flow constraints* limits how many downstream flow constraints will be taken into account; this can be used whether or not forecasting is on, and setting to the smallest valid value can greatly improve execution speed
- the *Check downstream.Maximum delay* only applies when forecasting is on; downstream flow constraints whose time-of-travel from the abstraction is greater than this value are ignored when the abstraction is deciding how much water it may supply; again, setting to the smallest valid value can greatly improve execution speed

Glossary of Terms

Advance Order

Order in which the calculations are advanced each day, specifically the order in which the different demands are satisfied. This is usually determined automatically by Aquator but can be manually set by the user.

Aquator Information Pane

The right hand window pane of the *Main Aquator Window* which displays information for the selected node in *the Aquator Tree View Pane*.

Aquator Menu Bar

Menu bar at the top of the *Main Aquator Window* which provides access to all the application menu commands.

Aquator Toolbar

The button toolbar at the top of the *Main Aquator Window* but beneath the *Aquator Menu Bar* which provides shortcut buttons for commonly used program actions.

Aquator Toolbox

The toolbar strip at the right hand side of *the Main Aquator Window* which contains all of the components that can be 'drag and dropped' on to the water system schematic.

Aquator Tree View Pane

The left hand window pane of the *Main Aquator Window* which lists all components and attributes for the project in a *Microsoft® Windows Explorer* format.

Aquator Thumbnail Pane

The lower-left window which shows a miniature view of the entire schematic.

Database

Single (.mdb) file containing each Project for a Water Resource System and the *Sequence* data.

Demand Order

Within each *Advance Order* and Regulation *Order*, demand-type components are ordered by Demand Order. This user-settable value increments from 1 such that each demand-type component with the same advance order has a unique demand order.

IDE

Integrated Development Environment, usually with reference to the VBA IDE where custom code is written and tested.

Flow

Water pushed into the river system.

Parameter

A value for a component or the model which affects its behaviour during a model run, but which typically does not change during a run, and often is not changed between runs. For example: the stage-area-volume curves of a reservoir would not normally be altered frequently.

Pass

The term used to denote water moved to satisfy a particular requirement.

Priority

A numerical value assigned to a route between a demand and a supply. Higher priority routes (smaller numerical value) are used first.

Profile

A sequence of numeric values that span exactly one year. A typical example would be a rainfall profile which consisted of the average daily rainfall from rainfall records of the last 50 years.

Project

Data which describes a Water Resource Simulation model. This will include each Component in the model, how it is linked to the other components and the *Property*, *Parameter*, *State*, *Variable* and *Sequence* values for all the components and the project. More than one project can be stored in a database.

Property

A value for a component or the model which has no hydrological significance. For example, the co-ordinates of a component on the schematic and the colours used to render the component's icon.

Regulation Order

Order in which regulators release water to rivers. This is determined by the user.

Resource state

A numerical value indicating the state of a resource on any day. A value of one (1.0) represents a resource in its nominal state. values greater than one indicate a resource which can supply excess water.

Scenario

A project scenario is a named collection of sets of *Parameters*, *States*, *Sequences* and *Variables*, with at least one set of each type.

Schematic

The diagram displayed on the *Aquator Information Pane* when the *Schematic Node* has been selected. It shows all of the components that have been added to the project and how they are linked together.

Sequence

A sequence is a requirement for daily data during a model run. Sequential data must be supplied to the component in order for the component to operate correctly. For example, a reservoir requires a rainfall sequence to compute water added by precipitation each day.

State

A value for a component or the model which affects its behaviour during a model run, and which typically does change during the run, and often is changed between runs. For example: the initial storage of a reservoir on the first day of the run would typically vary from run to run.

Supply

Water supplied to meet demands.

Time Series

A sequence of numeric values with a start date and an end date determined by what data has been imported into the database. A typical example would be a rainfall time series, which consisted of real-world measured values.

Variable

A value for a component or the model which is computed daily during a model run. This can include both predicted and observed values i.e. a sequence of observed storage linked to a reservoir component can be captured in a variable so that both observed and calculated storage can be plotted on the same chart.

VBA

Microsoft® Visual Basic® for Applications is the same software technology used in the *Microsoft Office®* suite of applications. With this tool the user can optionally customize the operating rules of any Aquator Component.