



Technical Documentation Version 6.2

Multiple Run Management



C A D S W E S

Center for Advanced Decision Support for Water and Environmental Systems

These documents are copyrighted by the Regents of the University of Colorado. No part of this document may be reproduced, stored in a retrieval system, or transmitted in any form or by any means electronic, mechanical, recording or otherwise without the prior written consent of The University of Colorado. All rights are reserved by The University of Colorado.

The University of Colorado makes no warranty of any kind with respect to the completeness or accuracy of this document. The University of Colorado may make improvements and/or changes in the product(s) and/or programs described within this document at any time and without notice.

DRAFT

Multiple Run Management Table of Contents

Introduction	1
Running a Pre-configured MRM model	1
Managing MRM Configurations	2
Creating a new configuration	2
Editing an existing configuration	3
Deleting an existing configuration	3
Copying an existing configuration	3
Setting Up A Multiple Run Configuration	3
Name	4
Description	4
Mode	4
Concurrent Runs.....	4
Consecutive Runs.....	5
Iterative Runs	6
Policy	9
Input	10
Initialization DMI.....	10
Input DMI Runs.....	10
Index-Sequential Runs	11
Combined Runs	13
Output	14
Saving and Restoring Initial Model State	16
Distributed Concurrent Runs	18
User Interface Overview	19
MRM Configuration.....	19
Remote Manager and Status dialog	20
Setting up the machines	21
Installing RwService.exe	22
Setting The Service Log On	23
How to Make a Distributed Run	23
Creating or changing a configuration	23
Re-running a configuration.....	24
How it Works	24
RiverWare Service.....	25
RiverWare Service Controller	26

RiverWare Remote Manager	26
XML Configurations	26
Combining RDF Files With CombineRdf.pl	30

DRAFT

Multiple Run Management

1. Introduction

Multiple Run Management is a RiverWare utility for setting up and automatically running many runs. The runs are set up through the Multiple Run Manager, which then carries out all runs and outputs the results into an RiverWare Data Format (RDF) and/or Excel file or through an output DMI.

A multiple run is defined as a set of one or more model runs, for all runs:

- The model configuration (object network) remains constant.
- The timestep size is constant.
- The same set of slots is saved to the output.

Runs are conducted automatically; i.e., RiverWare controls and invokes each run without user interaction.

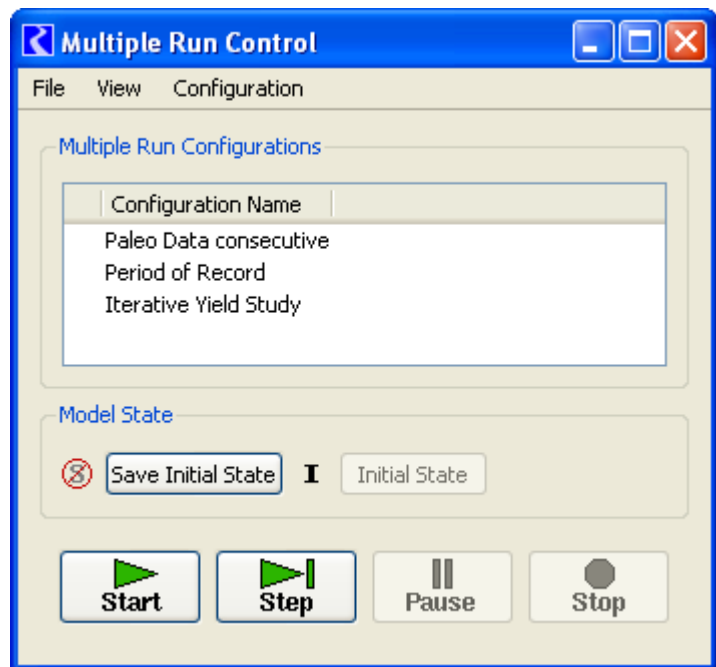
2. Running a Pre-configured MRM model

This section describes how to run a pre-configured MRM model. It is assumed that the MRM configuration is already set up and you wish to run the multiple runs and view the multiple run output.

- Open the MRM Dialog by selecting **Control** ➔ **MRM Control Panel...** from the main RiverWare menu bar or click on the MRM button on the toolbar.




- Highlight the desired configuration
- Press the start button
- After the run has finished, determine where the output was placed by selecting **Configuration** ➔ **Edit...** and clicking on the Output tab in the ensuing Multiple Run Editor dialog.
 - If there is a value in the **Control File** field this means that data was sent to the RDF file (or files) specified in the control file using the “file=” keyword. If there are no “file=” keywords specified in the control file, the data will go to the file specified in the **Data File** field. If the **Generate Excel files from RDF files** is checked, an Excel spreadsheet was also created in the same folder and by the same name as the control file.
 - If there is a value in the **DMI** field, this means that data was sent to a database using a DMI. Check the diagnostics output window for DMI diagnostics that show where output was sent. DMI slot diagnostics must be turned on during the run for this information to be printed.



For more information on output, see click [HERE \(Section 4.7\)](#).

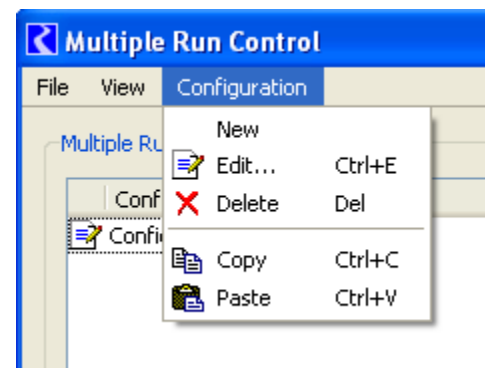
3. Managing MRM Configurations

This section describes how to manage MRM configurations using the Multiple Run Control dialog. The basics of creating, deleting, and opening a configuration for editing are described here.

- Open the MRM Dialog by selecting **Control** ➔ **MRM Control Panel...** from the main RiverWare menu bar or click on the MRM button on the toolbar. 

3.1 Creating a new configuration

- Select **Configuration** ➔ **New** from the Multiple Run Control Dialog



- Double click the newly created configuration (or highlight it and select **Configuration** ➤ **Edit...** or right-mouse click on the highlighted configuration to bring up a context menu and select **Edit...**)
- In the **MRM Configuration**, make any necessary edits to the new configuration and click **OK** (see “Setting Up A Multiple Run Configuration” section [HERE \(Section 4\)](#) for details)

3.2 Editing an existing configuration

- Highlight the desired configuration in the Multiple Run Control dialog and select **Configuration** ➤ **Edit...** or right-mouse click on the highlighted configuration to bring up a context menu and select **Edit...**
- Apply desired edits in the **MRM Configuration** dialog and click **OK**

3.3 Deleting an existing configuration

- Highlight the configuration to be deleted and select **Configuration** ➤ **Delete** or right-mouse click on the highlighted configuration to bring up a context menu and select **Delete**
- Apply the deletion by confirming the dialog

3.4 Copying an existing configuration

An easy way to create a new configuration similar to an existing one is by copying the existing configuration and making changes. Configurations can only be copy and pasted within one model (i.e., within one open session of RiverWare).

- Highlight the configuration to be copied and select **Configuration** ➤ **Copy** or right-mouse click on the highlighted configuration to bring up a context menu and select **Copy**
- Select **Configuration** ➤ **Paste** to paste the copied configuration into the open Multiple Run Control dialog or right-mouse click on the highlighted configuration to bring up a context menu and select **Paste**
- Highlight the “Copy of” configuration and select **Configuration** ➤ **Edit...** (or right-mouse click on the highlighted configuration to bring up a context menu and select **Edit...**)
- Change the name, if desired, and make any other edits. Click **OK** in the Multiple Run Editor.

Edit notations: Whenever a configuration is edited, the change is noted with an icon in the Multiple Run Control dialog until those changes have been either accepted or cancelled.



4. Setting Up A Multiple Run Configuration

This section provides the steps to setting up a multiple run configuration. Setting up a multiple run configuration requires specifying several parameters in the MRM Configuration. The parameters and brief descriptions are provided below.

- Open the **Multiple Run Control** dialog by selecting **Control → MRM Control Panel...** from the main RiverWare menu bar or click on the MRM button on the toolbar.
- Create a new configuration as described [HERE \(Section 3.1\)](#) and/or edit a created configuration as described [HERE \(Section 3.2\)](#).

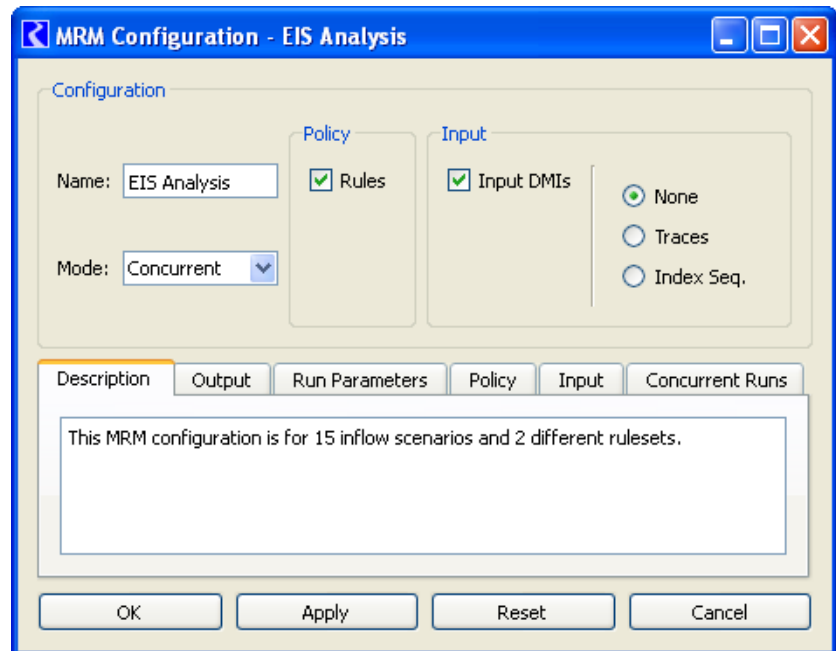


4.1 Name

Provide a unique name for the configuration in the **Name:** field.

4.2 Description

The configuration description may be multiple lines of text. This first non-blank line of the description appears in the MRM output file. This text is optional, and will not affect the MRM execution if left blank.



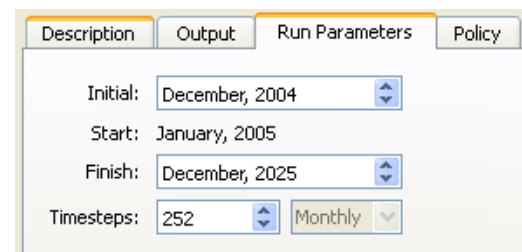
4.3 Mode

Select the **Mode** of the configuration (Concurrent, Consecutive or Iterative) from the **Mode:** menu. Selecting a mode will add the appropriate tabs to the dialog. Configuration of these tabs is described for each of the modes.

4.3.1 Concurrent Runs

Concurrent runs are multiple runs of which the time horizons are identical. The begin and end times, and timestep length, are the same for all runs. Concurrent runs are used to run the model multiple times with different inputs for each run. Inputs include rulesets (policy alternatives), input DMIs (i.e. series data like alternative hydrologies), and/or index sequential (data sampling technique).

- For concurrent mode, the **Run Parameters** tab is shown. The **Run Parameters** tab describe the initial and end date of each concurrent run. The run parameters also show the timestep size but timestep size is dependent on the model and can only be changed in the single run control dialog. Note, the Initial and Finish timestep for a concurrent run can be different than what is shown in the single Run Control.



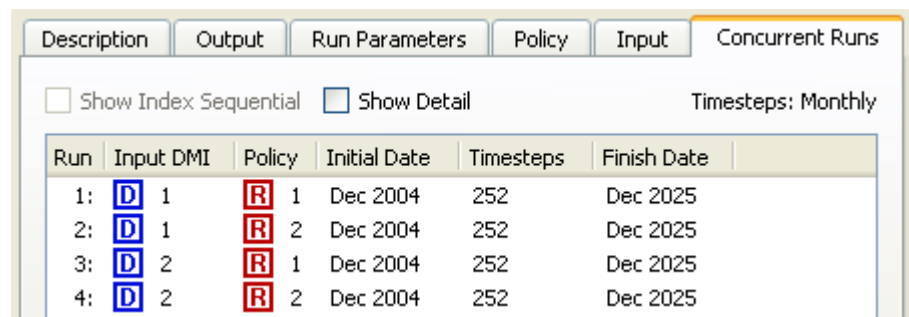
- The **Concurrent Runs** tab shows the number of runs that will be made.
 - In most cases, the total number of runs should be the product of the number of rulesets, the number of input DMIs, and the number of index sequential runs:

$$\text{\#of MRM runs} = \text{\#of Rulesets} * \text{\#of Input DMIs} * \text{\#of Index Seq Runs}$$

- If the Pairs option for DMI/Index Sequential Mode has been selected on the **Input** tab, the total number of runs should equal the number of pairs of Input DMIs and Index Sequential runs. If the number of input DMIs does not match the number of Index Sequential runs, then the number of index sequential runs equals the total number of possible pairs, (i.e., the minimum of the number of input DMIs and the number of Index Sequential runs). (See the Index Sequential / DMI Mode section for more details.)

$$\text{\#of MRM runs} = \min(\text{\#of Input DMIs}, \text{\#of Index Seq Runs})$$

Example of Concurrent Runs with 2 Input DMIs and 2 policy sets, no Index Sequential:

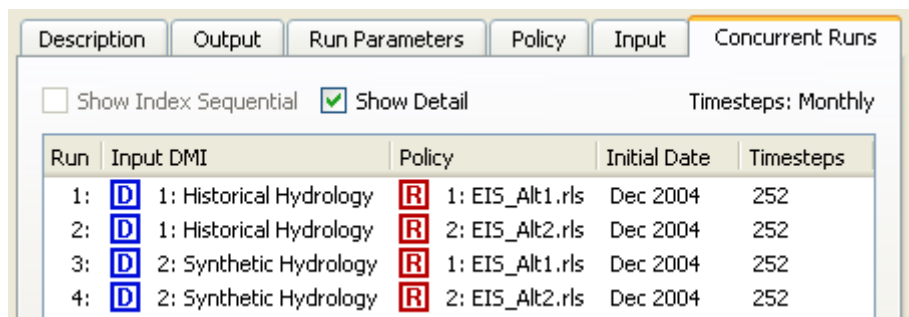


Run	Input DMI	Policy	Initial Date	Timesteps	Finish Date
1:	D 1	R 1	Dec 2004	252	Dec 2025
2:	D 1	R 2	Dec 2004	252	Dec 2025
3:	D 2	R 1	Dec 2004	252	Dec 2025
4:	D 2	R 2	Dec 2004	252	Dec 2025


Toggleing the Show Details box will show the names of the DMIs and policy sets:


4.3.2 Consecutive Runs

Consecutive runs are multiple runs where the time horizons are laid out consecutively. The end time of one run is the initial time of the next run. Timesteps do not vary among the different runs in a consecutive run.



Run	Input DMI	Policy	Initial Date	Timesteps
1:	D 1: Historical Hydrology	R 1: EIS_Alt1.rls	Dec 2004	252
2:	D 1: Historical Hydrology	R 2: EIS_Alt2.rls	Dec 2004	252
3:	D 2: Synthetic Hydrology	R 1: EIS_Alt1.rls	Dec 2004	252
4:	D 2: Synthetic Hydrology	R 2: EIS_Alt2.rls	Dec 2004	252










- On the **Consecutive Runs** tab, the **Edit** button indicates which fields in this dialog are editable.  If necessary, change the **Initial Date** for the consecutive runs. Do this by clicking on the existing initial date and toggling the up/down arrows in the ensuing date-time spinner.
- Change the number of timesteps for the first consecutive run by clicking on the existing number of timesteps and toggling the up/down arrows of the ensuing integer spinner. Notice that the **Finish Date** automatically updates.

- Append a new row for each desired additional consecutive run by clicking on the plus button. Or right-mouse click below the existing consecutive run and selecting **Append Row** from the ensuing context menu. Notice that the **Initial Date** is automatically set to match the **Finish Date** of the previous consecutive run. In the **Initial Date** column, only the first consecutive run can be changed. 

- Change the number of timesteps for each individual run, if necessary. The default is for newly appended runs to have the same number of timesteps as the previous run.

- To remove a run, click the minus button. This will always remove the last run.



Description		Output		Policy		Consecutive Runs	
<input type="checkbox"/> Show Index Sequential		<input type="checkbox"/> Show Detail		Timesteps: Monthly			
Run	Policy	Initial Date	Timesteps	Finish Date			
1:	 1	 Mar 2011	 12	Mar 2012			
2:	 1	Mar 2012	 12	Mar 2013			
3:	 1	Mar 2013	 12	Mar 2014			
4:	 1	Mar 2014	 12	Mar 2015			

4.3.3 Iterative Runs

Iterative runs are multiple runs where MRM rules at the beginning and/or end of each run examine the state of the system and, if appropriate, set values for the subsequent simulation run. If no values are set or the maximum number of iterations occurs, then the simulation ends. As in concurrent runs, the time horizons, begin and end times and timestep length are all the same for all runs.

The iterative runs can use any of the controllers as specified in the single Run Control dialog: simulation (with or without accounting), rulebased (with or without accounting) or optimization. If the run is rulebased or optimization, the same RPL set or Goal set, respectively, is used in each iteration.

An iterative run executes as follows:

1. Initialize the iteration count.
2. Execute the **Pre-MRM Run Rules**, if specified.

Note: **Pre-MRM Run Rules** are similar to Initialization rules for each run described [HERE \(Simulation.pdf, Section 5.1.2\)](#).

3. Perform a single run.
4. Execute the **Post-Run Rules**, if specified.
5. If the **Post-Run Rules** return “no change”, that is they do not assign one or more new (different) values, the iteration is complete.
6. Otherwise, the iteration count is checked. If it equals the maximum number of iterations specified, then the iteration is complete also.
7. If the iteration is not complete, then increment the iteration count and return to step 3 above.

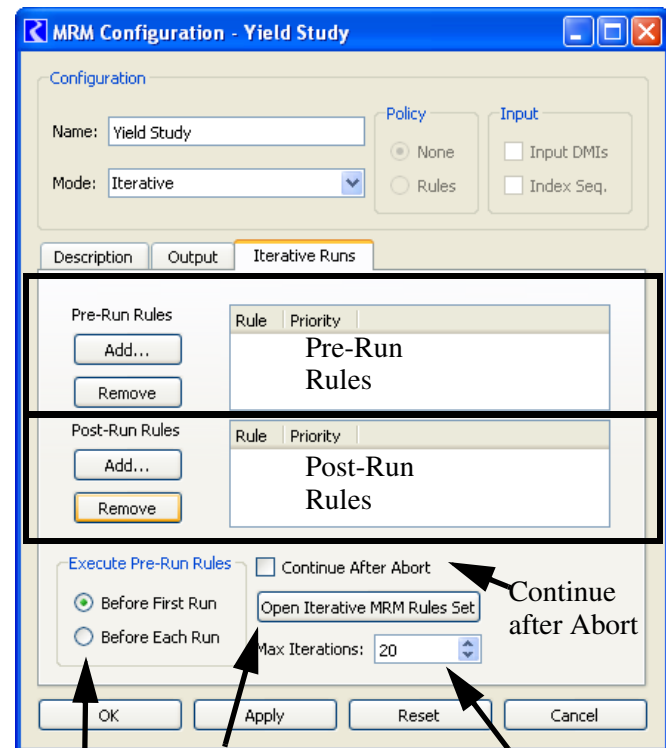
When an MRM rule, either pre-run or post-run, sets a value on a slot, the value is given the **i** flag indicating that it is a “Iterative MRM” flag. Values with the iterative MRM flag are cleared at the

beginning of an iterative MRM run but are not cleared between iterations of the MRM. This allows values set by the iterative MRM rules to persist between iterations but values are cleared at the beginning of the MRM run. In non-iterative MRM and single runs, values with an **i** flag are also cleared. The user should be aware of this behavior if switching from iterative MRM to another mode.

Values set by the Iterative MRM “i” flag behave with output semantics. That is, they can be overwritten by any other value. In rulebased simulation, they do not have a priority. As such, iterative MRM rules should only set values on data objects (typically integer indexed series slots as described at the end of this section), not simulation objects. Iterative MRM rules should be used to control the multiple runs; rulebased simulation rules within the run should be then used to set values on simulation slots that will actually be used in the run.

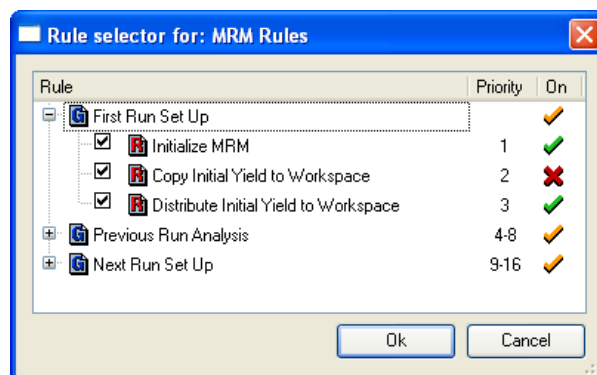
To configure an iterative run:

- Note that within the **MRM Configuration** dialog with the iterative mode, there is no Run Parameters tab. The iterative runs will begin at the start timestep of the model run. If users wish to change the model start timestep, this should be done in the single Run Control dialog.
- On the Iterative Runs tab are the following significant areas: the **Pre-MRM Run Rules**, the **Post-Run Rules**, the **Continue After Abort** toggle, **Pre-Run Rule Execution Time**, the **Open Iterative MRM Rules Set** button, and the **Max Iterations** spinner.
- Click **Open Iterative MRM Rules Set** button to open the Iterative MRM Rule set editor. This dialog operates very similarly to the standard RBS Ruleset Editor dialog. Create one (or more) Policy Group(s) and associated rule(s). The MRM Rules created are stored within the model file and may be available for use with any number of iterative MRM configurations. This set of rules can also be accessed from the workspace **Policy** ➔ **Iterative MRM Rules Set**.
- Once the MRM Rules have been built, return to the **Iterative Runs** tab of the MRM Configuration dialog. In this tab, the define which of the MRM rules should execute as part of the Pre-Run rules and which should execute after each iterative run, or both. These are defined in the appropriate areas using one of the following methods



Pre-Run Rule Execution time Rules RPL set Continue after Abort Open Iterative MRM Rules Set Max Iterations

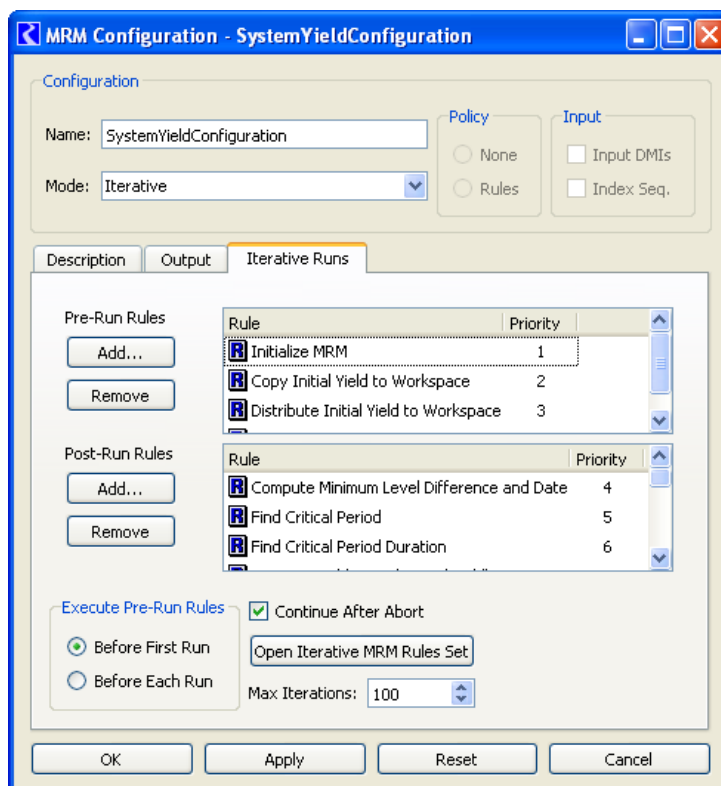
- Use the **Add** and **Remove** buttons. Click the **Add** button to bring up the **Rule Selector** as shown in the following figure. This dialog shows the rule groups in a tree view. Expand the tree-view to show individual rule names, their **Priority** and their **On** status. Click the box to check the desired rules. Click **Ok** to accept and return to the configuration dialog.
- Drag and drop rules from the MRM Ruleset to the **Pre-Run Rules** area or **Post-Run Rules** area.



Note: Note that the order of addition into the display does not affect the rule ordering; the order is strictly consistent with priorities defined in the MRM Rules.

- If you would like to view a rule directly from the **Iterative Runs** tab, double click on the rule's name to open the Rule Editor.
- Use the **Execute Pre-Run Rules** buttons to specify when **Pre-Run Rules** are executed:
 - **Before First Run:** This is the default. Choose this option to execute the **Pre-Run Rules** before the first single run only, but not before subsequent runs.
 - **Before Each Run:** Choose this option to execute the **Pre-Run Rules** before **each** single run.
- Consider activating the **Continue After Abort** option. If you want the **Post-Run Rules** to execute and possibly the next iteration to begin after an iteration is prematurely aborted (for any reason), check the box.
- Select the **Max Iterations** spinner and adjust it using the up / down arrows or by typing a value in the field to define the maximum number of iterations to run. A fully configured Iterative Runs tab is shown to the right.

Remember, the **Post-Run Rules** must make a significant change in values (for example, through a rule assignment or import of data through an input DMI) at the end of each run






for the controller to make the next iteration. If the **Post-Run Rules** do not set any values, (possibly due to convergence) the controller will assume that the goal of the iterations has been achieved and stop.

Also be aware that the MRM rules execute differently from the basic ruleset. When the MRM rules execute, each fires once and only once according to the execution order specified. There is no dependency functionality. In fact, they will all fire even if one of them aborts.

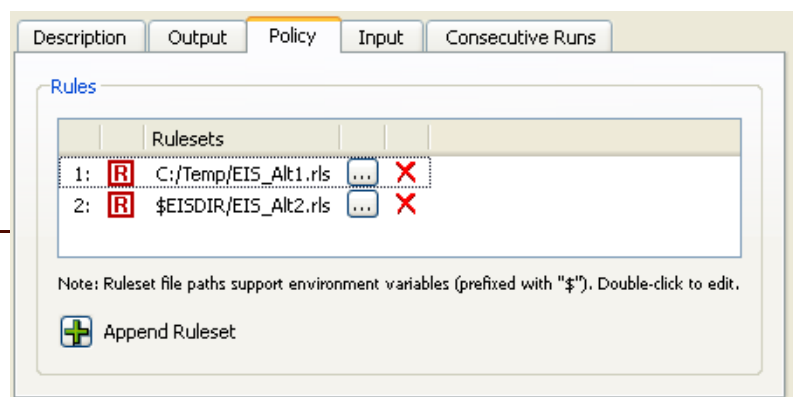
Integer indexed Series Slots work very well with Iterative MRM. Using these slots, the user can store inputs, outputs, or intermediate results based on the index of the run. For example, after each iterative run, the user could store the total volume of water released in an integer index slot in the row corresponding to the run index. At the end of the entire run, all of these volumes are stored and can be reviewed. The GetRunIndex predefined function can be used to get the index of the next iterative run. For more information on Integer Indexed Series, click [HERE \(Slots.pdf, Section 3.4\)](#). For more information on the GetRunIndex function, click [HERE \(RPLPredefinedFunctions.pdf, Section 71\)](#).

4.4 Policy

For concurrent and consecutive mode, ruleset(s) can be specified for the runs. The policy setup of the configuration (None or Rules) must be selected from the **Policy** section of the **MRM Configuration**. Selecting **Rules** will enable the Rulebased Simulation controller when the multiple run is started. Rulebased multiple runs are runs in which there are one or more rulesets. The user specifies the ruleset to use for each run. Variations in rules can be a function of differences in content, priorities, or both.

- Select **Rules** under in the **Policy** section of the **MRM Configuration** and click to the **Policy** tab
- Append a new row for each additional ruleset by clicking on the plus button  at the bottom.
- Select the ruleset by clicking on the file chooser button  or double click and type in the path name of the ruleset directly. This allows you to use environment variables in the path. Environment variables are prefixed with the “\$”.
- If necessary, remove a ruleset by clicking on the delete button . There is no confirmation, but you can “undelete” ruleset rows using the Reset button.

In the figure, the first ruleset was specified by browsing to the C: drive, the second ruleset was specified by typing the path in directly and using the environment variable EISDIR



Note: The Policy tab is not applicable to Iterative MRM runs, but an iterative run can be a rulebased run where one ruleset is used for all iterations. It must be explicitly opened and loaded before the iterative run is started.


4.5 Input

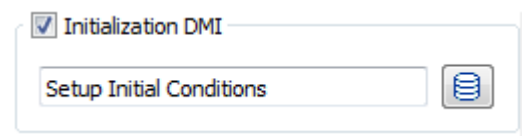
The inputs to the multiple runs are specified by toggling on options in the top of the dialog and then specifying configuration details in the appropriate tabs.

- If the Multiple Runs use an Input DMI, toggle the **Input DMIs** box in the **Input** section.
- Optionally specify one **Initialization DMI** described [HERE \(Section 4.5.1\)](#).
- Specify whether to use **Traces**, **Index Seq.**, or **None**. **Index Sequential** is described [HERE \(Section 4.5.3\)](#). **Traces** are described in the following section, Input DMI Runs.

Note: The Input section and tab are not applicable or available to Iterative runs.



4.5.1 Initialization DMI

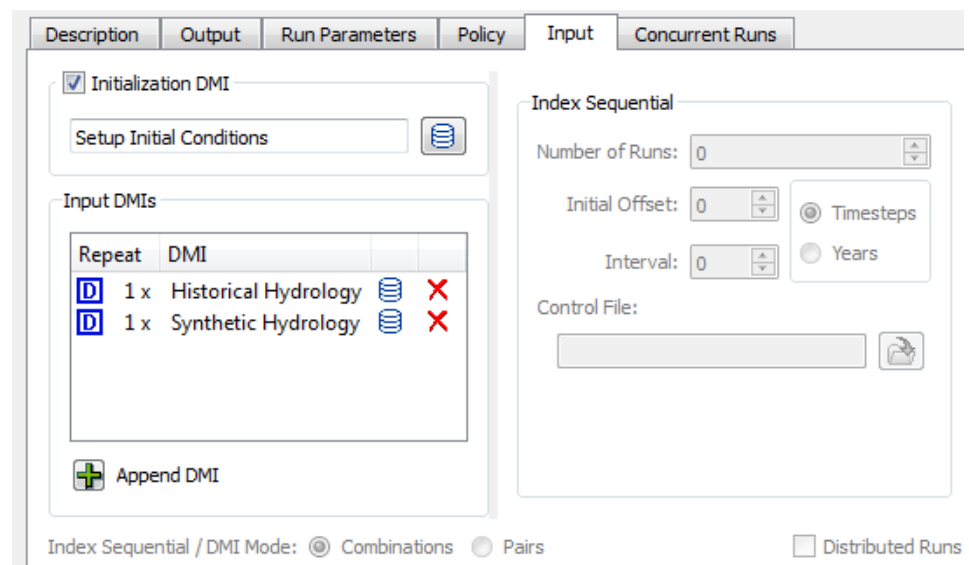
You can optionally specify a single DMI or group that is invoked at the beginning of the multiple run. Click the **Initialization DMI** toggle to show the entry field. Type in the name of an existing DMI or use the  button to select an the DMI or group.



4.5.2 Input DMI Runs

Input DMI runs vary the input data for a specified set of slots. For instance, ten runs might represent ten alternative release schedules for a reservoir. The user specifies the number of runs and a DMI for each run. The DMI loads the data into the model for each run. The DMIs must be previously defined in the RiverWare DMI interface.

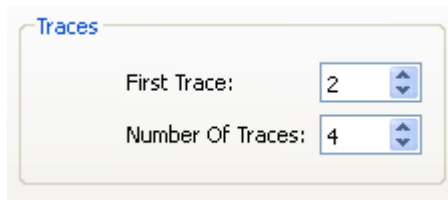
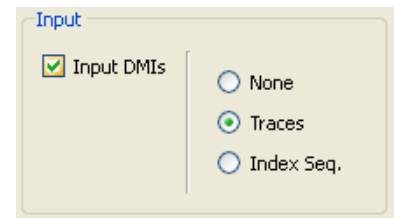
- When the Input DMIs is checked on the MRM Configuration, on the **Input** tab, the **Input DMIs** section becomes active.
- Append a new DMI by clicking on the plus button  at the bottom
- Select the input DMI by clicking on the DMI icon  and then choosing the previously configured DMI from the list.
- If the DMI will be called more than one time, change the number in the



Repeat column by double clicking on the cell and using the up/down arrows. DMIs might need to be called more than once in a MRM run, for example, with runs that execute multiple traces using the same DMI. In this situation, the DMI executable should reference the last command line parameter passed from RiverWare: `-STrace=traceNumber`.

- If you are using **Input DMIs** but not using Index Sequential, you can choose the **Traces** option from the Input section on the MRM Configuration. This allows you to run only a portion of the runs specified by the Input DMIs. When Traces are specified, the Input tab shows the Traces area. In this area, you can specify
 - the **First Trace**
 - the **Number of Traces**

If you wish to not run a subset of the runs, select **None**.

4.5.3 Index-Sequential Runs

Index Sequential runs use an input time series which is systematically shifted between runs. You specify the number of runs, the interval (number of timesteps) by which the input data is shifted from one run to the next, and an initial offset (number of timesteps) by which data is shifted for the first run. The slots with input data to be shifted are identified by in a DMI control file. This type of run is typically used to perturb (shift) historical hydrologic data in order to be able to conduct statistical analysis of the results.

Note: Index Sequential is not applicable to Iterative or Consecutive runs.

Index Sequential specifies that, given a run start time, run timestep, run duration, number of runs, and time offset, input time series, a run can be systematically perturbed between runs. For example, if an Index Sequential run is defined as:

- *input time-series:*
 - original value vector:* 1, 2, 3, 4, 5, 6, 7, 8, 9
 - original time vector:* Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep
- *offset:* 4 timesteps
- *interval:* 1 timestep
- *number of runs* = 3

then, the following sequence of runs occurs:

Setting Up A Multiple Run Configuration Input

Run 1: Jan = 5, Feb = 6, Mar = 7, Apr = 8, May = 9, Jun = 1, Jul = 2, Aug = 3, Sep = 4
 Run 2: Jan = 6, Feb = 7, Mar = 8, Apr = 9, May = 1, Jun = 2, Jul = 3, Aug = 4, Sep = 5
 Run 3: Jan = 7, Feb = 8, Mar = 9, Apr = 1, May = 2, Jun = 3, Jul = 4, Aug = 5, Sep = 6

To setup an Index-sequential run:

- Select Index Sequential in the **Input** section of the **MRM Configuration** and click on the **Input** tab

- Specify the **Number of Runs** by typing in the value or by toggling the up/down arrows
- Specify the **Initial Offset** by typing in the value or by toggling the up/down arrows
- Specify the **Interval** by which the data is shifted from one run to the next by typing in the value or by toggling the up/down arrows in the Interval field
- Specify the units for the Initial Offset and Interval as **Timesteps** or **Years**
- Specify the **Control File** by typing in the path directly or by using the file chooser to select it. Typing in the path and name directly allows for the use of environment variables. Take care to spell the entire path correctly. The control file specifies the slots that will be rotated for each run. It uses the same format specified [HERE \(DMI.pdf, Section 3.2\)](#) but no file name or keyword value pairs are necessary. Typically, it is just the object and the slots that are specified.
- On the **Concurrent Runs** tab, the combinations of policy sets and Input DMIs are set up for the index sequential run

Run	Input DMI	Policy	Initial Date	Timesteps	Finish Date
1:	D 1	R 1	Dec 2004	252	Dec 2025
2:	D 1	R 1	Dec 2004	252	Dec 2025
3:	D 1	R 2	Dec 2004	252	Dec 2025
4:	D 1	R 2	Dec 2004	252	Dec 2025
5:	D 2	R 1	Dec 2004	252	Dec 2025
6:	D 2	R 1	Dec 2004	252	Dec 2025
7:	D 2	R 2	Dec 2004	252	Dec 2025
8:	D 2	R 2	Dec 2004	252	Dec 2025

Note: The listed runs are repeated 2 times because of Index Sequential operations

- Toggle the “Show Index Sequential” box at the top of the **Concurrent Runs** tab to view each index sequential run listed individually

Run	Input DMI	Policy	Index Seq.	Initial Date	Timesteps	Finish Date
1:	D 1	R 1	IS 1	Dec 2004	252	Dec 2025
2:	D 1	R 1	IS 2	Dec 2004	252	Dec 2025
3:	D 1	R 2	IS 1	Dec 2004	252	Dec 2025
4:	D 1	R 2	IS 2	Dec 2004	252	Dec 2025
5:	D 2	R 1	IS 1	Dec 2004	252	Dec 2025
6:	D 2	R 1	IS 2	Dec 2004	252	Dec 2025
7:	D 2	R 2	IS 1	Dec 2004	252	Dec 2025
8:	D 2	R 2	IS 2	Dec 2004	252	Dec 2025

4.6 Combined Runs

By default, the runs that are made in concurrent MRM are a multiplicative combination of inputs. This set is called **Combined Runs**. Combined runs are defined as the Cartesian product of all the involved base-type runs. For example, a combined run consisting of two rulesets and four Input DMIs, results in $2 * 4 = 8$ model runs.

The order in which individual runs within a combined multiple run are executed is determined by the precedence level of each mode. Order of precedence (from highest (1) to lowest (3)) is as follows:

1. Input DMI runs,
2. Rulebased runs, and
3. Index-Sequential

Elements of lower precedence iterate before elements of higher precedence. For example, in case of a combined run containing two Input DMI runs and two Rulebased runs, the following sequence of four runs is made:

1. Input DMI 1 & Ruleset 1.
2. Input DMI 1 & Ruleset 2.
3. Input DMI 2 & Ruleset 1.
4. Input DMI 2 & Ruleset 2.

Thus, the default combinations mode results in the total number of runs equal to the product of the number of policy sets, the number of input DMIs, and the number of index sequential runs:

$$\text{\#of MRM runs} = \text{\#of Policy Sets} * \text{\#of Input DMIs} * \text{\#of Index Seq Runs}$$

In very specific circumstances, it is possible to alter the mode of how many MRM runs result from the combination of input DMIs, policy sets, and Index Sequential runs. If Index Sequential has been selected *and* there are as many (or more) Input DMIs as Index Sequential runs *and* there are **zero** or **one** rulesets selected, then it is possible to choose either **Combinations** or **Pairs** in the **Index Sequential / DMI Mode** selection

The **Pairs** mode results in the total number of runs equalling the number of pairs of **Input DMIs** and **Index Sequential** runs. If the number of input DMIs does not match the number of Index Sequential runs, then the number of index sequential runs equals the total number of possible pairs, (i.e., the minimum of the number of input DMIs and the number of Index Sequential runs). The **Pairs** mode is necessary to run the CRSS Lite model.

$$\# \text{ of MRM runs} = \min(\# \text{ of Input DMIs, } \# \text{ of Index Seq Runs})$$

When the pairs mode is specified, the runs can be distributed to multiple processors. For more information, click [HERE \(Section 6\)](#).

4.7 Output

During a multiple run, output can go to one or more RiverWare Data Format (RDF) text files and/or an output DMI. Click on the Output tab of the Multiple Run Editor. The following are configuration options:

- Control File:** Type or use the file chooser button to select a complete file path into the required control file field. The control file is used to specify which slots are output after each MRM run. The control file may contain “file =” specifiers for any line entries in the file. This causes data associated with those lines to go to the specified RDF output file. In the example control file below, if “fileName” is the same for each slot, then the output will go to a single RDF file; varying fileNames will send the output to multiple RDF files.

The screenshot shows the 'Output' tab of the Multiple Run Editor. It contains the following configuration options:

- RDF Options:**
 - Control File: C:/Temp/MRM_Output.ctl
 - Data File: C:/Temp/Reservoir_Output.rdf
 - Timesteps: Must Match (dropdown)
 - Allow Spaces In File Paths
- DMI:** (empty text field)
- Excel Options:**
 - Generate Excel Files
 - Delete RDF Files
 - Rows / Columns / Worksheets
 - Configuration: Timesteps / Runs / Slots (dropdown)
 - Slot Names: Index (dropdown)

```
MountainStorage.Inflow: file=fileName
MountainStorage.Outflow: file=fileName
MountainStorage.Storage: file=fileName file=fileName2
```

The user can optionally specify multiple files for a single slot as in the third line above. Also, slots with a timestep different than the model’s timestep can be output using the same syntax. The Timesteps control specifies whether RDF output files may contain slots whose timesteps differ. There are three choices:

- 1. Must Match** - All slots written to an RDF file must have the same timestep. Slots whose timesteps differ from the file’s timestep are skipped. (The file’s timestep is determined by the first slot MRM associates with the file.) The user is warned about slots being skipped, and asked whether to continue the MRM run.

2. **Use Smallest** - Slots written to an RDF file may have different timesteps, with the output written using the smallest timestep. For example, if monthly and yearly slots are written to an RDF file, monthly values will be written and the yearly slots will write 11 NaN followed by a value.
3. **Use Largest** - Slots written to an RDF file may have different timesteps, with the output written using the largest timestep. For example, if monthly and yearly slots are written to an RDF file, yearly values will be written and the monthly slots will write December values.

In the control file, there are four ways to specify where the data files are created:

1. Hard code the path: file=C:/DMI/Data/ResA.Inflow
2. Use an environment variable: file=\$(DMI_DATA)/ResA.Inflow
3. Use '~': file=~ /ResA.Inflow where '~' is replaced with \$RIVERWARE_DMI_DIR/<DMI name>
4. If the filename is not specified, the output file will be created in the directory in which RiverWare resides.

The second option is recommended as it is more transparent than '~' but still allows the model to be moved from machine to machine by setting the environment variable to the appropriate value on each machine.

- **Data File:** Optionally type or use the file chooser button to select a complete file path into the Data File field. This file is used as the RDF output file for any lines in the control file that do not have an explicit file specifier. If the data file field is used and there are no "file=" specifiers in the control file, then all output will go to this single data file.
- **DMI:** The optional DMI field allows selecting or entering an output DMI or a group of output DMIs that will be run after each single run of the multiple run. The DMIs must be previously configured in the DMI Manager in RiverWare. The executable that is configured for the DMI can reference the trace number of the run that is outputting by referencing the last command line parameter passed from RiverWare: -STrace=traceNumber
- **Excel Options:** Excel options are available on the Windows platform. Checking the **Generate Excel Workbooks** creates Excel files from all of the RDF output files after all runs have completed.

Check the **Delete RDF Files** to delete all RDF files deleted after the Excel files are created. (The separate RdfToExcel program contains the same functionality for creating Excel files from RDF files and can be used outside of RiverWare to process RDF files from an MRM run.)


Select a **Configuration** from the drop-down box to control how timestep, slot, and run data from RiverWare are mapped onto the Excel dimensions of rows, columns, and worksheets.

Select an option from the **Slot Names** drop-down controls to specify how slot names are written into the Excel workbook. Options are as

- **Index:** (Slot0, Slot1, etc.)
- **Short:** automatically shortened names (lower case vowels removed)
- **Full:** full slot names (limited to 31 characters for worksheet names, Excel's limit)

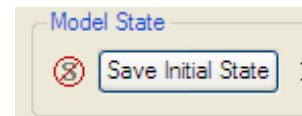
5. Saving and Restoring Initial Model State

This section describes how to save and restore the initial state of a model. The functionality allows users to save model data to a temporary file before a multiple run is invoked and then restore this information after the run has completed.

- Open the MRM Dialog by selecting **Control** ➤ **MRM Control Panel...** from the main RiverWare menu bar or click on the MRM button on the toolbar. 
- Save the initial state of a model before invoking a MRM run by pressing the **Save Initial State** button in the **Model State** area of the MRM Control Panel.
- Start the MRM run by pressing the **Start** button.
- After the run has finished, restore the initial model state by pressing the **Restore Initial State** button.

Pressing the **Save Initial State** button saves the entire contents of the model, including both input and output slots, TableSlots, selected user methods, run information, and configuration to a file in the user's temporary directory (defined by the TMP environment variable or system temporary variable as shown in the **Help** ➤ **About RiverWare** menu, then click on the **Show System Info...** button). Once the initial state has been saved, the button is disabled and reads “**Initial State Saved.**”

Save Initial State button prior to saving state.

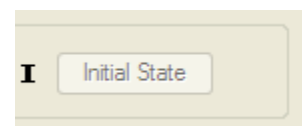


Save Initial State button after saving state.

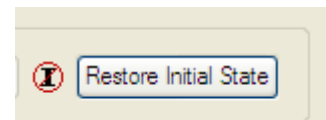
Once an initial state has been saved, it can be restored by clicking on the **Restore Initial State** button. This re-loads the entire contents of the model at the time its state was saved, clearing all data and configuration in the current state of the model. Until a multiple run has occurred, this button is disabled and reads “**Initial State**”.



Restore Initial State button before a multiple run.



Restore Initial State button after a multiple run.



- The MRM configuration must be in **Pairs** mode [HERE \(Section 4.6\)](#).

This document is organized to present an overview of the user interface, how to make a run, and how the utility works to distribute the runs.

6.1 User Interface Overview

The interface for distributing MRM runs across multiple processors consists of two components, the MRM configuration within RiverWare and the Distributed MRM dialog which is external to RiverWare.

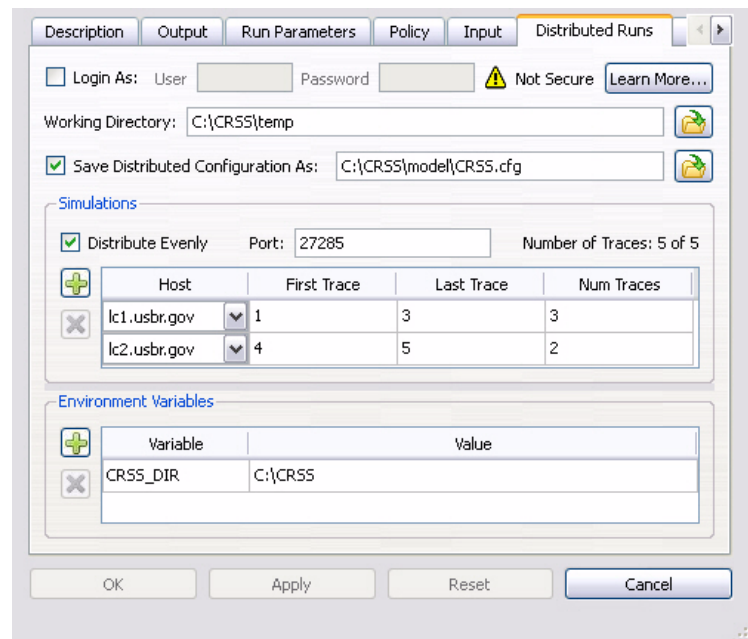
6.1.1 MRM Configuration

Currently, MRM runs can only be distributed when in Pairs mode, [HERE \(Section 4.6\)](#). Thus,

when in **Pairs** mode, the **Input** tab has a **Distributed Runs** check box that becomes enabled. When checked, the **Distributed Runs** tab is added to the dialog. As discussed in the sections that follow, the **Distributed Runs** tab allow you to configure:

- Whether a login is necessary, and optionally the user and password.
- The working directory.
- Whether the configuration should be saved to a file.
- The TCP/IP port.
- For each individual simulation, the host address, the first trace and the number of traces.
- Environment variables and their values.

Here's a screenshot of the Distributed MRM tab:



6.1.1.1 Login

If a login is required, check the box to enable the user and password fields. If either the user or password is omitted but required when the model is run, you will be prompted for it.

Note: Although it is not explicitly shown, entering the password here is **not** secure. The password is not encrypted in the model file, so anyone who has access to the model file can find the password. However, when a distributed run is made, the password **is** written across the network using a secure, encrypted TCP socket. Click the **Learn More** button to open a dialog explaining the security issues in detail.

6.1.1.2 Working Directory

A distributed concurrent run creates several “working” files - batch script files, control files, intermediate RDF files and log files among them. The working directory specifies where the working files will be created. Importantly, it should be a network directory which the controller computer and all simulation computers have access to (via the same path).

6.1.1.3 Save Configuration as

When a distributed concurrent run is started, RiverWare writes a configuration file, invokes the Remote Manager process, and exits. If a user elects to save the configuration to a named file, it is possible to invoke the Remote Manager directly bypassing RiverWare.

6.1.1.4 Host Table

The host table defines the simulation computers and the traces they will simulate.

Distribute Evenly: If “Distribute Evenly” is checked the traces are distributed evenly among the simulation computers, and the “First Trace” and “Num Traces” fields aren’t editable.

Port: The TCP/IP port the controller computer will use to connect to the simulation computers. It may vary by site based on firewall issues and other installed applications.

Number of Traces: The number of traces assigned to each processor. In the above screenshot 400 of 1000 traces have been configured. (The 1000 comes from the “Input” tab.) If the traces are “under configured” or “over configured” the text will be displayed in red.

Host, First Trace and Num Traces: For each processor, you specify the host name (typically the name of the machine you are on right now) and the range of traces it will simulate (unless “Distribute Evenly” is checked). The host name or IP address is entered or selected using an editable combo box (left and right, respectively).

RiverWare will remember previously entered host names and IP addresses, simplifying configuration. There will be a menu item to clear the host list, for example prior to distributing the model.

6.1.1.5 Environment Variables

You can enter environment variables and value pairs.

6.1.2 Remote Manager and Status dialog

As mentioned above, the Remote Manager includes a user interface which displays the status of the simulations. This dialog is not within RiverWare but is a separate executable in the installation directory. It is also opened automatically when you make a distributed MRM run from the RiverWare MRM Run Control. More specifically, the Remote Manager user interface allows you to:

- Start and stop the simulations individually or collectively.
- View RiverWare’s diagnostic output for the simulations.

- View the multiple and single run status.
- See an estimated time remaining.
- See the status of the post-processing (combining the RDF files).

The screenshot displays a dialog box with two rows of status panels. Each row contains three panels: 'Process Status', 'Multiple Run Status', and 'Single Run Status'. The top row shows a process on host 127.0.0.1 in a 'Running' state. The 'Multiple Run Status' panel for 'Traces 1 - 2' shows a progress bar at 50%, with 'Execution State: Running' and 'Current Run: 2 of 2'. The 'Single Run Status' panel shows a progress bar at 56%, with 'Execution State: Running' and 'Current Timestep: February, 2010'. The bottom row shows a similar process on host 127.0.0.1. The 'Multiple Run Status' panel for 'Traces 3 - 5' shows a progress bar at 33%, with 'Execution State: Running' and 'Current Run: 2 of 3'. The 'Single Run Status' panel shows a progress bar at 64%, with 'Execution State: Running' and 'Current Timestep: April, 2010'. At the bottom of the dialog, there is an 'Estimated Time Remaining: 00:00:26' and two buttons: 'Start All' and 'Stop All'.

From left to right are the “Process Status” panels, the “Multiple Run Status” panels and the “Single Run Status” panels. In better renditions, here is the “Process Status” panel with the “start”, “stop” and “view diagnostics” buttons. The buttons allow you to start, stop, and view diagnostics for the individual run/process.

The “Multiple Run Status” and “Single Run Status” panels (which are very similar to RiverWare’s run status dialog) display the progress of each MRM run and each individual run:

This close-up shows a 'Process Status' panel for host 127.0.0.1. The state is 'Finished Successfully'. It includes three buttons: a play button (start), a stop button (red circle), and a document icon (view diagnostics).

This close-up shows two panels: 'Multiple Run Status (Traces 3 - 5)' and 'Single Run Status'. The 'Multiple Run Status' panel shows a progress bar at 33%, with 'Execution State: Running' and 'Current Run: 2 of 3'. The 'Single Run Status' panel shows a progress bar at 64%, with 'Execution State: Running' and 'Current Timestep: April, 2010'.

The bottom of the dialog shows the estimated time remaining based on available data after the first run has completed.

Estimated Time Remaining: 00:00:26

6.2 Setting up the machines

Before making a run, you must set up the machine. This section assumes you have installed RiverWare on each machine and it is located in “C:\Program Files\CADSWES\RiverWare 6.2”. Following are instructions to set up each machine for the distributed run. For now, the installation is performed

manually and requires administrator permissions. There are four executables which are co-located with riverware.exe:

- RwService.exe
- RwSvcCtrl.exe
- RwRemoteMgr.exe
- CombineRdf.pl

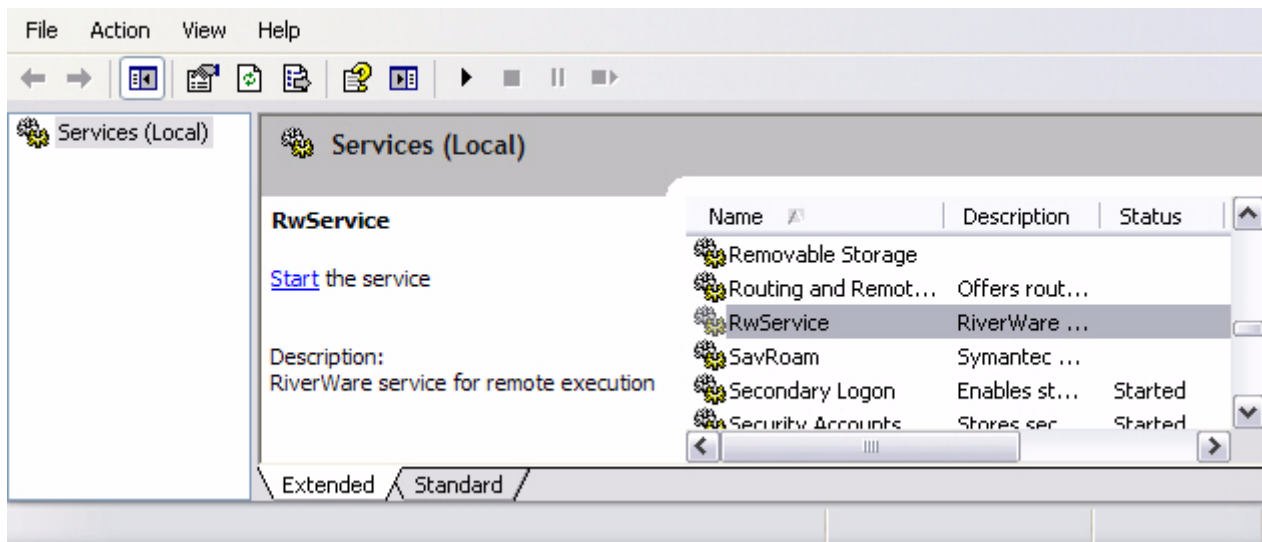
All must be present but for the most part, you will only interact with RwRemoteMgr.exe and RwService.exe.

6.2.1 Installing RwService.exe

The RiverWare service, RwService.exe, must be running on each simulation computer. The RiverWare Service Controller, RwSvcCtrl.exe, is used to install RwService.exe. From a Windows Command Prompt:

```
cd C:\Program Files\CADSWES\RiverWare 6.2
RwSvcCtrl.exe -i "C:\Program Files\CADSWES\RiverWare 6.2\RwService.exe"
```

Once the service has been installed it can be started using the Windows Service dialog: TODO HOW DO YOU GET THIS?



Although not shown, the service can be stopped, paused and restarted using the Service dialog (although it must be started and not paused for the distributed simulations to run).

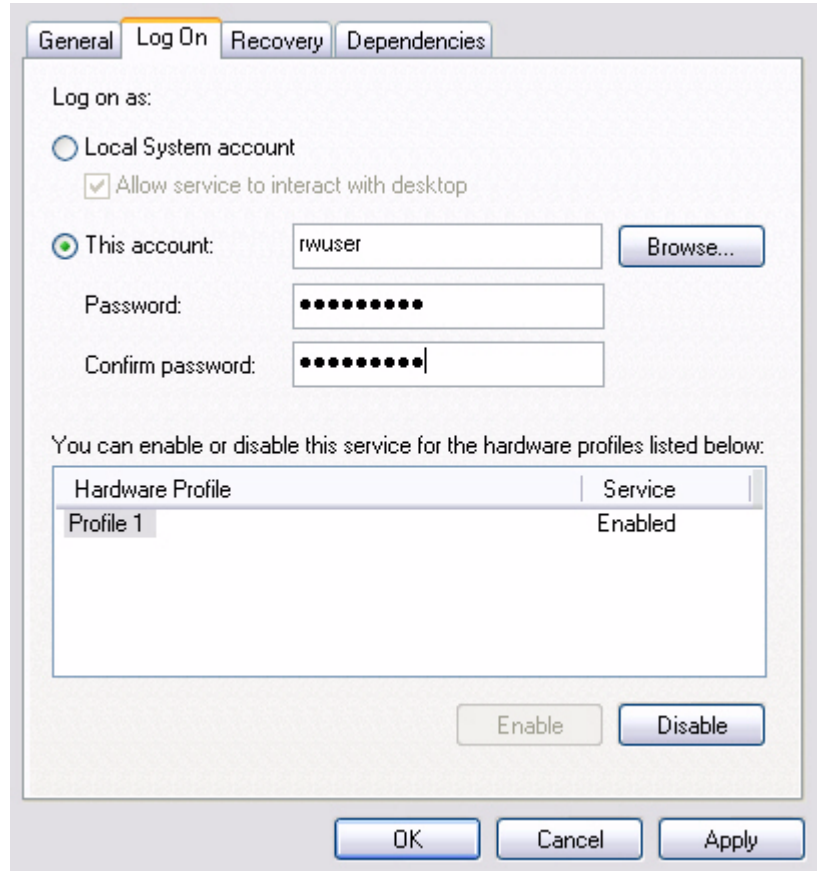
RwSvcCtrl.exe is also used to uninstall the service:

```
cd C:\Program Files\CADSWES\RiverWare 5.2
RwSvcCtrl.exe -u "C:\Program Files\CADSWES\RiverWare 5.2\RwService.exe"
```

6.2.2 Setting The Service Log On

By default a program started by a service has very limited permissions. RiverWare is started by the RiverWare service on the simulation computers, so by default it has very limited permissions. For example, it might not be able to access network directories or write in certain directories. To alleviate this, it might be necessary for the RiverWare service to run as a specific user. To set this up, right-mouse-click on the “RwService” line in the Service dialog, select “Properties” and select the “Log On” tab. Here you can specify the user the RiverWare service will run as:

Although not shown, in the Properties dialog, in the “General” tab, it’s possible to specify whether RwService is started manually (through this dialog, the default) or automatically (when the computer starts up).



6.3 How to Make a Distributed Run

You make a Distributed MRM run by starting execution from the controller computer. There are two options, creating/changing the configuration (within RiverWare) or re-running a configuration (can be external to RiverWare). These are described in the next two sections:

6.3.1 Creating or changing a configuration

If you are creating a new configuration or changing an existing configuration, the changes must be made from within RiverWare. This will allow RiverWare to create the necessary configuration files that will be passed to the simulation controllers. When you click start, RiverWare will create the necessary configuration files, start the RiverWare Remote Manager, and then exit. The RiverWare Remote Manager controls the execution of the MRM runs on the simulation computers. Here are the steps to making the runs in this case:

1. Open RiverWare

2. Fully define the configuration or make any changes to an existing configuration in the MRM Run Control Configuration dialog. Click [HERE \(Section 6.1\)](#) for the options.
3. Apply the changes.
4. **SAVE THE MODEL.** The distributed runs open and run the model that is saved on the file system. Therefore, you should save the model now, so that the configuration is preserved.
5. Click Start on the Multiple Run Control Dialog. RiverWare will start the Remote Manager and then start the shutdown sequence. It will prompt you for confirmation so you can cancel at any time.
6. From the Remote Manager [HERE \(Section 6.1.2\)](#), click the start button to start the distributed runs.
7. The individual runs start and the status is shown including an estimate of the time remaining.
8. When all runs are complete, the output RDF files are combined into one final RDF file.

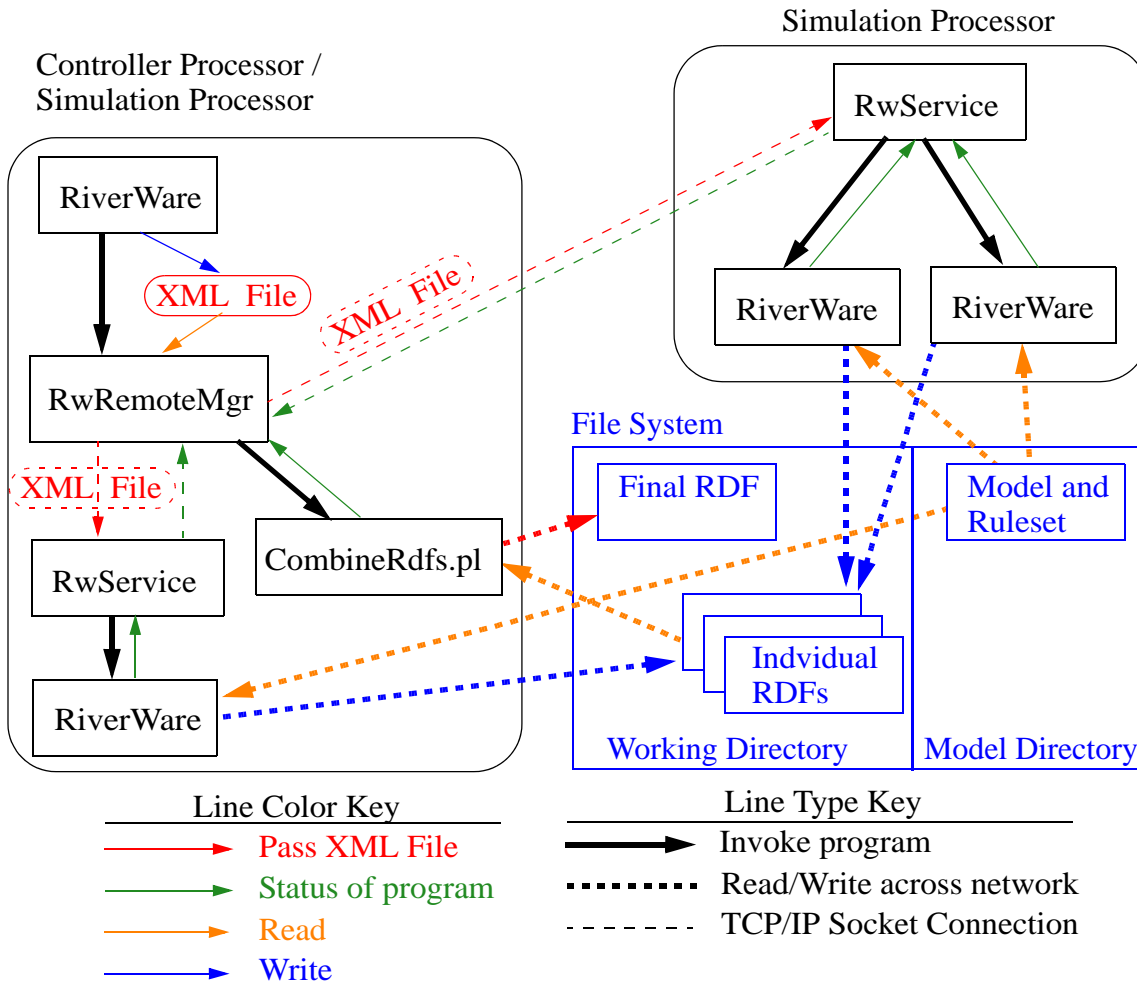
6.3.2 Re-running a configuration

If you are repeating a previously saved configuration, then you can execute the RiverWare Remote Manager directly and not open the RiverWare. In that case, you start at step 6. above.

6.4 How it Works

In the distributed architecture there is a “controller” processor and one or more “simulation” processors.

The distributed architecture introduces three new processes - the RiverWare Remote Manager (RwRemoteMgr), RiverWare Service (RwService), and the RiverWare Service Controller (RwSvcCtrl, not shown).



6.4.1 RiverWare Service

The RiverWare Service runs on each simulation computer; it's a Windows Service which runs in the background, listening to the network for incoming connections (over a TCP/IP socket). Each connection represents a simulation; the Service reads the simulation's XML configuration from the socket, creates a batch script file, and invokes RiverWare in batch mode. The Service reads RiverWare's output and writes it to the socket.

The RiverWare Service is generic - it reads an XML configuration, creates a batch script file and invokes RiverWare in batch mode. The XML configuration can define any batch mode invocation.

6.4.2 RiverWare Service Controller

The RiverWare Service Controller controls the RiverWare Service - it allows a user to install, start, stop, pause, resume, and uninstall the Service. Once installed the Service is a “real” Windows Service and can also be controlled through the Windows Service interface. The Service must be installed and started (and not paused) to receive incoming network connections.

6.4.3 RiverWare Remote Manager

The RiverWare Remote Manager runs on the controller computer. It parses an XML configuration file which defines the simulations and:

- Creates an XML configuration for each of the simulations.
- Configures its user interface (a dialog which shows the status of the simulations).
- For each simulation, connects to the simulation computer (over a TCP/IP socket), writes the XML configuration to the socket, and reads RiverWare’s output from the socket (which it uses to update its status dialog).
- When all simulations have finished, combines the partial RDF files to create the final RDF files.

6.4.4 XML Configurations

Previous sections have referred to XML configurations. These are the files that RiverWare creates to control/define the distributed MRM runs.

Note: This section is a technical reference providing more information on how this utility works. The XML files are generated by the utility and the user does not need to edit these files to make a distributed MRM run.

There are three distinct XML configurations which exist as top-level elements in an XML document; the configurations are identified by their names and can coexist within a document:

- “distrib” identifies a distributed concurrent run (hereafter referred to as the “distributed configuration”).
- “RW” identifies a RiverWare batch mode invocation (hereafter referred to as the “RiverWare configuration”).
- “GenApp” identifies a generic application invocation.

For example, an XML document which defined a distributed concurrent run, two RiverWare batch mode invocations and a generic application invocation would have top-level elements:

```
<document>
  <distrib>
  ...
</distrib>
  <RW>
  ...
```

```

</RW>
<RW>
...
</RW>
<GenApp>
...
</GenApp>
</document>

```

The following sections describe the Distributed Configuration, RiverWare Configuration and GenApp.

Distributed Configuration: The distributed configuration document is created by RiverWare when a user starts a distributed concurrent run. DistribMrmCtrl parses the distributed configuration and creates the RiverWare configurations for the simulations. Some elements are from the MRM configuration, the others are provided by RiverWare. A “Sample” of this XML file is shown below with key elements preceded by brief descriptions:

<distrib>

If required, the login information from the MRM configuration. Missing user or password is prompted for.

```
<login user="" passwd=""/>
```

The RiverWare executable which started the distributed concurrent run. The assumption is that all simulation computers have this executable installed.

```
<app>C:\Program Files\CADSWES\RiverWare 5.2\riverware.exe</app>
```

The model loaded when the user starts the distributed concurrent run. The assumption is that all simulation computers can access the model using the same path.

```
<model>R:\CRSS\model\CRSS.mdl</model>
```

The config attribute is the MRM configuration selected when the user starts the distributed concurrent run. The mrm elements are from the MRM configuration and identify the individual simulations - the simulation computer and the traces to simulate. Although the mrm element supports per-simulation-computer port numbers, the assumption is that all simulation computers use the same port number (and the MRM configuration dialog allows a single port number to be entered).

```
<mrmlist config="Powell Mead 2007 ROD Operations">
```

```
<mrm addr="lc1.usbr.gov" port="27285" firstTrace="1" numTrace="200"/>
```

```
<mrm addr="lc2.usbr.gov" port="27285" firstTrace="201" numTrace="300"/>
```

```
</mrmlist>
```

The rdflist element is a list of the final RDF files, while the slotlist element is a list of the slots which are written to the RDF files. Slots can be written to multiple RDF files; they're associated with the RDF files by the idxlist attribute, whose value is a comma-separated list of RDF file indices. RiverWare initializes the RDF DMI and mines its data structures to generate rdflist and slotlist.

```
<rdflist num="2">
```

```
<rdf name="R:\CRSS\results\Res.rdf" idx="0"/>
```

```

<rdf name="R:\\CRSS\\results\\Salt.rdf" idx="1"/>
</rdflist>
<slotlist>
  <slot name="Powell.Outflow" idxlist="0,1"/>
  <slot name="Powell.Storage" idxlist="0"/>
</slotlist>

```

The *envlist* element specifies RiverWare's runtime environment; *RIVERWARE_HOME* is from the version of RiverWare which starts the distributed concurrent run, all others are from the MRM configuration.

```

<envlist>
  <env>RIVERWARE_HOME_516=C:\\Program Files\\CADSWES\\RiverWare 5.1.6 Patch</env>
  <env>CRSS_DIR=R:\\CRSS</env>
</envlist>

```

The *tempdir* element is from the MRM configuration and is the intermediate directory where the individual simulations write the partial RDF files.

```

<tempdir>R:\\CRSS\\temp</tempdir>
</distrib>

```

RiverWare Configuration: The RiverWare configuration defines a RiverWare batch mode invocation. In a distributed concurrent run, *DistribMrmCtrl* creates the RiverWare configuration; in other contexts a user could create the RiverWare configuration. This example shows a RiverWare configuration from a distributed configuration; not all elements are valid in other contexts. Some elements are from the MRM configuration, the others are provided by *DistribMrmCtrl*. Key elements, preceded by brief descriptions, are:

```
<RW>
```

The simulation computer, from the distributed configuration's *mrm* element.

```
<host addr="lc1.usbr.gov" port="27285"/>
```

RiverWare's executable, from the distributed configuration's *app* element.

```
<app>C:\\Program Files\\CADSWES\\RiverWare 5.1.6 Patch\\riverware.exe</app>
```

Creates the batch script file; the name attribute is provided by *DistribMrmCtrl*, with each individual simulation having a unique name.

```
<script name="R:\\CRSS\\temp\\script0.rcl">
```

Adds the *OpenWorkspace* command to the batch script file.

```
<openws>R:\\CRSS\\model\\CRSS.mdl</openws>
```

Adds the *StartController* command to the batch script file; *firstTrace*, *numTrace* and *ctlFile* are *StartController* !MRM options. [HERE \(BatchMode.pdf, Section 4.3\)](#). The *config* attribute and *firstTrace* and *numTrace* elements are from the distributed configuration. The *ctlFile* element's value is provided by *DistribMrmCtrl*, with each individual simulation having a unique name. *DistribMrmCtrl* creates the control file from the distributed configuration's *rdflist* and *slotlist* elements.

```

<start>
  <mrm config="Powell Mead 2007 ROD Operations">
    <firstTrace>1</firstTrace>
    <numTrace>200</numTrace>
    <ctlFile>R:\\CRSS\\temp\\control0.ctl"</ctlFile>
  </mrm>
</start>

```

Adds the `CloseWorkspace` command to the batch script file.

```

<close/>
</script>

```

Specifies RiverWare's output file; the value is provided by `DistribMrmCtrl`, with each individual simulation having a unique name. The value is used with the new "--remout" option, described below.

```

<output>R:\\CRSS\\temp\\output0.log</output>

```

RiverWare's runtime environment, from the distributed configuration.

```

<envlist>
  <env>RIVERWARE_HOME_516=C:\\Program Files\\CADSWES\\RiverWare 5.1.6 Patch</env>
  <env>CRSS_DIR=R:\\CRSS</env>
</envlist>
</RW>

```

GenApp Configuration: The GenApp defines a generic application. In the standard case, it will executed the perl script that combines the individual RDF files into one RDF file.

```

<GenApp>
  <app>C:\\Perl\\bin\\perl.exe</app>
  <arglist>
    <arg>R:\\CRSS\\bin\\CombineRdf.pl</arg>
    <arg>-o</arg>
    <arg>R:\\CRSS\\results\\Res.rdf</arg>
    <arg>R:\\CRSS\\temp\\Res.*.rdf</arg>
  </arglist>
  <output>%scratchfile%</output>
  <envlist>
    <env>CRSS_DIR=R:\\CRSS</env>
  </envlist>
</GenApp>

```

Configuration Output Options: Both the RW configuration and the GenApp configuration specify “output” files - for RW configurations, it’s the script file, the control file and the output file; for GenApp configurations, it’s the output file. If the controller computer and the simulation computers have access to the same network directories then the files can be created in a network directory by naming them, e.g.:

```
<output>R:\\CRSS\\temp\\output0.log</output>
```

If they don’t have access to the same network directories then the files must be created on the simulation computer, and it’s possible the controller computer (where the files are specified) won’t know where to create the files on the simulation computer. To accommodate this, “output” files can have the following special values:

- `%tempfile%` - The file is a temporary file which persists.
- `%scratchfile%` - The file is a temporary file which is removed.
- `%devnull%` - No file is created (the equivalent of redirecting a process’s output to `/dev/null`).

A temporary file is a guaranteed unique file in a temporary directory. (The temporary directory might vary, but on Windows it’s commonly `C:\WINDOWS\Temp`.)

6.4.5 Combining RDF Files With `CombineRdf.pl`

Each simulation writes an intermediate RDF file which is a portion of the final RDF file; when all simulations have finished the intermediate files are combined into the final file. A Perl script, `CombineRdf.pl`, combines the intermediate RDF files. Its syntax is:

```
CombineRdf.pl <number of simulations> <number of traces> <intermediate RDF files> <final RDF file>
```

`DistribMrmCtrl` invokes the Perl script, so a user needn’t know the specifics of it.

Note that the choice of Perl requires that Perl be installed on the controller computer. If this is not desirable, the program could easily be re-implemented in C++.