RiverWare Demonstration Tutorial

Center for Advance Water and Environ

Center for Advanced Decision Support for Water and Environmental Systems (CADSWES)

This RiverWare informational tutorial introduces you to the capabilities of RiverWare as an advanced water resource modeling tool and consists of three sections, **Simulation in River-Ware**, **Rulebased Simulation**, and **Multiple Run Management**. These tutorials are independent documents but later topics do build on skills you gain from earlier exercises, so if you have not gone through the earlier topics or have not used RiverWare recently it is recommended that you start with the tutorial on **Simulation in RiverWare**.

Before you start...You must have RiverWare 9.0.4 or later installed on your computer and have a valid license file. For more information about downloading RiverWare, visit RiverWare.org. To acquire a demonstration license, please email Installation Support (installsupport@colo-rado.edu).

You must have downloaded the files associated with this tutorial and moved them to a reasonable folder like **C: /temp**. The following files are used in this tutorial:

- MuddyBartlettBasin_RBS_Start.mdl.gz Starting MODEL
- MuddyBartlett_Info_RBSstart.rls.gz Starting RULESET
- MuddyBartlett_Info_RBSfinal.rls.gz (this is just the final resulting ruleset of this module)

2 Rulebased Simulation

This informational tutorial introduces you to Rulebased Simulation in RiverWare. You will be introduced to:

- Running a Rulebased Simulation
- Examining functions and rules
- Writing a rule using the RPL Editor and Palette
- Obtaining more information on a run using the diagnostics and debugger.

To get started let's open the model in RiverWare:



Double-click the RiverWare icon on the desktop.

Close the Quick Start window as we'll show you how to open a model from the workspace.

Select **File** and then **Open Model**... from the main workspace menu bar or select the Open Model File toolbar button.



Open the file MuddyBartl ettBasi n_RBS_Start. mdl.gz

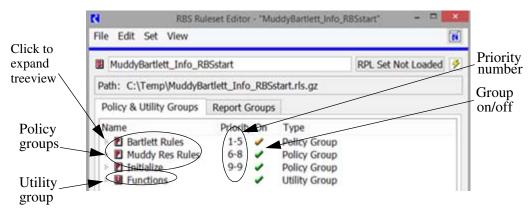
2.1 Running a Rulebased Simulation

Rulebased simulation is an extension of basic Simulation in which some of the data to solve an under-determined model is provided by a set of user-defined rules. Rule execution alternates with dispatching to simulate the effects of the rules in the model.

The policy for the basin is in the Ruleset.

In the **Policy** menu of the main workspace, select **Ruleset** and then **Open...**

□ In the file chooser, select the following ruleset: MuddyBartlett_Info_RBSstart.rls.gz



The Ruleset Editor initially displays three **Policy Groups** and one **Utility Group**. A Policy Group is a folder for holding rules; Utility Groups contain functions.

Select the triangle/arrow to the left of each of the groups to expand the treeview.

Now you can see all individual rules within each Policy Group. Note that the check mark next to Bartlett Rules is orange because the priority 1 rule, **Bartlett Flood Control**, is turned off.

For this simple model, there are nine rules (indicated by a red \mathbf{R} symbol) and eight functions (indicated by the \mathbf{F} symbol).

Before running, you must tell RiverWare that this is the ruleset you wish to use, i.e. Loading the ruleset. Loading a ruleset validates it for unspecified expressions, illegal object and slot names, conflicting expression types, and syntax errors.

Load your ruleset by clicking on the **RPL Set Not Loaded** button. This validates the ruleset and loads it into the model.

Run the simulation by selecting 🧏 in the main workspace.

Notice that the controller is set to Rulebased Simulation

Click Start

When the run is finished, close the **Run Status** and **Run Control** dialogs.

On each timestep, each rule is executed, then the objects solve as necessary to simulate the effects of rules setting values. Rule #9, the lowest priority, is executed first, while rule #1, the highest priority, fires last. Now let's briefly examine the results of the run and see how the rules set values in the model:

Open the **Model Run Analysis** dialog by clicking *(D)* in the main workspace.

For a more in depth explanation of the Model Run Analysis dialog, see pages 14-16 of the Introductory Information Tutorial.

Sort: Custom	 Sor 	< > O						Details					
	06-04-1996	06-05-1996	06-06-1996	06-07-1996	06-08-1996	06-09-1996	06-10-1996	06-11-1996	06-12-1996	06-13-1996	06-14-1996	06-15-1996	06-16-1996
Muddy Reservoir	<u> </u>	<u>†0</u> ↓8R	<u>+</u> 70 ↓8R	<u>+</u> 0 ↓8R	<u>+</u> 0 ↓8R	<u>+</u> + + 8R	<u>†0</u> ⇒8R	<u>+</u> 70 ↓8R	<u>+</u> 0 ↓8R	<u>+</u> + 8R	<u>+</u> 0 ↓8R	<u>+</u> 70 ↓8R	<u>†(</u> ↓8
Gage Above Bartlett													
Bartlett Reservoir	<u>178</u> ↓2R	<u>†8</u> ↓2R	<u> </u>	<u>†8</u> ↓2R	<u> †8</u> ↓2R	<u>†8</u> ↓2R	<u>†8</u> ⇒2R	<u>†8</u> ⇒2R	<u>18</u> ↓2R	<u>†8</u> ↓2R	<u>+</u> 78 ↓2R	<u>178</u> ↓2R	<u>†</u> ₹ ↓2
IC Delivery													
Bartlett to Border	Ŷ2	Ŷ2	Ŷ2	Ŷ2	Ŷ2	Ŷ2	Ŷ2	Ŷ2	Ŷ2	Ŷ2	Ŷ2	Ŷ2	Ŷź
> Yahoo Canyon	Ϋ́2R	Ŷ2R	Ŷ2R	Ŷ2R	Ŷ2R	Ŷ2R	τ̂2R	Ϋ́2R	Ŷ2R	Ϋ́2R	τ̂2R	Ŷ2R	Ŷ2

Scroll through the timesteps horizontally and notice the reservoirs and reaches were dispatched in accordance with various rules in the model ruleset.

Find the dispatch cell for **Bartlett Reservoir** on June 10 (06-10-1996) and select it.

~ Scr	oll: Ja	nuary 1	, 1996		E
06-08-1996	06-09-1996	06-10-1996	06-11-1996	06-12-1996	2001 61 20
<u> </u>	<u> </u>	<u>†8</u> ↓2R	<u> </u>	<u>†8</u> ↓2R	<u>†</u>
	06-08-1996	日本 109-09-1996 109-09-1996 109-09-1996 109-09-1996	日本 100-1996 100-1996 100-1996 100-1996 100-1996	48 84 84 84 84 84	84 84 84 84 84 84 84 84 84 84 84 84 84 84 85 66-10-1996 84 84 85 66-11-1996 84 84

Text at the bottom of the window gives details about dispatch method and the rules that set the values on the dispatch slots

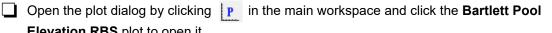


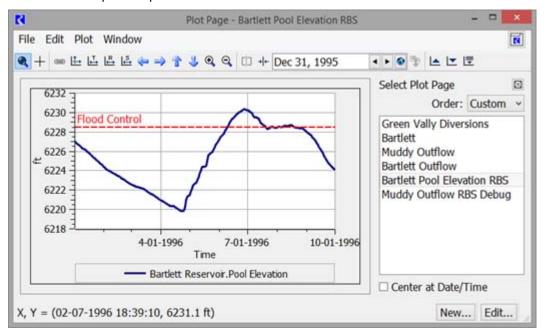
The Inflow was set by resulting output of rule 8- **Green Valley Diversion** requirements, the Outflow by rule 2- **IC Delivery** requirements. You can select **View** and then **Grid Cell Legend** for more information.

In this way, you can determine the policy in effect for each timestep and each object in the run.

Close the Model Run Analysis dialog.

Elevation RBS plot to open it.





If you remember, we saw that the Flood Control rule is turned off. Therefore there is no logic or policy in effect to keep the Bartlett Pool Elevation from exceeding maximum prescribed elevation. We see this illustrated in the plot in July 1996. In the next section, we'll turn on the rule and see its effect.

Close the plot dialog.

2.2 Examining functions and a rule

The Ruleset contains the policy of your basin. You write the rules and then give each one a priority starting with 1 up through the number of rules you have.

Rules are set up using logic in the RiverWare Policy Language (RPL). You can make use of predefined as well as user-defined functions to build up your logic and make it understandable by others. Let's start by walking through the logic of a rule:

Open the **Bartlett Flood Control** rule by double clicking on its name in the **Ruleset Editor**. Set Bartlett Reservoir.Outflow



The rule contains logic (IF, THEN), slot names (e.g. Bartlett.Pool Elevation), and functions, both user defined and predefined functions. In general, it says:

Object.Slot []= IF (A) THEN

Minimum (B, C)

END IF

This rule is setting the value of the **Barlett Reservoir Outflow** slot at the current timestep. This rule is executed at each timestep, so the logic is written with that assumption.

We'll go through each item a, b, and c to describe the logic:

a. If it is BartlettFloodSeason() AND the current Bartlett.Pool Elevation is above the Bartlett Data.Flood Control Elevation,

This expression is used to check the state of the system. We only want this policy in effect during specific conditions. Thus, we are defining the logic for which we want this policy. The first item BartlettFloodSeason is a User-defined function, which allows you to encapsulate an expression and give it a meaningful name. You can then call this function from any other rule or function. This allows you to simplify and modularize your policy.

Open the **BartlettFloodSeason** function by double clicking on the function name in the rule.



This is a function created to define the time of year when there could potentially be flooding. It evaluates to of either True or False. The statement asks whether the current time (@ "t") is

at least January 1st (>= @ "January 1") AND

• no later than June 30th ($\leq a$ "June 30").

Close the **Function Editor** window.

Now let's look at the two values to which the rule could evaluate: b or c. The rule computes the minimum of these two.

b. the Outflow to lower the Pool Elevation to the Flood Control Elevation

The **SolveOutflow** function (in green) is a predefined RiverWare function that performs a mass balance to compute a reservoir's outflow given its inflow, its previous storage, and the storage specified at the current timestep (in this case determined by the **ElevationToStorage** function).

For further information about this or any predefined function, see the RiverWare Help docu-

mentation, which can be accessed by clicking ? and choosing **RPL** and then **RPL**

Predefined Functions from the bookmarks menu.

Let's look at the last possibility:

c. the maximum outflow allowed by the physical limitations of the reservoir's structures.

Open the **MaxOut** function.

S F M	axOut		RPL Se	et Loaded	4
Arguments:	OBJECT res	2	Return Type:	NUMERIC	~
GetMaxC	OutflowGivenInflow (res , res .	"Inflow" []	, @"t")		^
					\checkmark

This function computes a number (it's return type is NUMERIC) by evaluating a predefined RiverWare function, **GetMaxOutflowGivenInflow**, at the current timestep when a reservoir, res is passed in as an argument.

The GetMaxOutflowGivenInflow is another predefined function that computes the maximum outflow from a reservoir considering the previous timestep storage, inflow, all sinks, side flows, and sources, and the physical characteristics of the outlet structures. In the case of a storage reservoir (like Muddy), the maximum release is interpolated from the Max Release table based on its elevation.

Close the Function Editor window and close the Bartlett Flood Control rule

So now that we've looked at the details, let's go back to the overall rule. Basically, it assigns the Barlett Reservoir Outflow slot to the value that is computed on the right side of the assignment (i.e. the equal sign). During flood season and when the reservoir is high, this rule computes and sets the Outflow to ensure that the level of water in the reservoir is reduced to be able to accommodate possible flooding while not exceeding the flow that the turbines and spillways can handle.

Now we will see how the rule affects reservoir operations:

Turn on the Bartlett Flood Control rule by clicking the red X in the Ruleset Editor.

The X turns into a green check mark and the orange check mark next to Bartlett Rules turns green.

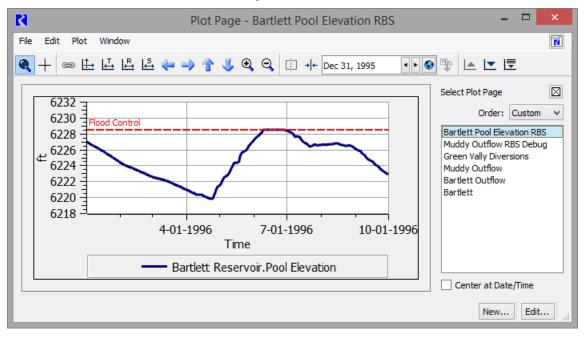
Run the simulation again (click \swarrow and then **Start**). Close the dialogs after the run finishes.

Open the Model Run Analysis (click (D)) and scroll to the June 10th entry for Bartlett Reservoir.



As you can see, the outflow is now set by rule 1 (1R) where before it was set by rule 2. In fact, all of the Bartlett Reservoir outflows through June 27th are set by rule 1.

Now open the Pool Elevation plot you looked at before by clicking **P** and choosing **Bartlett Pool Elevation RBS** from the list on the right.



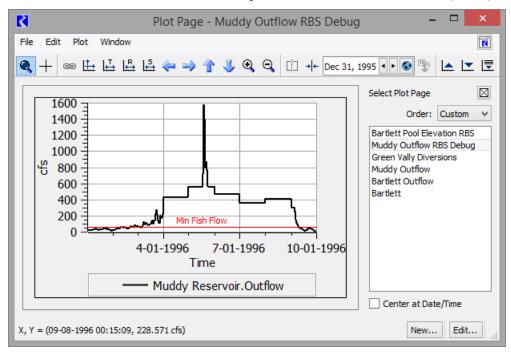
Now that the **BartlettFloodControl** rule is activated, Pool Elevation does not exceed the Flood Control height at any time. Because the rule has the highest priority, no other operation request (i.e. rule) can override it.

2.3 Building a rule

The current ruleset is overlooking a basin requirement to keep the downstream fish population healthy. This operation is especially important during the winter months when natural stream-flow is generally low, as you can see in the following plot:

Open the plot of Muddy Reservoir.Outflow by clicking on the plot icon relation and clicking on the Muddy Outflow RBS Debug plot.

The red line shows the minimum flow required for downstream fish health (55 cfs).



Zoom in on the beginning of the simulation (January to April) by clicking and dragging a rectangle over the far left portion of the graph (resize the window if necessary).

Notice the Outflow is below the minimum.

Close the plot.

You will now build a simple rule which sets Muddy Reservoir's Outflow to meet this downstream environmental flow requirement of 55cfs. Add a new rule to the Muddy Reservoir Policy Group by highlighting **Muddy Res Rules** and right clicking. In the menu, choose **Add** and then **Add Rule**.

e Edit Set View					R		
MuddyBartlett_Info_RBSsta	rt			RPL Set Loaded	9		
Path: R:\doc\Demonstration\In	formationalTut	orial\8.3.2\	Inform	ationalTutorialRBS\Muddy	yBartl		
Policy & Utility Groups Rep	ort Groups						
Name		Priority	On	Туре			
Bartlett Rules			V	Policy Group			
Bartlett Flood Cor	ntrol	1	 Image: A second s	Rule			
R IC Delivery		2	 Image: A set of the set of the	Rule			
Supplement Raftir	ng	3	V	Rule			
R Supplement Diver	ters	4	~	Rule			
Pass Muddy Outfl	ow for Diverter	s 5	V	Rule			
Muddy Res Rules			1	Policy Group			
R Muddy Flood	Open Editor			le			
Pass Natural F	Close Editor			le			
Green Valley [le			
Initialize	Rename			licy Group			
Functions	Change Tex	t Color		lity Group	lity Group		
F MaxOut	Toggle Activ	/e	nction	nction			
F MuddyFloodS	Select Expo		Group	nction			
F DiversionShor				nction			
ExcessToMax	Select Expo	rt For All in	Set	nction			
ExcessStorag	Delete			nction			
F IsRaftingWee	Сору			nction			
F RaftingShorta				nction			
F BartlettFloodS	Append			nction			
	Duplicate						
	Find						
		Doplace					
	Search and	Replace					
	Add			Add Policy Gro	up		
	Add To Rep	ort Group		Add Utility Gro	up		
				Add Rule			
				Add Function	•		

A new entry called "Rule" appears in the Policy Group as priority 9.

Open **Rule** by double clicking on it and change the name from "Rul e" to "Mi n Fi sh Fl ow" by entering it into the field at the top of the **Rule Editor** and pressing Enter.

Rules are constructed by adding statements at the top level and then filling in the expressions in the statements. Statements can be added from the **Rule** menu. The most common statement is the **Assignment** which is used to set a slot value.

Add an assignment statement to the empty rule by selecting **Statement** and then **Add Assignment**.

Now we must now specify which slot to set (the left side) and logic to determine the value to set on the right side. We will fill in both of the unspecified expressions. Unspecified expressions are portions of the rule which have not yet been defined, for example: **<numeric expr>**. The two **<numeric expr>** expressions which appear in your rule indicate that both sides must eventually evaluate to a numeric value. Building rules involves coding the unspecified expressions one by one into functional expressions.

You specify the expressions using the **Palette**. Depending on the type of the expression highlighted, the Palette only enables the buttons for expressions of that type. In this way, the Palette provides syntax support and helps guide you in building rules.

3	RPL F	Palette	- 🗆 🗙					
File Function			R					
Buttons User-E	efined Functions	Predefined Functions Clipboa						
Mathematical / Lo	gical	Conditional / Iterative						
E + E	E-E	IF	IF ELSE					
N * N	N/N	ELSE	ELSE IF					
N ^ N	IsNaN N	FOR	WHILE					
B AND B	B OR B	WITH	SUM					
E > E	E >= E	AVE						
E < E	E <= E	Lists						
E == E	E != E	{E}	E,E					
Objects / Slots		L-L	L ^ L					
Slot [E]	Slot [E,E]	L UNION L	INTERSECTION					
Slot []	NaNToZero N	L <n></n>	INSERT					
Obj . Slot	Object Selector	LENGTH L	APPEND					
Obj ^ Acct . Slot	Account Selector	E IN L	D TO D					
Obj ^ Acct & Slot	Slot Selector	REMOVE	SUB					
Unary		FIND	MAP LIST					
- N	NOT B	Miscellaneous						
Values		E CONCAT E	(E)					
@"t		STRINGIFY E	STOP_RUN E					
Drif	t 🔤	WARNING	(E,E) 🔻					
		Add Comment	Delete Comment					

Open the **Palette** by selecting **Rule** and then **Palette...**. Or right select the expression and select **Palette**.

When the Palette window is active, you can hover over any button on the Palette to get a "tool tip" describing that button.

Highlight the left **<numeric expr>**. This will be the slot to which the rule will assign a value.

	In the Objects/Slots section of the Palette , select the Slot[] button. This replaces the unspecified numeric expression with a series slot expression.
	The square braces following a slot expression refer to the timestep or the row and column of the slot. The three possibilities are:
	1. Slot [E] A series slot at a given row (often specified by a date).
	2. Slot [] A series slot at the current controller timestep or a scalar slot.
	3. Slot [E, E] A table slot at a specified row and column.
	Mouse over each of these three buttons to see their description
	Now highlight just the unspecified portion of the series slot expression, <expr></expr> , and select the Slot Selector button on the Palette.
	In the slot selector, select the Object Type : Reservoir , Object : Muddy Reservoir , Slot : Outflow , then click OK .
	You will need to scroll down to locate Outflow.
R	Select Single Slot for RPL Expression
Create Selection	O Use Slot Set

6 Object Types: 1 (of 9) Objects: 1 (of 2) \triangleleft Slots: 1 (of 23) ₽₽, Ľ₽, All Invert Show Columns: AggSeries All Invert Obj Type Object Slot Cc ^ Type Object Confluence Μ 🖄 Muddy Reservoir Flow TO Pumped Storage 1 DataObj tlett Rese 网 Δ Inflow Muddy Reservoir Muddy Reservoir 1 StorageR Reach 2 M ∕∆ Muddy Reservoir Inflow Sum 1 Reservoir ⊞ werRes Muddy Reservoir Max Iterations 1 LevelPowerReservoir Ħ ddy Decemain May Doloa StorageReservoir M Muddy Reservoir Outflow 1 StreamGage Muddy Reservoir Pool Elevation M Â WaterUser 1 兩 ∕∆ Muddy Reservoir Release 1 < > Ok Cancel Apply

You have finished the left-hand side (LHS) of the rule. When the rule is executed, the rulewill attempt to set a value on the Muddy Reservoir Outflow slot to the value returned from the right-hand side (RHS) of the assignment.

Min Fish Flow

The right-hand side of the rule should check the existing flow and increase it if necessary:

Highlight the RHS **<numeric expr>**.

From the Palette choose the IF option under the Conditional/Iterative section.
Highlight <boolean expr=""></boolean> to specify the type of condition, then select E < E to signify that you want to determine if the value of a slot is less than the value of another.
Muddy Reservoir.Outflow [] = IF (<expr> < <expr>) THEN <numeric expr=""> END IF</numeric></expr></expr>
Highlight the left <expr></expr> and designate it as a slot at the current timestep, Slot [] , then highlight just <expr></expr> again and select the Slot Selector button in the Objects/Slots section of the Palette.
In the slot selector, select the Object Type: Reservoir, Object: Muddy Reservoir, Slot: Outflow, then click OK.
Muddy Reservoir.Outflow [] = IF (Muddy Reservoir.Outflow [] < <numeric expr="">) THEN <numeric expr=""> END IF</numeric></numeric>
Highlight <numeric expr=""> and designate it as a slot at the current timestep, Slot [], then highlight just <expr> again and select the Slot Selector button in the Objects/Slots section of the Palette.</expr></numeric>
In the Slot Selector, select the Object Type: DataObj, Object: Muddy Data, Slot: Min Fish Flow, then click OK.
= IF (Muddy Reservoir.Outflow [] < Muddy Data.Min Fish Flow []) THEN <numeric expr=""> END IF</numeric>
The conditional statement is complete. Now we will specify the value of the Outflow slot using a predefined RiverWare function.
Highlight <numeric expr=""></numeric> and navigate to the Predefined Functions tab of the Palette.
At the bottom of the Palette window, check the box next to "Show only functions with a return type matching the selected expression"
Scroll through the alphabetical list to the Min function. Double select it to insert it into the rule.
The rule now reflects that we want the rule to compare two values and choose the smallest.
The purpose of the rule is to set outflow to a level that protects the fish downstream, but that should not conflict with the maximum outflow possible for the reservoir, so the smaller of the two values will be used.
Highlight the first <numeric expr=""> and make it a slot at the current timestep: Slot [], then highlight <expr> and in the Slot Selector specify Object Type: DataObj, Object: Muddy Data, Slot: Min Fish Flow, then click OK.</expr></numeric>

Highlight the remaining **<numeric expr>** and navigate to the User-Defined Functions tab of the Palette. Double select MaxOut.

K	RPL Palette – 🗖 🗙								
File Function			R						
Buttons Use	er-Defined Functions	Predefined Functions	Clipboard						
Set Name: Mu	uddyBartlett_Info_RBSs	start							
ReturnType	Name	Arguments	^						
NUMERIC	F DiversionShortag	e							
NUMERIC	ExcessStorageAs	Flow							
NUMERIC	ExcessToMax								
NUMERIC	F MaxOut	OBJECT res							
NUMERIC	RaftingShortage		~						

Highlight the <object expr> argument to specify which reservoir you want to look at and choose Object Selector (from the Palette Buttons tab). In the selector dialog choose **Object Type: Reservoir**, **Object: Muddy Reservoir**, then click **OK**.

S P 9 Min Fish Flow RPL Set Loade	d 🗳
Muddy Reservoir.Outflow []	~
= IF (Muddy Reservoir.Outflow [] < Muddy Data.Min Fish Flow []) THEN Min (Muddy Data.Min Fish Flow [] , MaxOut (Muddy Reservoir)	
END IF	

Your finished rule should look like the above figure.

Close the Rule Editor window and return to the Ruleset Editor window.

The Min Fish Flow rule you just wrote is currently at priority 9, but it needs to be at priority 7.

Click and drag the rule to right on the **Muddy Flood Control** rule and release the left mouse button. A confirmation window will appear asking if you want to finalize the move from priority 9 to Priority 6 or Priority 7.

Click **To Priority 6**.

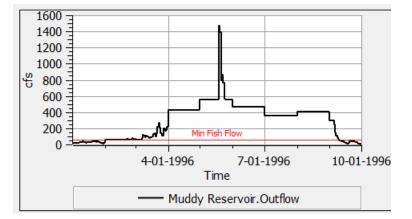
Now Min Fish Flow is priority 6 and the remaining rules have shifted down one priority.

Save the new Ruleset via File and then Save RBS Ruleset

Run the model.

Open the same plot of Muddy Reservoir.Outflow you looked at in section Section 2.3, "Building"

a rule" by clicking on the plot icon **P** and clicking on the **Muddy Outflow RBS Debug** plot.



The red line shows the minimum flow required for downstream fish health (55 cfs).

Zoom in on January and February 1996 (resize the window if necessary).

After zooming in we can see that the violations were fixed starting in February, but not in January. Let's use the Model Run Analysis to see which rules were in effect for this month.

- Open the Model Run Analysis dialog by clicking on the *D* icon, and scroll left to the January 1, 1996 timestep.
- Select the cell for Muddy on January 1.

Muddy Reservoir's Outflow was set by rule 8 for the whole month of January but the **Min Fish Flow** rule that you wrote is at priority 6, a higher priority that should have overwritten rule 8. Obviously there is a breakdown somewhere. In the next section we debug this rule to find the problem.

	1
<u> </u>	
(↓8R `	Ì
\sim	í

2.4 Debugging

After setting up a rulebased simulation model, it is important to verify the results and debug any issues. Some problems within a model or ruleset will cause the run to abort generating an error message that can direct you to the problem. Other times a model may run to completion, but it may not produce the correct results. These problems can be more difficult to debug. This section will present two tools: Diagnostics and the RPL Debugger, which can assist in this debugging process.

- From the workspace, select Utilities and then Diagnostics Output....
- Select click **Edit** and then **Clear Messages** from the menu, to clear the messages from all the previous runs.
- On the Run Control, start a new run.

Since we only want to look at diagnostics for how rules set the Muddy Reservoir Outflow in January, 1996, the diagnostics have been pre-configured using context filters.

File	Edit	Settings	Search:		٠	Filter: (C 38					¥ [Ignore Ca	nse 🗌 Regi	🛛 🗹 Au	to Scroll	1
	Conte	ext					Di	agnostics Mes	sage								
67:	24:00	January 1,	1996; OBJEC	T: Mut	ddy I	Reservoir	Pr	iority 0: Inflov	(Governi	ing)							
68:	24:00	January 1,	1996; OBJEC	T: Mue	ddy I	Reservoir	Pr	iority R 8: Out	flow (Gov	erning)							
69:	24:00	January 1,	1996; OBJEC	T: Mut	ddy i	Reservoir	Re	ady for dispat	ch metho	d ("Solve	given Inflo	w, Out	flow").				
70:	24:00	January 1,	, 1996; OBJEC	T: Mu	ddy I	Reservoir	Ex	ecuting dispat	ch metho	d (Solve g	iven Inflov	v, Outf	ow) at prior	rity 8.			
71:	24:00	January 1,	, 1996; OBJEC	T: Mut	ddy I	Reservoir	SI	ot ("Seepage") is alread	ty in know	n set.						
72:	24:00	January 1,	1996; OBJEC	T: Mut	ddy I	Reservoir	SI	ot ("Storage")	added to	known se	t.						
73:	24:00	January 1,	, 1996; OBJEC	T: Mu	ddy I	Reservoir	SI	ot ("Pool Eleva	tion") add	ded to kno	wn set.						
74:	24:00	January 1,	1996; OBJEC	T: Mu	ddy I	Reservoir	SI	ot ("Spill") is a	Iready in	known set	L						
75:			1996; OBJEC					ot ("Release")									
76:	24:00	January 1	1996; OBJEC	T: Mu	ddy I	Reservoir		ne dispatch me									
77:			, 1996; RULE:				1000	ecuting rule #									-
			, 1996; RULE:					isignment initi									
79: 80:	24:00) January 1,	, 1996; RULE:	: (6) Mi	n Fis	sh Flow		aluation of the Rulebased					tatement te	erminated e	arly for the	following	H
81:							"N	1uddyBartlettB	asin_RBS_	_Start.mdl	.gz at 09:	29:04 9	September 2	29, 2021 (0	seconds)"		
82:																	
<						-	> 0										5

Scroll to lines which show "24:00 January1, 1996 RULE: (6) Min Fish Flow" in the context panel.

These entries show the process RiverWare went through to implement the Min Fish Flow rule. The third diagnostic message for the Min Fish Flow rule specifically illuminates the problem:

36: 24:00 January 1, 1996 RULE: (6) Min Fish Flow: Evaluation of the right-hand side of the Assignment statement terminated early for the following reason: Encountered invalid value in the series slot "Muddy Data.Min Fish Flow" at the date 24:00 January 1, 1996. This occurred at the following location within the expression: \$ "Muddy Data.Min Fish Flow" [].

The first piece of important information that we get from this message is that the rule "terminated early." A rule terminates early when it does not have enough information to evaluate the rule, i.e. it finds a NaN ("Not a Number") in a referenced slot. When it terminates early, it does not set any values. This explains why rule 6 did not set the **Muddy Reservoir Outflow** for January, 1996. The message then tells us why the rule terminated early: **Encountered invalid** value in the series slot "**Muddy Data.Min Fish Flow**" at the date 24:00 January 1, 1996.

The diagnostic message suggests that we need to look at Muddy Data.Min Fish Flow.

Open the Muddy Data object, and open the Min Fish Flow slot.

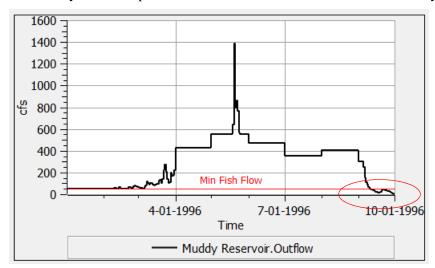
🕅 Min F	ish Flow		ust
Value:			c
Dec 31, 1995		< > §	Alt Units 🔳 🖞
	-		
Input Mode: 🔘 Seri	ies 🖲 Period	ic	
	cfs		Period (Base 1900
12-31-1995 Sun	55.00 I 0		th Interval erpolate 💿 Looku
01-01-1996 Mon	NaN O		•
01-02-1996 Tue	NaN O		cfs
01-03-1996 Wed	NaN O	Jan	Na
01-04-1996 Thu	NaN O	Feb	55.0
01-05-1996 Fri	NaN O	Mar	55.0
01-06-1996 Sat	NaN O	Apr	55.0
01-07-1996 Sun	NaN O		55.0
01-08-1996 Mon	NaN O	May	
01-09-1996 Tue	NaN O	Jun	55.0
01-10-1996 Wed	NaN O	Jul	55.0
01-11-1996 Thu	NaN O	Aug	55.0
01-12-1996 Fri	NaN O	Sep	55.0
01-13-1996 Sat	NaN O	Oct	55.0
01-14-1996 Sun	NaN O		55.0
01-15-1996 Mon	NaN O	Nov	
01-16-1996 Tue	NaN O	Dec	55.0
01 17 1006 Wed			
Show: Descripti	on		

When the model was set up the January cell was left without a value, so for the whole month there was no minimum flow data for RiverWare to meet.

Click in the cell, type "55" and press Enter.

Rerun the model and see that the rule was successfully implemented.

Diagnosti							cs Output Window - RiverWare 8.3.3 - MuddyBartlettBasin_RBS_Start.mdLgz			
File	Edit Settings Set	rch:		A Filte	er: C	ж	✓ Ignore Case □ RagEx	🕑 Auto		
	Context						Diagnostics Message			
38:	38: 24:00 January 1, 1996; OBJECT: Muddy Reservoir						The dispatch method "Solve given Inflow, Outflow" successfully solved.			
	39: 24:00 January 1, 1996; RULE: (6) Min Fish Flow						Executing rule #6 ("Min Fish Flow", within group "Muddy Res Rules")			
	40: 24:00 January 1, 1996; RULE: (6) Min Fish Flow						Assignment initiated (the left-hand side is "\$ "Muddy Reservoir.Outflow" []").			
	41: 24:00 January 1, 1996; RULE: (6) Min Fish Flow					<	Evaluation of Assignment statement successful; will attempt assignment: Muddy Reservoir.Outflow[January 1, 1996]	= 55.00 c		
	2: 24:00 January 1, 1996; SLOT: Muddy Reservoir.Outflow					flow	Set value = 55.00000 * 1.0cfs = 1.557427 * 1.0cms.			
43:	24:00 January 1, 1996; OBJECT: Muddy Reservoir						Slot ("Outflow") is already in known set.			
44:	24:00 January 1, 1996; SLOT: Muddy Reservoir.Outflow					flow	Propagate value to slot ("Confluence.Inflow2").			
45:	5: 24:00 January 1, 1996; OBJECT: Muddy Reservoir						Priority 8: Spill (Non-Governing)			
46:	24:00 January 1, 199	; OBJEC	T: Mud	Idy Reserv	voir		Priority 0: Diversion (Non-Governing)			
47:	7: 24:00 January 1, 1996; OBJECT: Muddy Reservoir						Priority 0: Return Flow (Non-Governing)			
48:	24:00 January 1, 199	; OBJEC	T: Mud	idy Reserv	voir		Priority 8: Seepage (Non-Governing)			
49:	24:00 January 1, 199	; OBJEC	T: Mud	idy Reserv	voir		Re-dispatching using slot priorities.			
50:	24:00 January 1, 199	; OBJEC	T: Mud	idy Reserv	voir		Priority 0: Inflow (Governing)			
51:	24:00 January 1, 199	; OBJEC	T: Mud	idy Reserv	voir		Priority R.6: Outflow (Governing)			
52:	24:00 January 1, 199	; OBJEC	T: Mud	dy Reserv	voir		Ready for dispatch method ("Solve given Juflow, Outflow").			
53-	24:00 January 1, 199	- RUE	(6) Mr	Eich Elos			The rule finished successfully.			



The Muddy Outflows plot shows the min fish flow is now met in January.

Looking at the right side of the Muddy Outflows plot, we see that the Min Fish Flow is not being met at the end of the simulation period either. It appears that there could be another problem with either the policy or the date. We can use the RPL Debugger to find out why the minimum flow is not being met.

In the RiverWare workspace menu bar, select **Policy** and then **RPL Debugger...**

This opens the RPL Debugger dialog. At this point the debugger is not yet enabled.

Rulesets can be very large and complicated, and consequently it can be difficult to determine why a set is doing what it is doing. The RPL debugger is designed to help understand the behavior of rulesets by pausing execution, looking at the values of RPL expressions as they are evaluated, and stepping through RPL execution.

In the RPL Debugger dialog, select **Debug** and then **Enable RPL Debugging**.

We will set a breakpoint in the Min Fish Flow rule, but we only want to debug on a single timestep. If we set the breakpoint now, the debugger will stop the run once every timestep when it gets to the rule. Instead we will set the **Run Controller** to pause the run before the timestep we want. Then we can set our breakpoint.

☐ In **Run Control** dialog (click to open), check the box for **Pause Before Timestep** at the bottom of the window.

File View	Rún	Control	×			
Controller	Cinculation					
Rulebased	Simulation		~			
-Run Parame	eters					
Initial:			• •			
	January 1, 1996					
Finish: Timesteps:						
			Day 🗸			
	nize Slots with Ru	e Run Parameters	_			
	ppiy or cancer the	Run Farameter eute	2			
🔆 Enabled]					
Start	Init	Pause	Stop			
✓ Pause Befo	ore Timestep:	September 10, 199		>		
Run Status						
			100%			
Execution St	tate: Finished					
Current Tim	estep: Septembe	er 30, 1996				
lick Start	to run the r	model				
			before S	entember 10th		
			, before S	eptember 10th	l.	
'he run pa	uses at 93%	% completion		_		hen
he run pa	uses at 93% Min Fish Fl	% completion	ie main w	orkspace, sele	ect Policy and th	
he run pa open the N luddyBar	uses at 93% Min Fish Fl rtlett_Info_	6 completion ow rule (in th RBSstart if	e main w the rulese	orkspace, sele et windows ha	ect Policy and the ve been closed).	
he run pa open the N luddyBar	uses at 93% Min Fish Fl rtlett_Info_	6 completion ow rule (in th RBSstart if	e main w the rulese	orkspace, sele	ect Policy and the ve been closed).	
the run pa open the N IuddyBar lote that t	uses at 93% Min Fish Fl rtlett_Info_ here is now	% completion ow rule (in th _RBSstart if y an added ma	e main w the rulese argin on th	orkspace, sele et windows ha ne left of the R	ect Policy and the ve been closed). Rule Editor.	
he run pa Open the N IuddyBar Tote that t	uses at 93% Min Fish Fl rtlett_Info_ here is now bottom of	% completion ow rule (in th RBSstart if y an added ma the column o	ne main w the rulese argin on th n the left o	orkspace, sele et windows ha ne left of the R of the Rule Ed	ect Policy and the ve been closed).	
he run pa Open the N IuddyBar lote that t lick at the ne rule) to	uses at 93% Min Fish Fl rtlett_Info_ here is now bottom of set a brea	% completion ow rule (in th _RBSstart if y an added ma	ne main w the rulese argin on th n the left o	orkspace, sele et windows ha ne left of the R of the Rule Ed	ect Policy and th ve been closed). Rule Editor. litor (below all o	f the expr
the run pa open the N IuddyBar lote that the lick at the ne rule) to S R 6 N	uses at 93% Min Fish Fl rtlett_Info_ here is now bottom of set a brea Min Fish Flow	6 completion low rule (in th RBSstart if 7 an added ma the column o kpoint at the	ne main w the rulese argin on th n the left o	orkspace, sele et windows ha ne left of the R of the Rule Ed	ect Policy and the ve been closed). Rule Editor.	f the expr
the run pa open the N IuddyBar fote that the lick at the ne rule) to S R 6 N Muddy	uses at 93% Min Fish Fl rtlett_Info_ here is now bottom of set a breat fin Fish Flow Reservoir.Ou	6 completion low rule (in th RBSstart if 7 an added ma the column of kpoint at the of utflow []	e main we the rulese argin on th n the left o end of the	orkspace, sele et windows ha ne left of the R of the Rule Ed e rule.	ect Policy and the ve been closed). Rule Editor. Note: The total state of total state	f the expr
the run pa open the N IuddyBar fote that the lick at the ne rule) to S R 6 N Muddy	uses at 93% Min Fish Fl rtlett_Info_ here is now bottom of set a brea Min Fish Flow Reservoir.Ou F (Muddy R	6 completion low rule (in th RBSstart if 7 an added ma the column of kpoint at the utflow [] utflow []	the main we the rulese argin on the n the left of end of the w [] < Mu	orkspace, sele et windows hav ne left of the R of the Rule Ed e rule. ddy Data.Min Fi	ect Policy and th ve been closed). Rule Editor. litor (below all o	f the expr
the run pa open the N IuddyBar fote that the lick at the ne rule) to S R 6 N Muddy	uses at 93% Min Fish Fl rtlett_Info_ here is now bottom of set a brea in Fish Flow Reservoir.Ou F (Muddy R Min (Mud	6 completion low rule (in th RBSstart if 7 an added ma the column o kpoint at the utflow [] dy Data.Min F	the main we the rulese argin on the n the left of end of the w [] < Mu ish Flow []	orkspace, sele et windows hav ne left of the R of the Rule Ed e rule. ddy Data.Min Fi	ect Policy and the ve been closed). Rule Editor. Note: The the total of	f the expr
the run pa open the N JuddyBar fote that the elick at the ne rule) to S R 6 N Muddy = I	uses at 93% Min Fish Fl rtlett_Info_ here is now bottom of set a brea Min Fish Flow Reservoir.Of F (Muddy R Min (Mud Max	6 completion low rule (in th RBSstart if 7 an added ma the column of kpoint at the utflow [] utflow []	the main we the rulese argin on the n the left of end of the w [] < Mu ish Flow []	orkspace, sele et windows hav ne left of the R of the Rule Ed e rule. ddy Data.Min Fi	ect Policy and the ve been closed). Rule Editor. Note: The the total of	f the expr
the run pa open the N JuddyBar fote that the elick at the ne rule) to S R 6 N Muddy = I	uses at 93% Min Fish Fl rtlett_Info_ here is now bottom of set a brea in Fish Flow Reservoir.Ou F (Muddy R Min (Mud	6 completion low rule (in th RBSstart if 7 an added ma the column o kpoint at the utflow [] dy Data.Min F	the main we the rulese argin on the n the left of end of the w [] < Mu ish Flow []	orkspace, sele et windows hav ne left of the R of the Rule Ed e rule. ddy Data.Min Fi	ect Policy and the ve been closed). Rule Editor. Note: The the total of	f the expr
the run pa open the N JuddyBar fote that the elick at the ne rule) to S R 6 N Muddy = I	uses at 93% Min Fish Fl rtlett_Info_ here is now bottom of set a brea Min Fish Flow Reservoir.Of F (Muddy R Min (Mud Max	6 completion low rule (in th RBSstart if 7 an added ma the column o kpoint at the utflow [] dy Data.Min F	the main we the rulese argin on the n the left of end of the w [] < Mu ish Flow []	orkspace, sele et windows hav ne left of the R of the Rule Ed e rule. ddy Data.Min Fi	ect Policy and the ve been closed). Rule Editor. Note: The the total of	f the expr
the run participation of the run participation	uses at 93% Min Fish Fl rtlett_Info_ here is now bottom of set a breat fin Fish Flow Reservoir.Ou F (Muddy R Min (Mud Max END IF	6 completion Iow rule (in th RBSstart if 7 an added ma the column of kpoint at the of utflow [] teservoir.Outflo ddy Data.Min F cOut (Muddy F	w [] < Mu ke servoir)	orkspace, select et windows hav ne left of the R of the Rule Ed or rule.	ect Policy and the ve been closed). Rule Editor. Note: The t	f the expr
the run participation of the run participation	uses at 93% Min Fish Fl rtlett_Info_ here is now bottom of set a brea Min Fish Flow Reservoir.Ou F (Muddy R Min (Mud Max END IF sign (octag	6 completion low rule (in th RBSstart if 7 an added ma the column of kpoint at the utflow [] teservoir.Outflo ddy Data.Min F COut (Muddy F gon) appears i	w [] < Mu ke servoir)	orkspace, select windows have the left of the Rule Ed of the Rule Ed or rule. ddy Data.Min Find the R of that a break point of the Rule and the R	ect Policy and the ve been closed). Rule Editor. Note: The the total of	f the expr

The run will pause at the breakpoint, and the **RPL Debugger** dialog will come to the front. Note that the middle panel of the debugger lists the single breakpoint. The top panel shows the current location in the evaluation of rules and functions, called the call stack.

Now we can see what values the different parts of the rule produce:

First, in the Min Fish Flow rule, select the brackets [] after Muddy Data.Min Fish Flow

The bottom panel in the Debugger reads "55 cfs".

Next, select the **MaxOut** function.

The bottom panel in the Debugger reads "49.97 cfs".

R	RPL Viewer -	MuddyBartlett_Inf				RPL Deb	ugger	-		x
File Edit F	Rule Statement View	*	File	Debug Br	reakpoint	s				R
Min Fish Flo	w 🗵][])	F ÇE	₽			
	Ain Fish Flow		all Sta	:k						
	IF (Muddy Reservoir.Out Min (Muddy Data.Min MaxOut (Muddy	Tish Flow [],	Caller Set Group Min Fis S MuddyBart P Muddy Res Rules							
E	END IF	E	reakpoi							
			١	Where	Whe	n	Set	Group		
				_		Execution	S MuddyBart	P Muddy F	Res R.	••
			49.97 "	Selected Ex						

Finally, select the brackets [] after Muddy Reservoir.Outflow

The bottom panel in the Debugger reads "49.97 cfs".

As the rule dictates, **Muddy Reservoir.Outflow** is set to the minimum of the two values. Although we wanted to release 55cfs, the **MaxOut** function computed that only 49.97cfs could be released based on the amount of water currently in the reservoir. This is the purpose of including the **MaxOut** function in the rule. It assures that the rule will not try to release more water than is physically possible, so an Outflow below the Min Fish Flow is the correct outcome of the operating rules.

Stop the debugging process by clicking the red **Stop** buttonin the **RPL Debugger**, then disable the Debugger in the **Run Control** dialog by clicking <u>kenabled</u>.

Un-check the box next to **Pause Before Timestep** and click **Start** to complete a run before ending the RiverWare session.

The run status reaches 100%.

Although debugging is a very advanced RiverWare topic, it is useful for even the novice user to know that it is possible to understand exactly what the software is doing. Debugging is critical for determining how the simulation proceeds. This section presented two tools for accessing the details within a rulebased simulation:

- **Diagnostics** provide a play-by-play of every step in the simulation. Once context filters are applied, you can pinpoint the exact interaction you want to analyze. More information on this topic can be found in the **Diagnostics** section of the help.
- The **RPL debugger** takes you inside a rule to see what values are produced, in order to check that you have the proper logic. More information on this topic can be found in the **RPL Debugging Tools** section of the help.