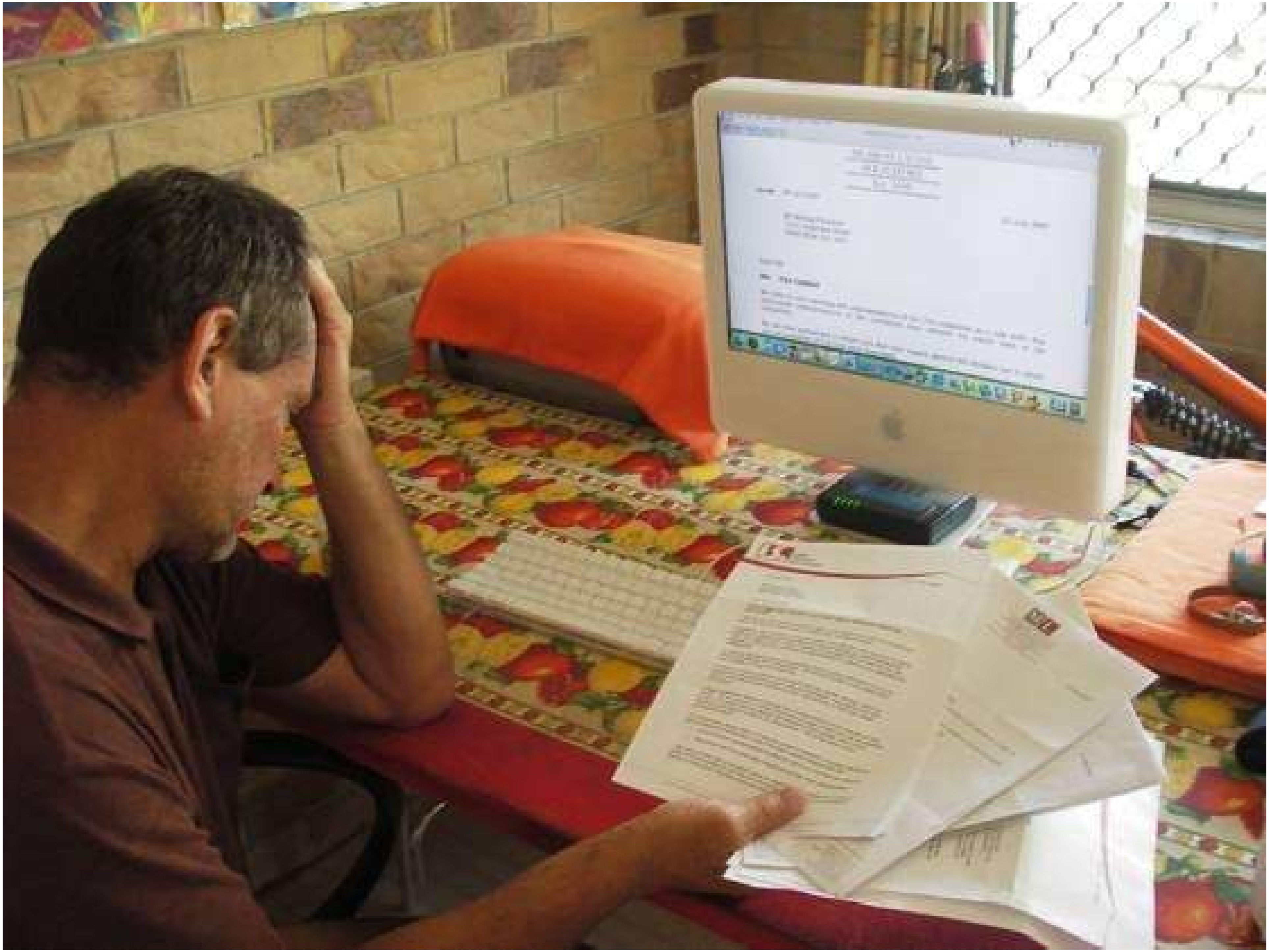
Implementation of the Water Rights Solver vs Rule Solution for the Colorado River, TX

Brad Vickers, Wave Engineering Inc.

History

- Presented how to solve water rights with accounting at users group four years ago
- Rule version of LCRA model to insure that concept would work
- LCRA picked up funding of implementing water rights about three years ago (pretty much finished 6 to 8 months ago)



Implementation of Water Rights

- Seven Phases
  - Initialization
  - Compute Demands
  - Solve for water rights
  - Meet remaining demands from storage
  - Environmental
  - Physical
  - Summary

Initialization

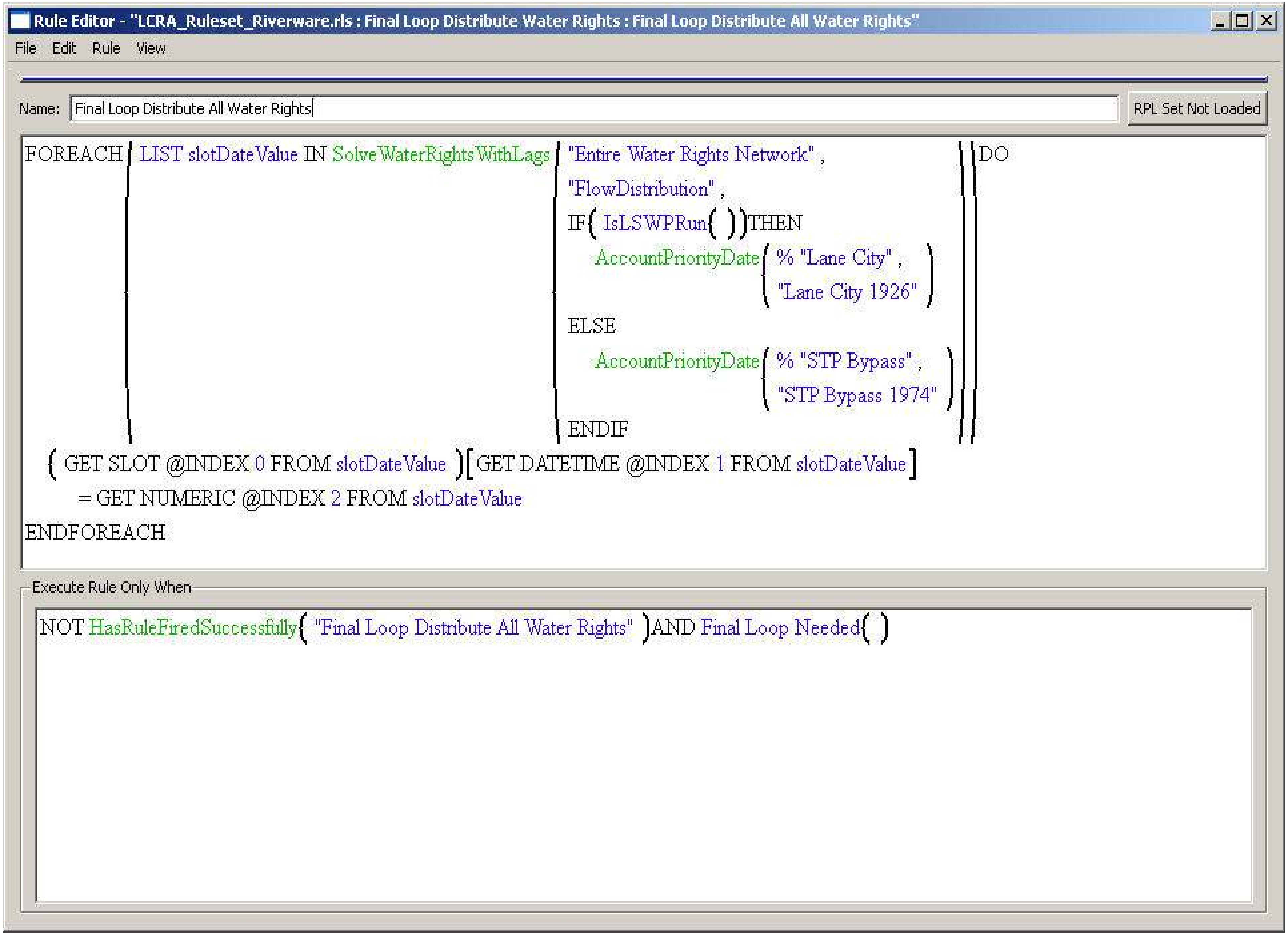
- Set up accounting to solve
- Set up model to solve physically

Compute Demands

- Solver version Riverware calculates water rights demands with computational subbasins (user simply inputs annual demands and disaggregation tables, computes storage account demands via various methods)
- Rules only version user must make rules to disaggregate and set diversion requests for both diversion and storage accounts

Solve Water Rights

- Solver version simple call to pre-defined function
- Rules version 2 rules for every water right every time it needs to solve. Obviously this takes lots of rules.


Rule Editor - "LCRA_Ruleset_Riverware.rls : Final Loop Distribute Water Rights : Final Loop Distribute All Water Rights"
File
Edit
Rule
View
RPL Set Not Loaded
Execute Rule Only When

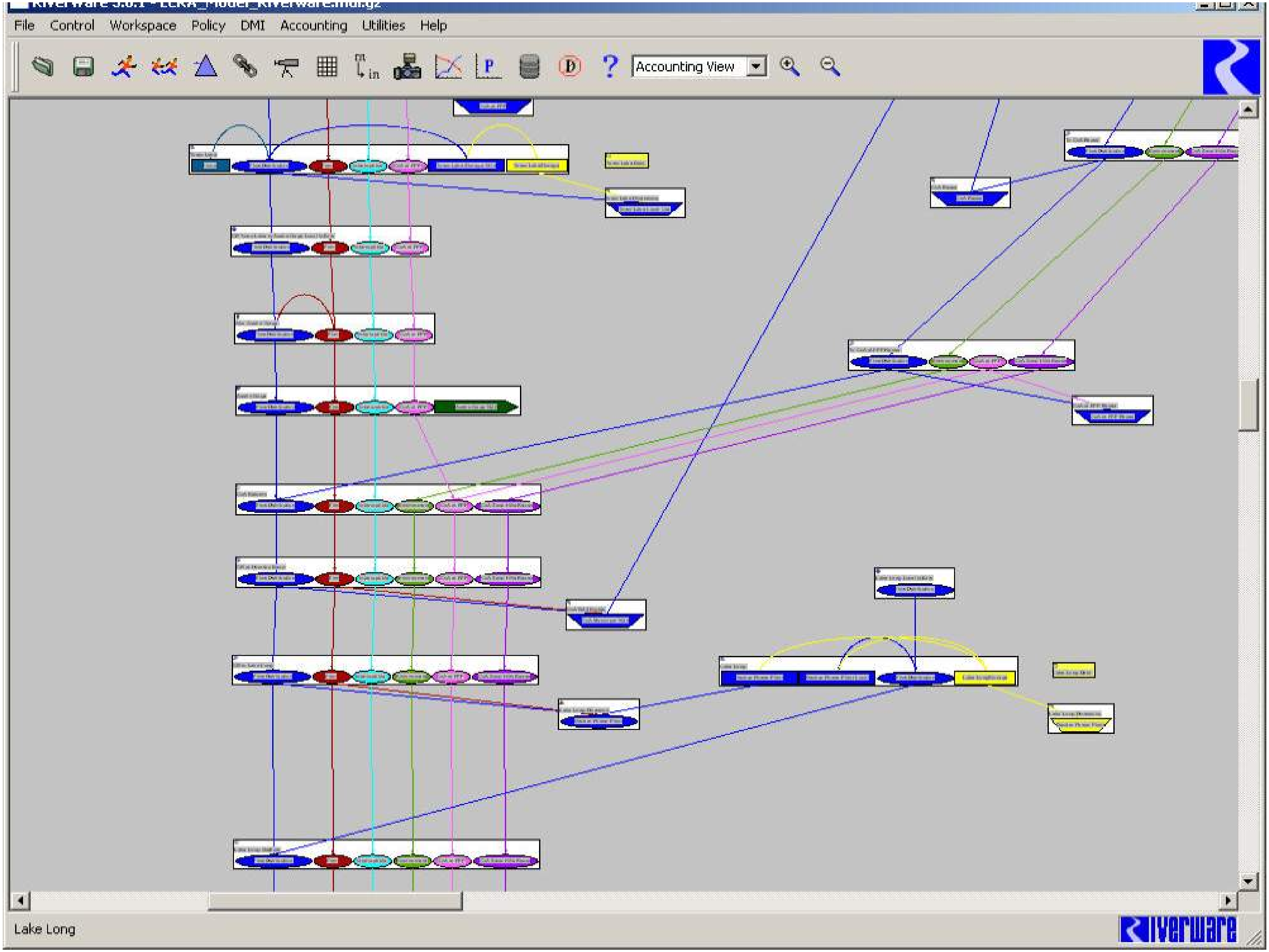
Name: Final Loop Distribute All Water Rights

```
FOREACH ( LIST slotDateValue IN SolveWaterRightsWithLags ( "Entire Water Rights Network" ,      ) ) DO
                                                           "FlowDistribution" ,
                                                           IF ( IsLSWPRun( ) ) THEN
                                                              AccountPriorityDate( % "Lane City" ,
                                                                                   "Lane City 1926" )
                                                           ELSE
                                                              AccountPriorityDate( % "STP Bypass" ,
                                                                                   "STP Bypass 1974" )
                                                           ENDIF
   ( GET SLOT @INDEX 0 FROM slotDateValue ) [ GET DATETIME @INDEX 1 FROM slotDateValue ]
      = GET NUMERIC @INDEX 2 FROM slotDateValue
ENDFOREACH
```

```
NOT HasRuleFiredSuccessfully( "Final Loop Distribute All Water Rights" ) AND Final Loop Needed( )
```

Meet Remaining Demands from Storage

- Provide storage water to water users
- User must supply logic
- In solver version this was made much simpler by the addition of a new pre-defined function named ObjAcctSupplyByWaterTypeRelTypeDestType
  - With this function subbasins no longer had to be maintained
  - Access to stored water made by simply changing ReleaseType/Destination of supplyt

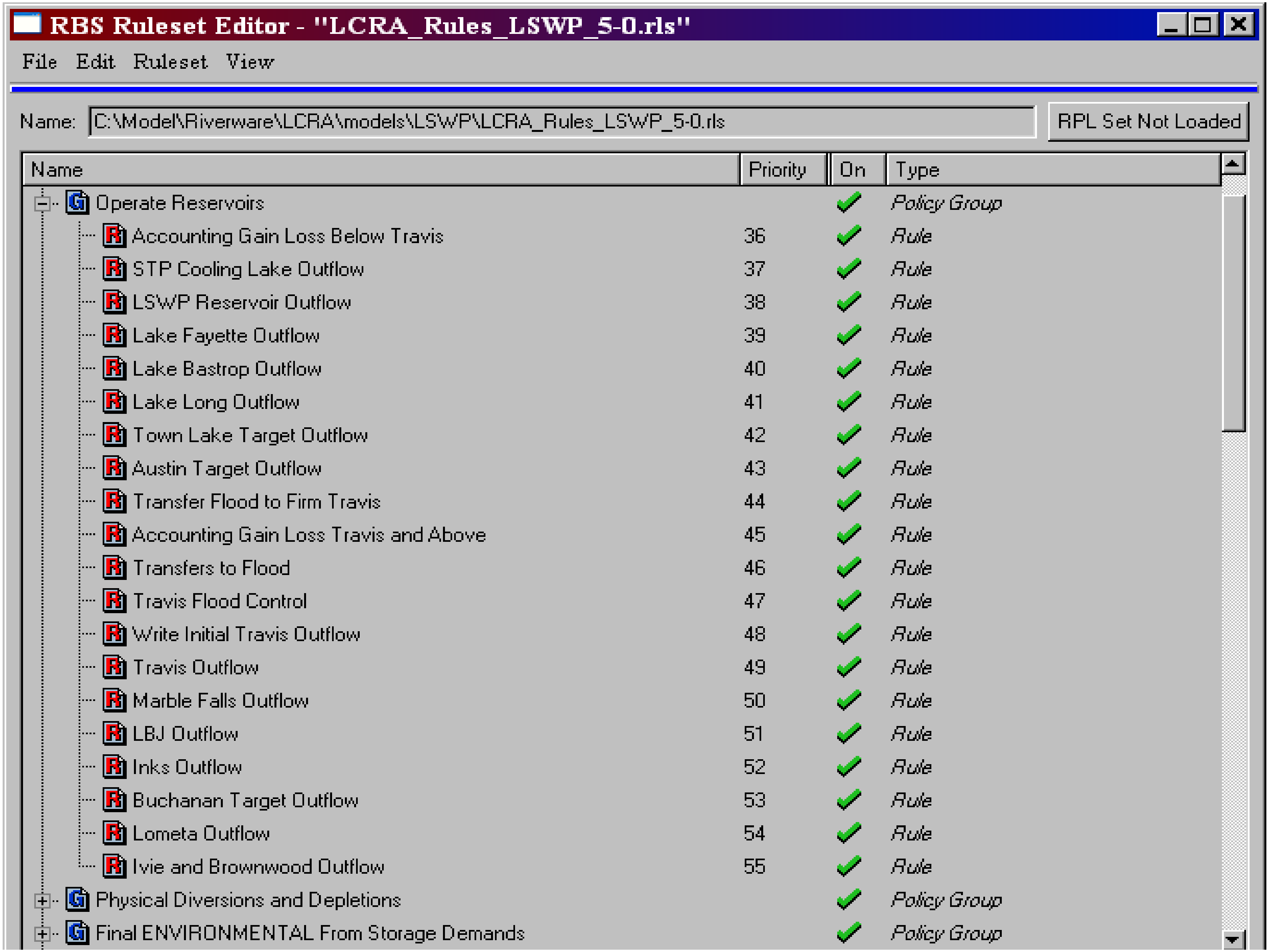

RiverWare 5.0.1 - LCRA_Model_RiverWare.mdl.gz
File
Control
Workspace
Policy
DMI
Accounting
Utilities
Help
Accounting View
Lake Long
RiverWare

Environmental

- Most difficult part to get right
- Iterative solution

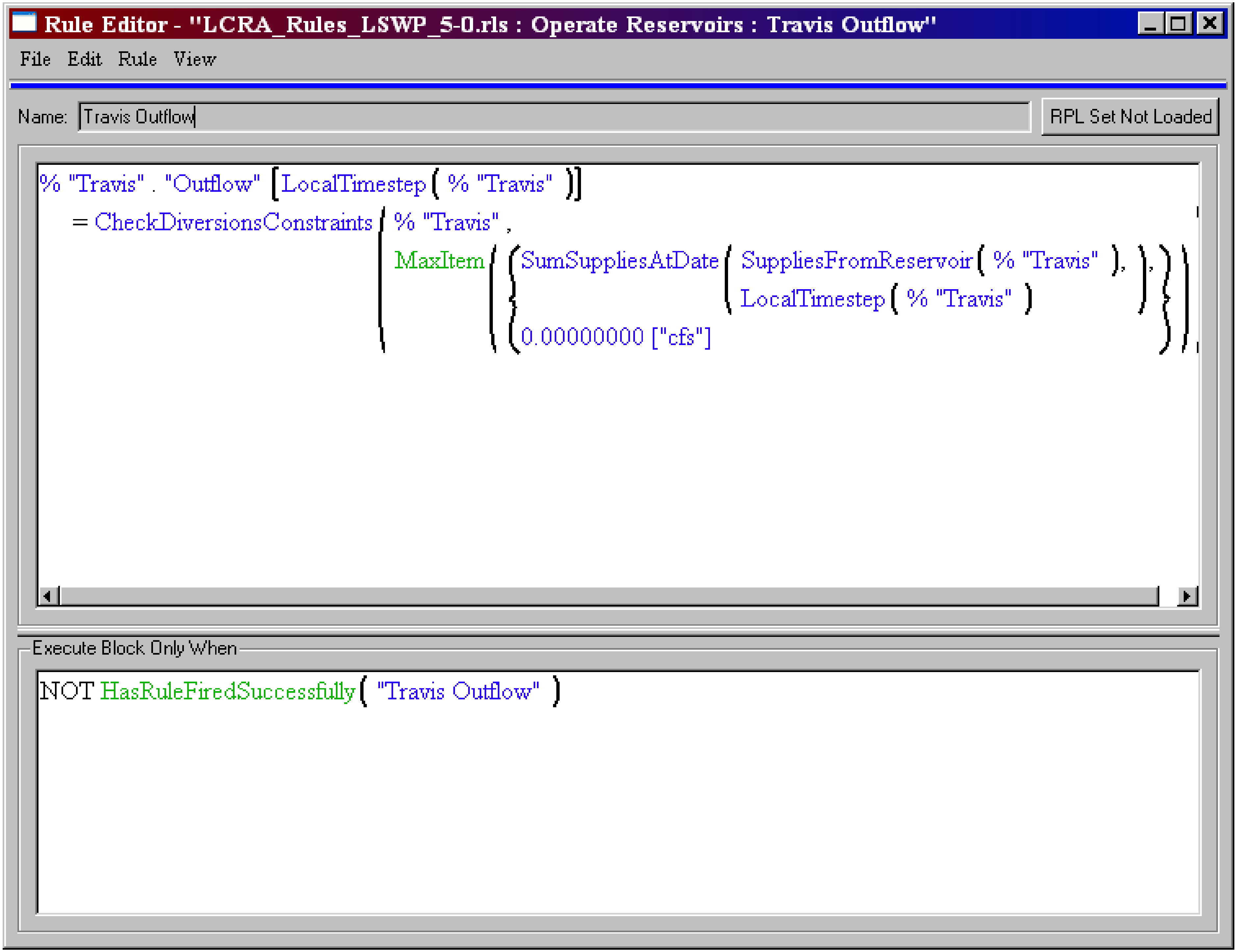
Physical Operation

- All “real” modeling is done in accounting
- Physical Operation is mostly simply summing up accounting supplies
- If physical operations (such as flood control) change outflows, the accounting system and physical system must be reconciled by user



Name: C:\Model\Riverware\LCRA\models\LSWP\LCRA_Rules_LSWP_5-0.rls

Name	Priority	On	Type
Operate Reservoirs		✓	*Policy Group*
Accounting Gain Loss Below Travis	36	✓	*Rule*
STP Cooling Lake Outflow	37	✓	*Rule*
LSWP Reservoir Outflow	38	✓	*Rule*
Lake Fayette Outflow	39	✓	*Rule*
Lake Bastrop Outflow	40	✓	*Rule*
Lake Long Outflow	41	✓	*Rule*
Town Lake Target Outflow	42	✓	*Rule*
Austin Target Outflow	43	✓	*Rule*
Transfer Flood to Firm Travis	44	✓	*Rule*
Accounting Gain Loss Travis and Above	45	✓	*Rule*
Transfers to Flood	46	✓	*Rule*
Travis Flood Control	47	✓	*Rule*
Write Initial Travis Outflow	48	✓	*Rule*
Travis Outflow	49	✓	*Rule*
Marble Falls Outflow	50	✓	*Rule*
LBJ Outflow	51	✓	*Rule*
Inks Outflow	52	✓	*Rule*
Buchanan Target Outflow	53	✓	*Rule*
Lometa Outflow	54	✓	*Rule*
Ivie and Brownwood Outflow	55	✓	*Rule*
Physical Diversions and Depletions		✓	*Policy Group*
Final ENVIRONMENTAL From Storage Demands		✓	*Policy Group*


Rule Editor - "LCRA_Rules_LSWP_5-0.rls : Operate Reservoirs : Travis Outflow"
File
Edit
Rule
View
RPL Set Not Loaded

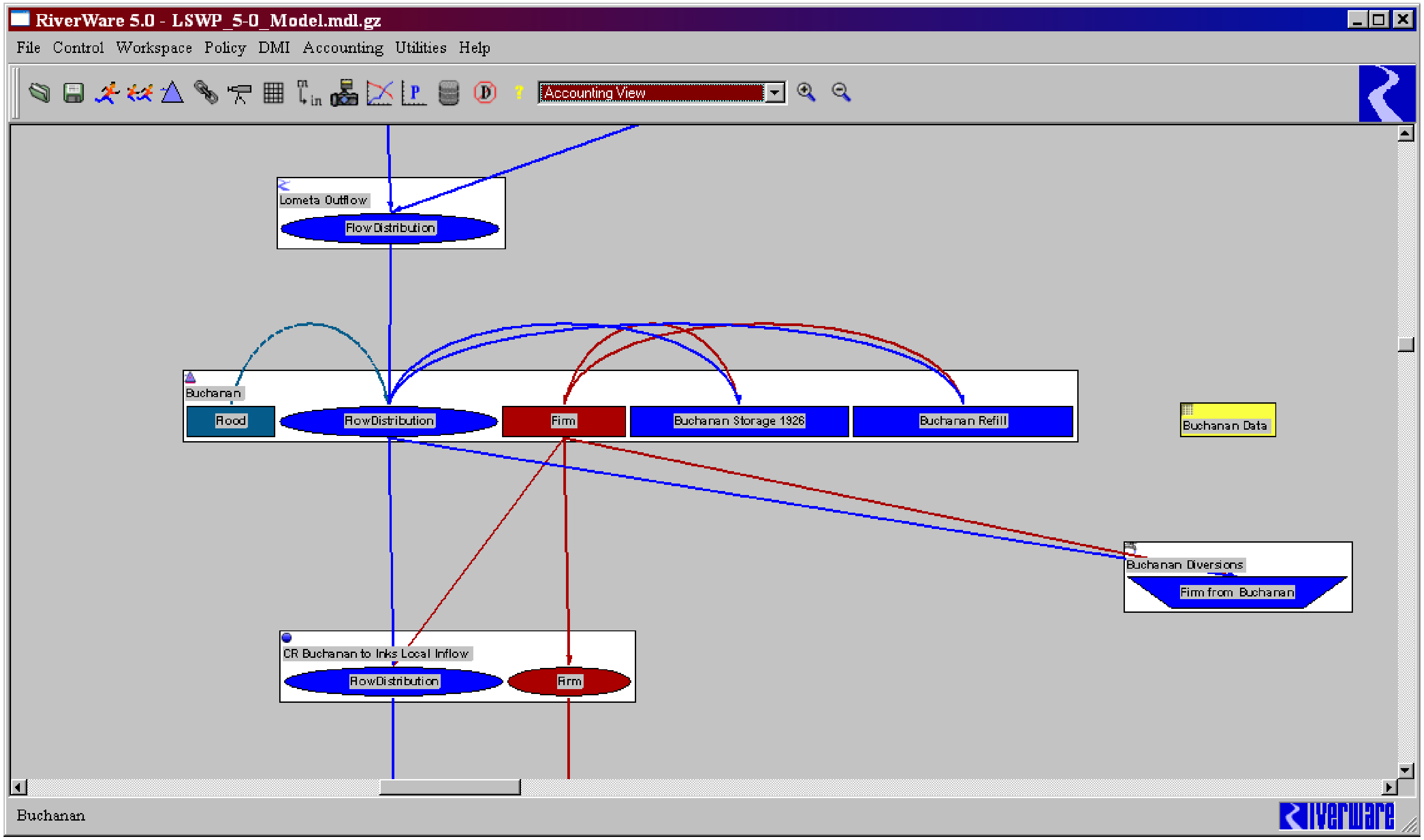
Name: Travis Outflow

```
% "Travis" . "Outflow" [LocalTimestep( % "Travis" )]
    = CheckDiversionsConstraints( % "Travis" ,
                                  MaxItem( { SumSuppliesAtDate( SuppliesFromReservoir( % "Travis" ),
                                                                LocalTimestep( % "Travis" ) ) ,
                                             0.00000000 ["cfs"] } ) ),
```

Execute Block Only When

```
NOT HasRuleFiredSuccessfully( "Travis Outflow" )
```

Flood Control



Summarization

- Sum Daily results into Monthly
- Sum Monthly results into Annual

Reduction in Rules

- Solver version about 150 rules
- Rules version about 1800 rules
- About a 92% reduction in the number of rules
- Bad side effect every rule and function had to be rewritten

Reduction in Objects

- Objects in rules version 380
- Objects in solver version 290
- Reduction of about 25 %

Improvements to Run Time

- Solver version 10 minutes for 730 timesteps
- Rules version (prototype) 45 minutes 730 timesteps
- ~ 75 % improvement in runtime

Items to improve water rights solution in Riverware

- Implementation of annual (or shorter timeframe) limits of diversion.
  - This will involve the addition of having accruals in Riverware work on a supply by supply basis
- Ability to have subordinated senior rights upstream of junior rights
- Controller available that allows accounting to solve first

Lessons learned

- Development always takes 3 times longer than your best estimate
- Implementation of water rights in Riverware would have been much more difficult and maybe impossible without rules prototype
- Proto-type will probably have to be totally re-written, but lessons learned will deliver a better finished product
- The finished product compared to the prototype should be like the different between

Catching this

And this

In the end you can claim