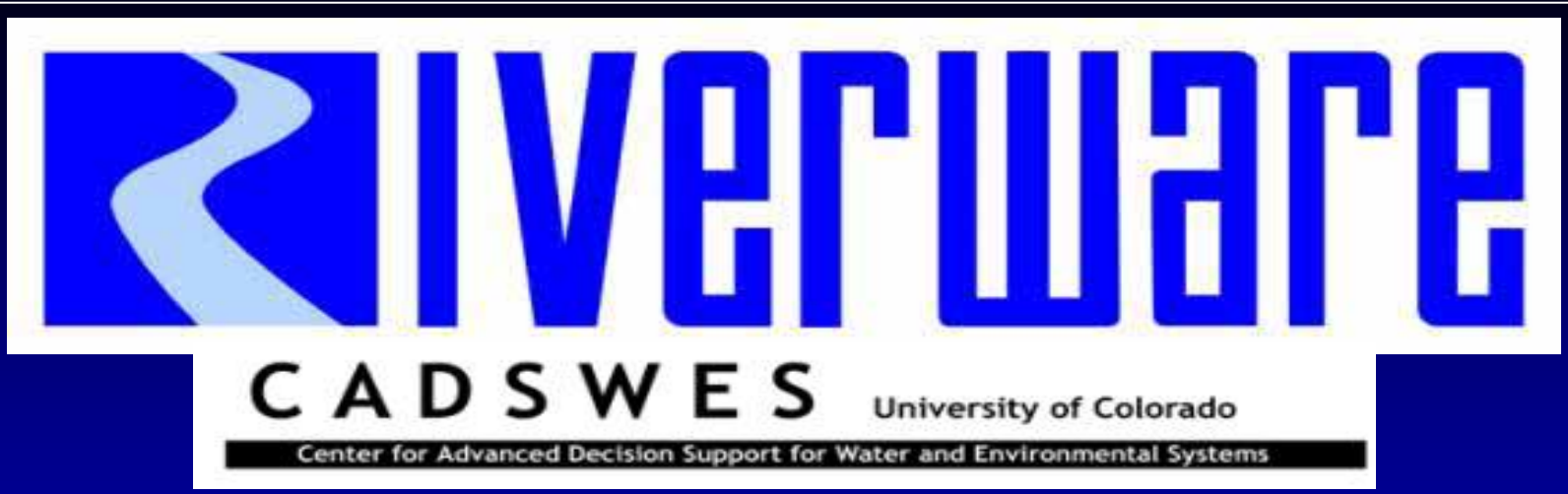


Water Accounting Modeling

RiverWare User Group Meeting
February 6 - 7, 2007

Presentation Outline

- Water Accounting Capabilities
  - Water Accounting Motivation and Definitions
  - Accounts
  - Supplies
  - Solution Algorithm
  - User Defined Accounting Methods
  - Exchanges
  - Demonstration as we go….
- Demonstration of Water Accounts Summary utility
- Accounting Class Outline
- Prioritized Water Rights Modeling and Allocation

Motivation

- Basins are operated for many purposes
- In many basins, it is necessary to track not only volume of water but the ownership and type
- Operating decisions are dependent on a user's available water, legal restrictions, physical constraints, and exchange mechanism
- Priority water rights allocation

- A framework is necessary to model water type and ownership

Solution:

Water Accounting

Requirements for Water Accounting

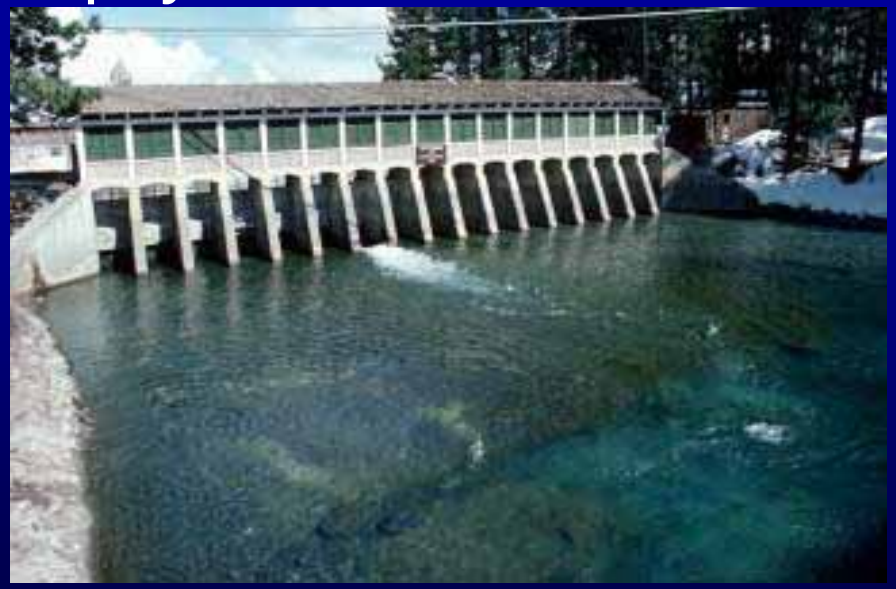
- Track ownership and type of water through all objects and at all timesteps
- Flexible to model accounting in any basin with unique policies and structure
- Operating decisions look at and set account, transfer and exchange values
- Must be able to allocate water based on Water Right Priority date
- Must be able to visualize the accounting network

How is water accounting modeled in RiverWare?

- Physical and Paper water are modeled separately;
  - There is a separate network of accounts on the simulation objects
  - Accounts are linked indicating the possible transfers
- Legal Accounts – Storage, Diversion, Instream Flow
- Non Legal - Passthrough accounts track transfer of water between legal accounts
- Accounts are labeled by ownership and type and can be given a priority date
- Rules can access accounting information and also set account transfers
- Can simulate water accounting components like accrual, exchanges, carryover, allocation, etc.

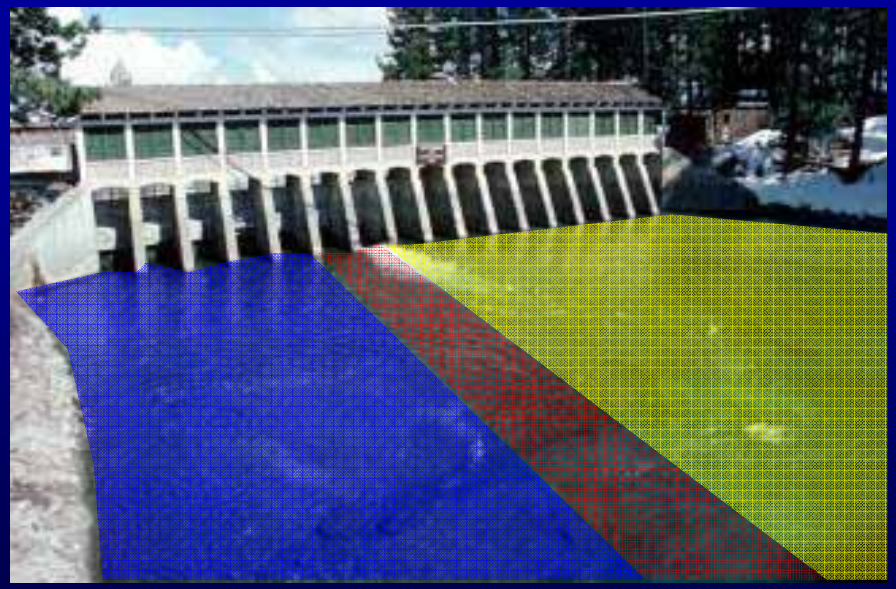
“Physical” vs. “Paper” water in RiverWare

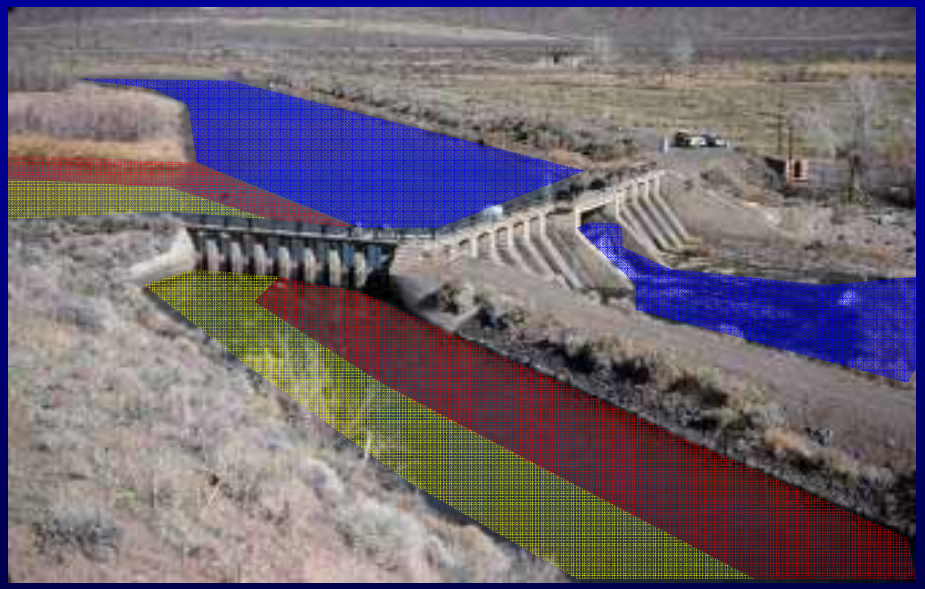
- Physical (wet) Water:
  - Total volume/flow of water in a river basin object. For example, storage in a reservoir represents the volume of physical water in the reservoir



“Physical” vs. “Paper” water in RiverWare

- Paper (colored) Water - type and ownership:
  - Volume/flow of water classified by type or ownership. For example, a certain agency owns 500AF of physical water in the reservoir.

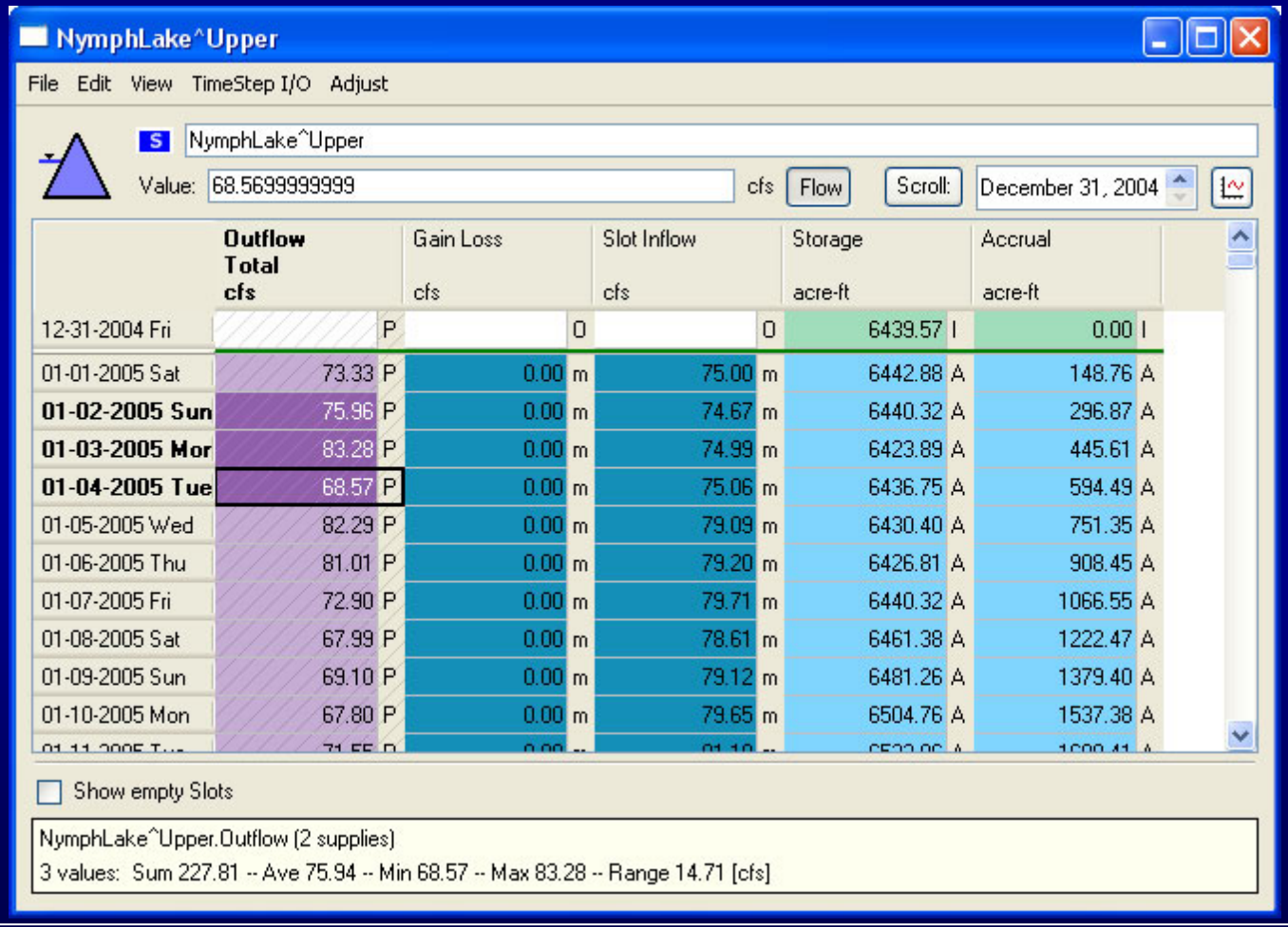




Types of Accounts

- Legal Accounts:
  - Storage Account - on Reservoirs
    - Storage, gain/loss, accrual, carry-over, inflow and outflow, transfers
  - Diversion Account - on Water Users, Aggregate Diversions
    - diversion, consumption, and return flow
  - Instream Flow – Track total flow into all accounts
- Non-Legal Accounts
  - Pass-through Accounts - Reaches, Confluences, Gages, Reservoirs, etc…
    - keep track of water moving between legal accounts

Storage Account

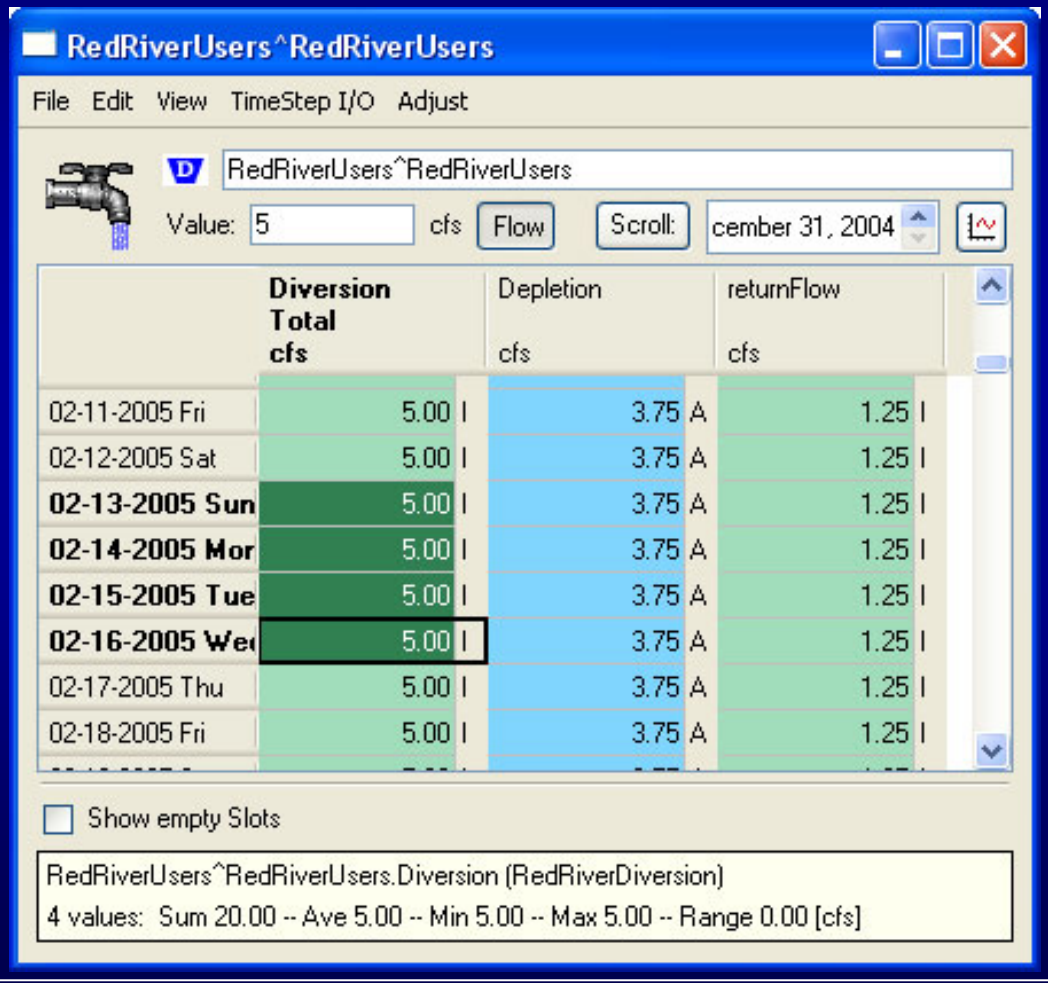


	Outflow Total cfs		Gain Loss cfs		Slot Inflow cfs		Storage acre-ft		Accrual acre-ft	
12-31-2004 Fri		P		0		0	6439.57	I	0.00	I
01-01-2005 Sat	73.33	P	0.00	m	75.00	m	6442.88	A	148.76	A
01-02-2005 Sun	75.96	P	0.00	m	74.67	m	6440.32	A	296.87	A
01-03-2005 Mon	83.28	P	0.00	m	74.99	m	6423.89	A	445.61	A
01-04-2005 Tue	68.57	P	0.00	m	75.06	m	6436.75	A	594.49	A
01-05-2005 Wed	82.29	P	0.00	m	79.09	m	6430.40	A	751.35	A
01-06-2005 Thu	81.01	P	0.00	m	79.20	m	6426.81	A	908.45	A
01-07-2005 Fri	72.90	P	0.00	m	79.71	m	6440.32	A	1066.55	A
01-08-2005 Sat	67.99	P	0.00	m	78.61	m	6461.38	A	1222.47	A
01-09-2005 Sun	69.10	P	0.00	m	79.12	m	6481.26	A	1379.40	A
01-10-2005 Mon	67.80	P	0.00	m	79.65	m	6504.76	A	1537.38	A

Storage Account – Mass Balance

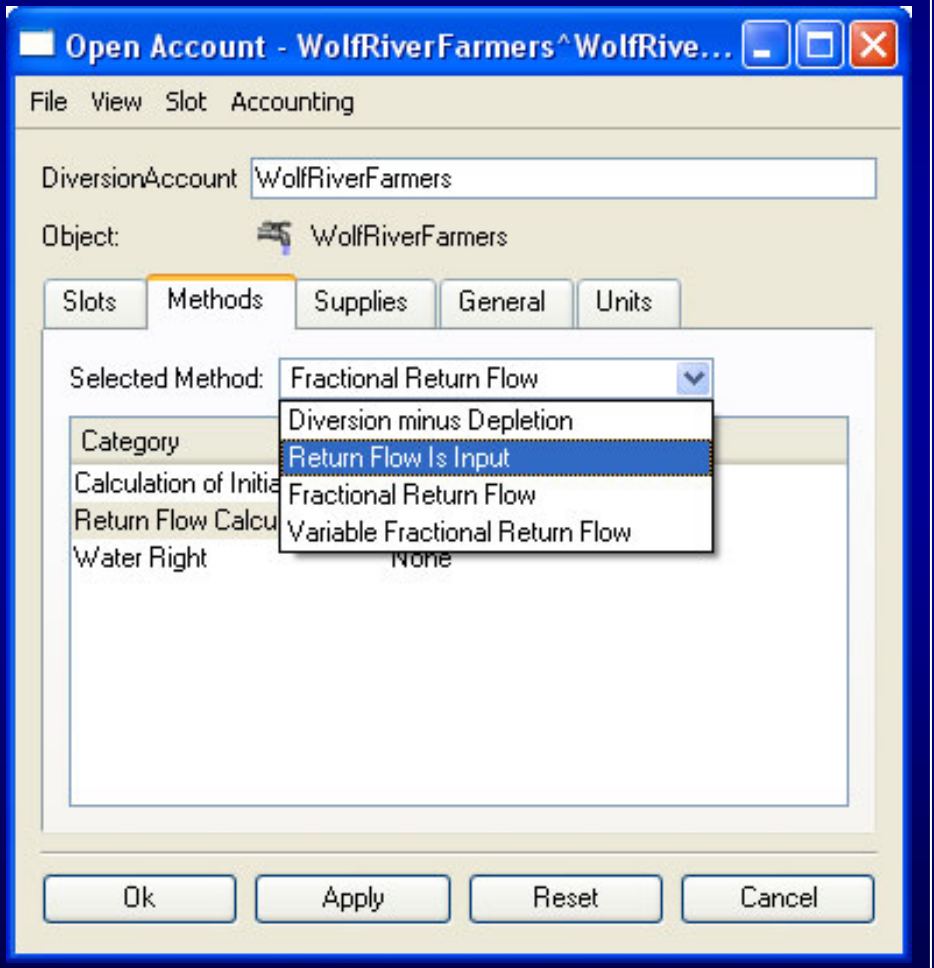
- Storage(t) = Storage(t-1) + GainLoss(t) + TimestepLength*(Inflow(t) – Outflow(t) + SlotInflow(t) – Diversion(t) – ReturnFlow(t) + TransfersIn(t) – TransfersOut(t))
- Accrual(t) = Accrual(t-1) + TimestepLength*(Inflow(t) + SlotInflow(t))
  - Required Knowns: Storage(t-1), Accrual(t-1), Outflow(t), Slot Inflow(t), Gain Loss(t)
  - Note: if it is the begin accrual date, use BeginYearAllocation(t) in place of Storage(t-1) and add in Carryover(t)

Diversion Account



Diversion Account – Mass Balance

- Depends on selected return flow method
- Depends on knowns and unknowns



Diversion Account – Mass Balance

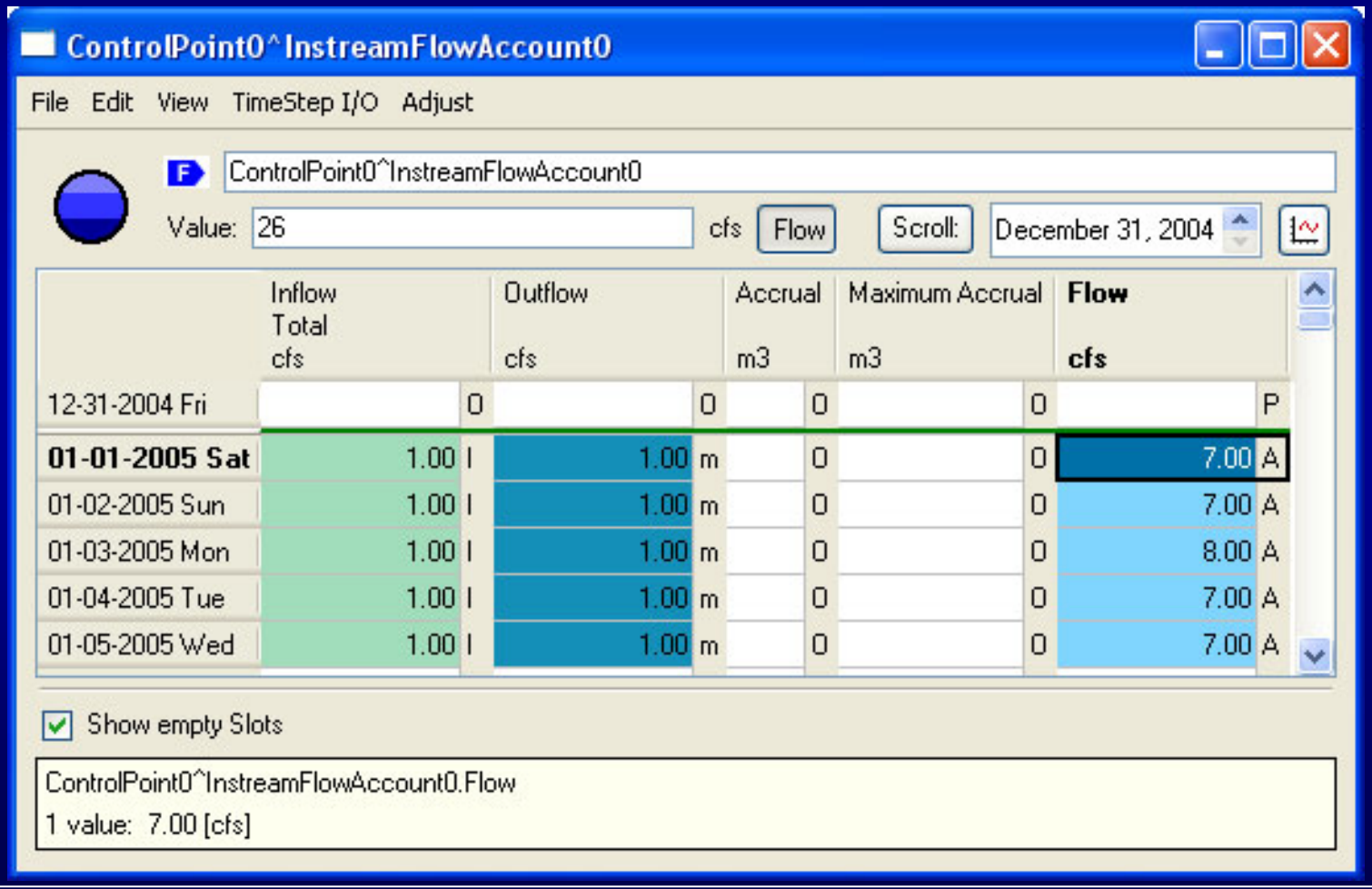
- Diversion minus Depletion
  - ReturnFlow = Diversion – Depletion

  OR
  - Depletion = Diversion – ReturnFlow
  - Also: Accrual = Accrual(t-1) +(Depletion)(TimestepLength)
- Return Flow is Input
  - Depletion = Diversion – ReturnFlow
  - Accrual = Accrual(t-1) + (Depletion)(TimestepLength)

Diversion Account – Mass Balance

- Fractional Return Flow
  - ReturnFlow = (Diversion)(FractionOfDiversion)
  - Depletion = Diversion – ReturnFlow
  - Accrual = Accrual(t-1) + (Depletion)(TimestepLength)
- Variable Fractional Return Flow
  - ReturnFlow = (Diversion)(VariableFractionOfDiversion)
  - Depletion = Diversion – ReturnFlow
  - Accrual = Accrual(t-1) + (Depletion)(TimestepLength)

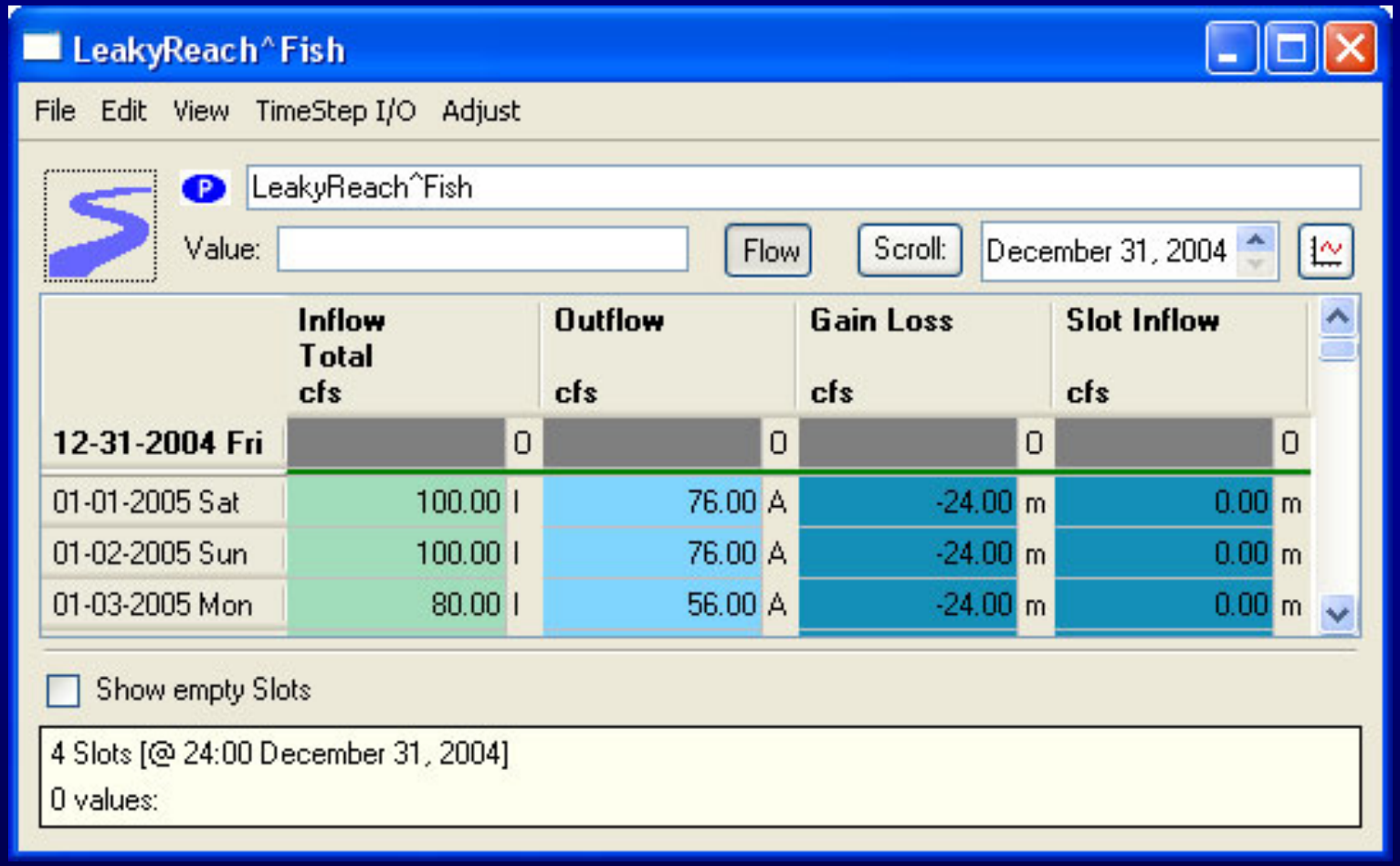
Instream Flow Account



Instream Account – Mass Balance

- Outflow = Inflow
- Accrual = Accrual(t-1) +(Inflow)(TimestepLength)
- Flow = Σ (All account inflows into object)

Passthrough Account

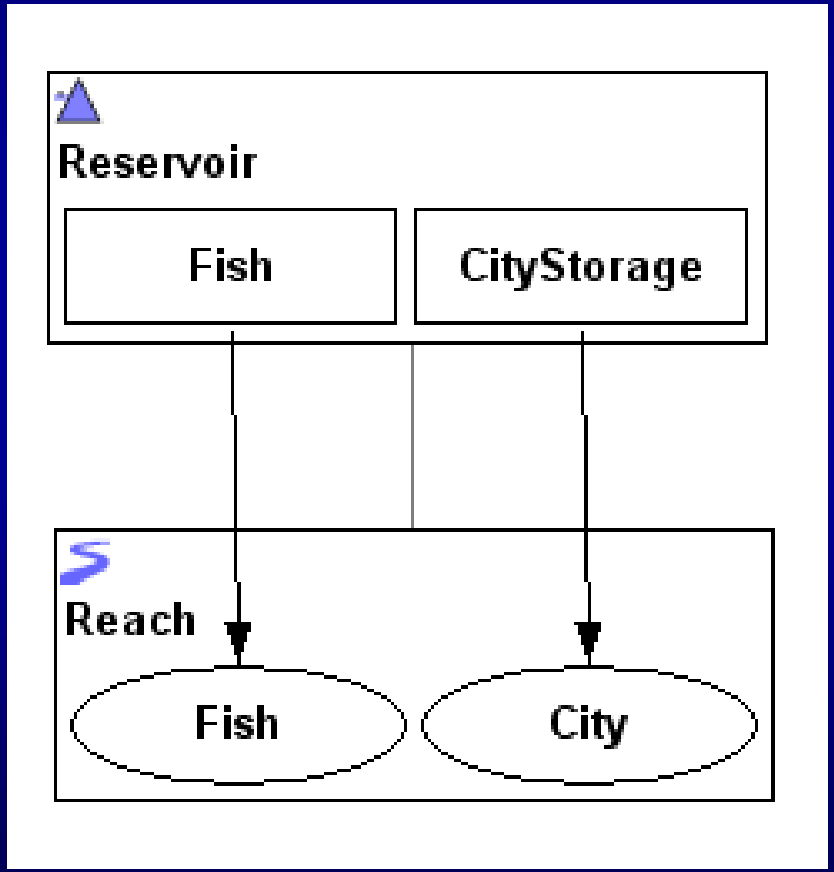


Passthrough Account – Mass Balance

- General equation:
  - Outflow = Inflow + Slot Inflow – Gain Loss – Diversion + Return Flow + Transfers In – Transfers Out
  - Required knowns: Either Inflow, Gain Loss, or Slot Inflow must be known for the account to solve. If none are known, the account will not solve. If one or two of these slots are known, the other(s) default to zero
- On reservoirs, temporary storage can be allowed when operating constraints preclude releasing all inflows

Linking Accounts

- Accounting network is formed by links between account called “supplies”
- Supply definition:
  - A supply is used to link two accounts. Specifically, a supply to an account means that paper water is moving into that account from another (often upstream) account.



Optional Properties of Accounts and Supplies

- Account
  - Water Type – For example, trans-basin water or local water
  - Water Owner – For example, Contractor A, Contractor B, and City
- Supply
  - Type – Inflow/Outflow, Diversion/Return Flow, Transfer
  - Destination – For example, Diversion A, Red Reach
  - Release Type – For example, Diversion, Fish

Accounting Solution

- "Spreadsheet" type solution
  - Account solves when it has the required knowns
  - Account solves whenever it gets a new value in any of the mass balance components (not controlled by dispatcher during a run)
  - Account slots and supplies are set by user input or rules

Accounting Solution

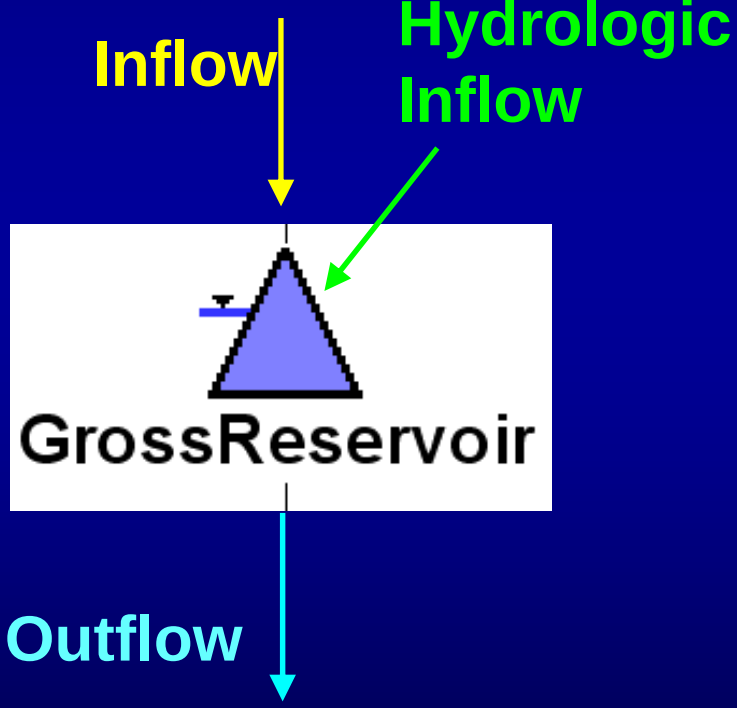
- Mass balance solutions:
  - Storage accounts solve only for storage balance (user specifies storage account outflows)
  - Passthrough accounts solve only downstream for outflow
  - Diversions accounts solve for depletion and/or return flow
  - Instream Flow accounts solve for
    - Flow (Total accounting flow into object)
    - Outflow

Object Level Accounting Methods

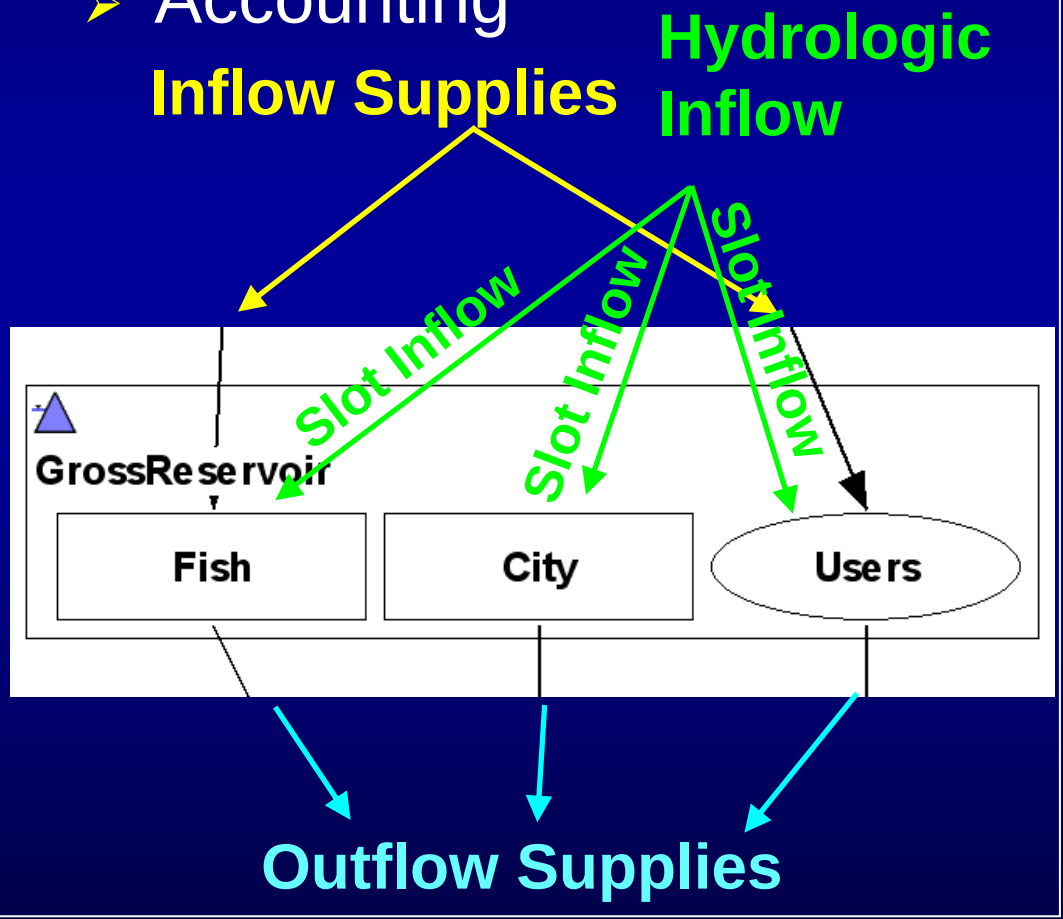
- Purpose: Distribute physical water on simulation objects to the accounts
- Methods are on the object and apply to all accounts
- Categories
  - Gain Loss: allocate physical gains and losses
  - Slot Inflow: allocate local inflows
  - Reconciliation: have accounting system match physical
- Two ways to specify the methods
  - Compiled: Simple methods or basin specific
  - User Defined: Written in the RiverWare Policy Language (RPL)

Slot Inflow Example

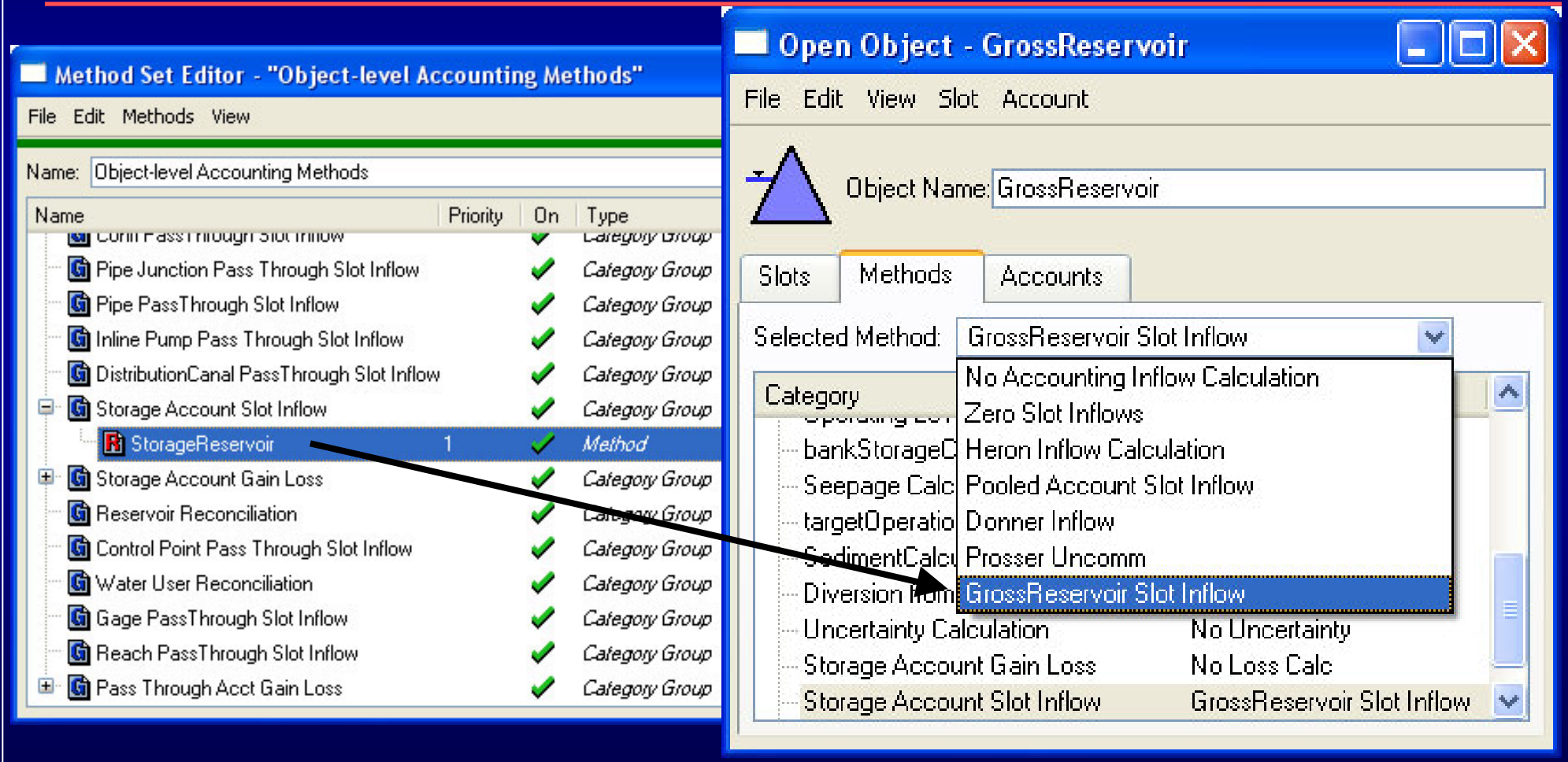
- Physical



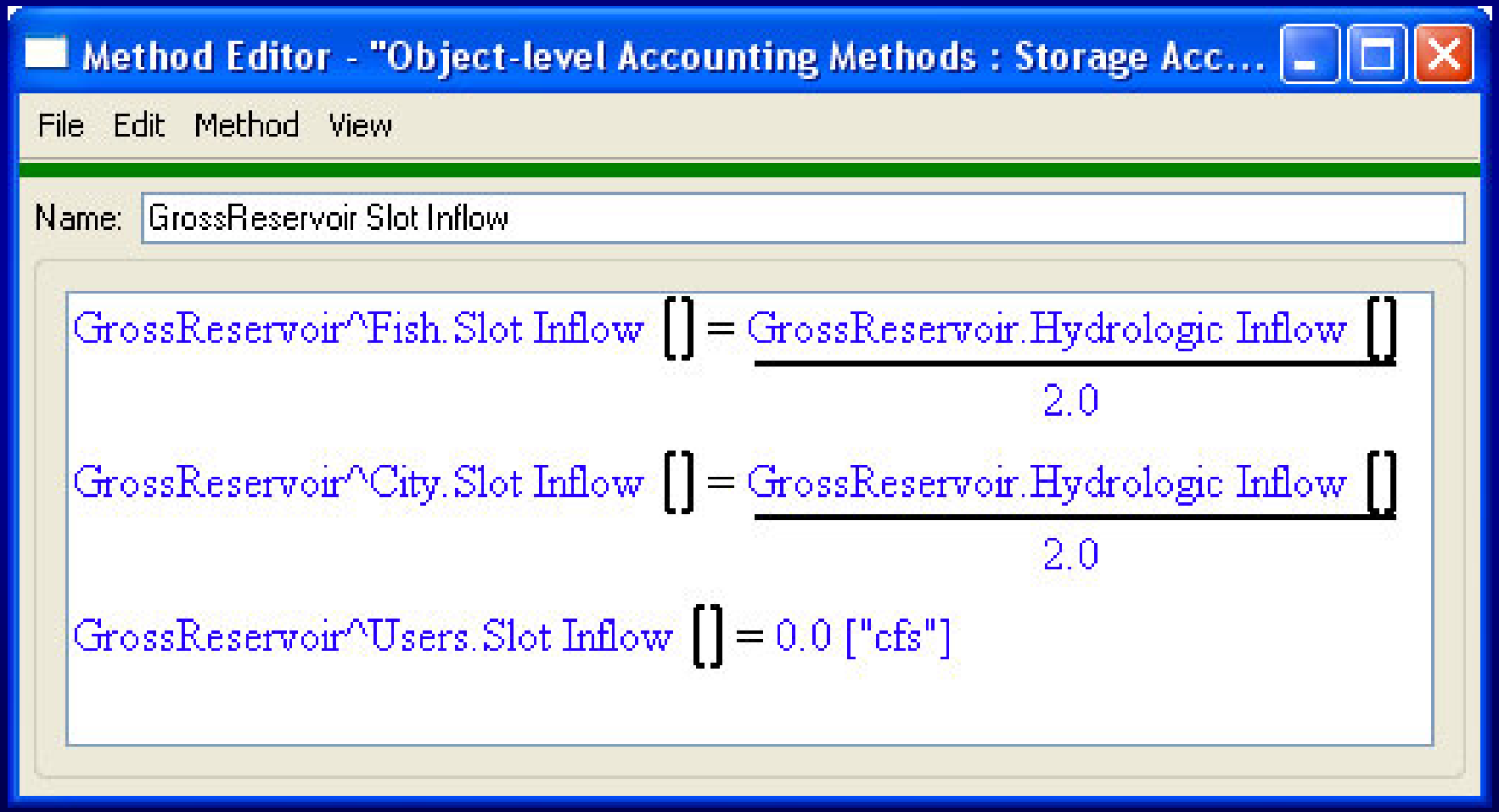
- Accounting



Slot Inflow Example (cont.)



Slot Inflow Example (cont.)



Use Rules to Control Accounting

Rules can be used to set supplies (links between accounts)

For example:

FishSupplyName.Supply [ ] =

FlowTarget() -
SumOtherReservoirReleasesOfType(“Fish”)

If account Slot Inflow and Gain Loss are known, this causes the account to solve for Storage

Predefined functions for Accounting

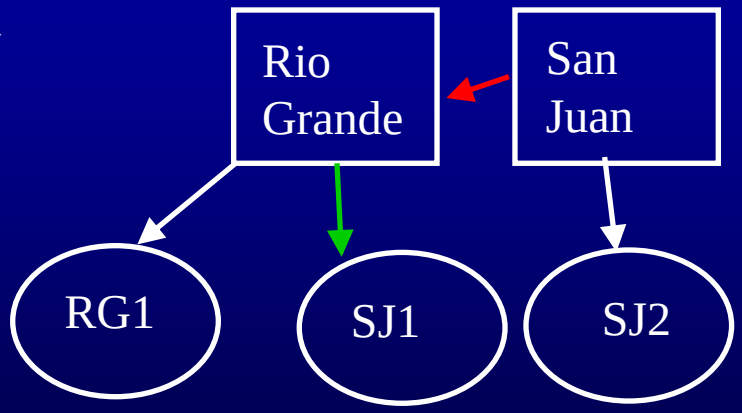
- Many predefined functions specific to Accounting, for example:
  - AccountNamesByWaterType
  - DestinationsFromObjectReleaseType
  - ObjectsFromAccountName
  - SumAccountSlotsByWaterType
  - WaterOwners
  - ….
- Useful to generalize rule writing

Reconciliation

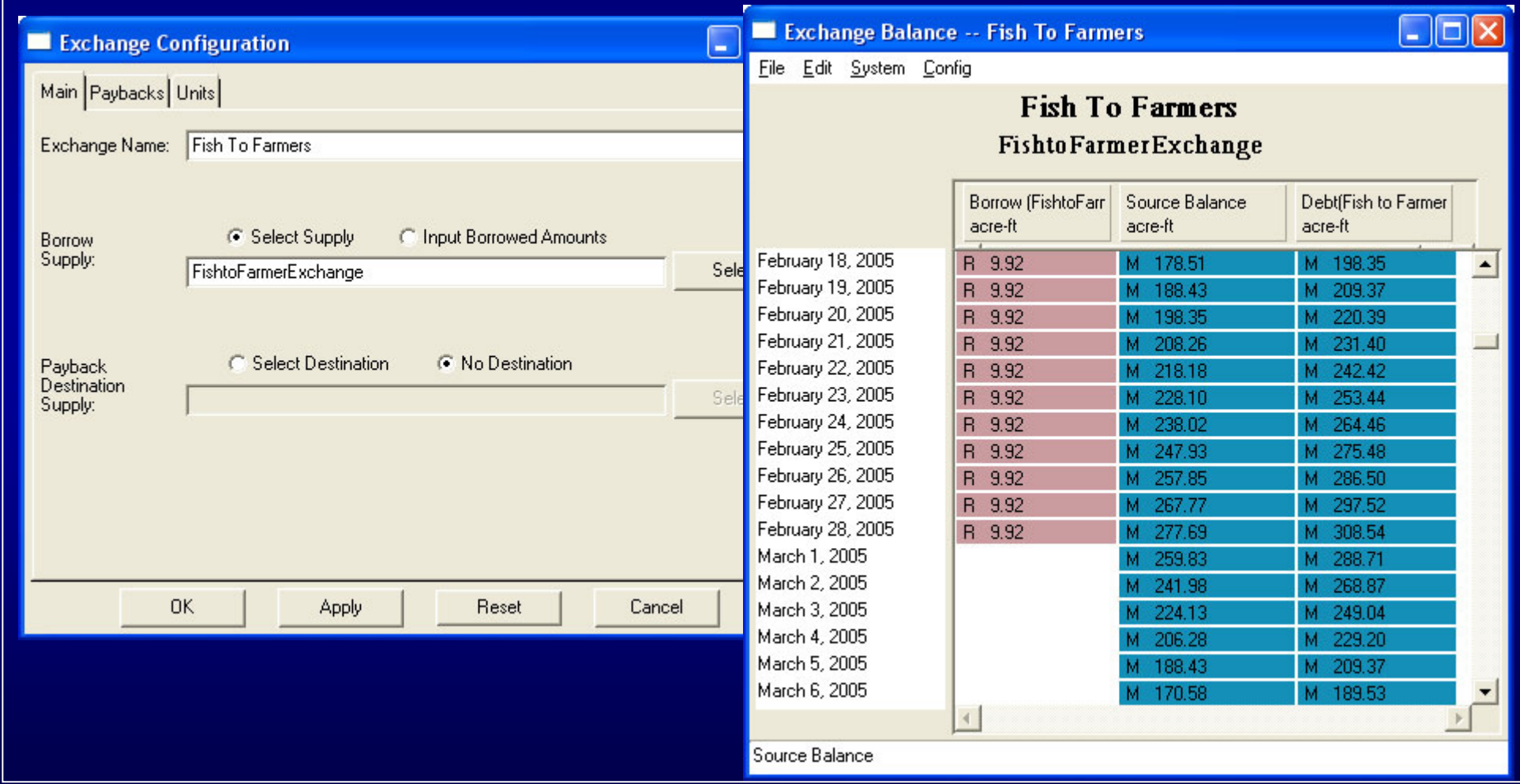
- Total physical releases/storage does not have to equal accounting releases/storage
- Matching the accounting system to the physical system is up to the user using:
  - Object Level Accounting Methods
  - Rules

Water Exchanges

- Track water exchanges and transfers
- Borrow - a supply to an account or a user input
- Source Payback - an outflow (demand) from an account
- Destination Payback (optional) - a supply to an account
- In the exchange utility, the debt is updated whenever a new value is set in the borrow or payback
- Rules can access the debt using predefined functions



Exchange Manager Utility

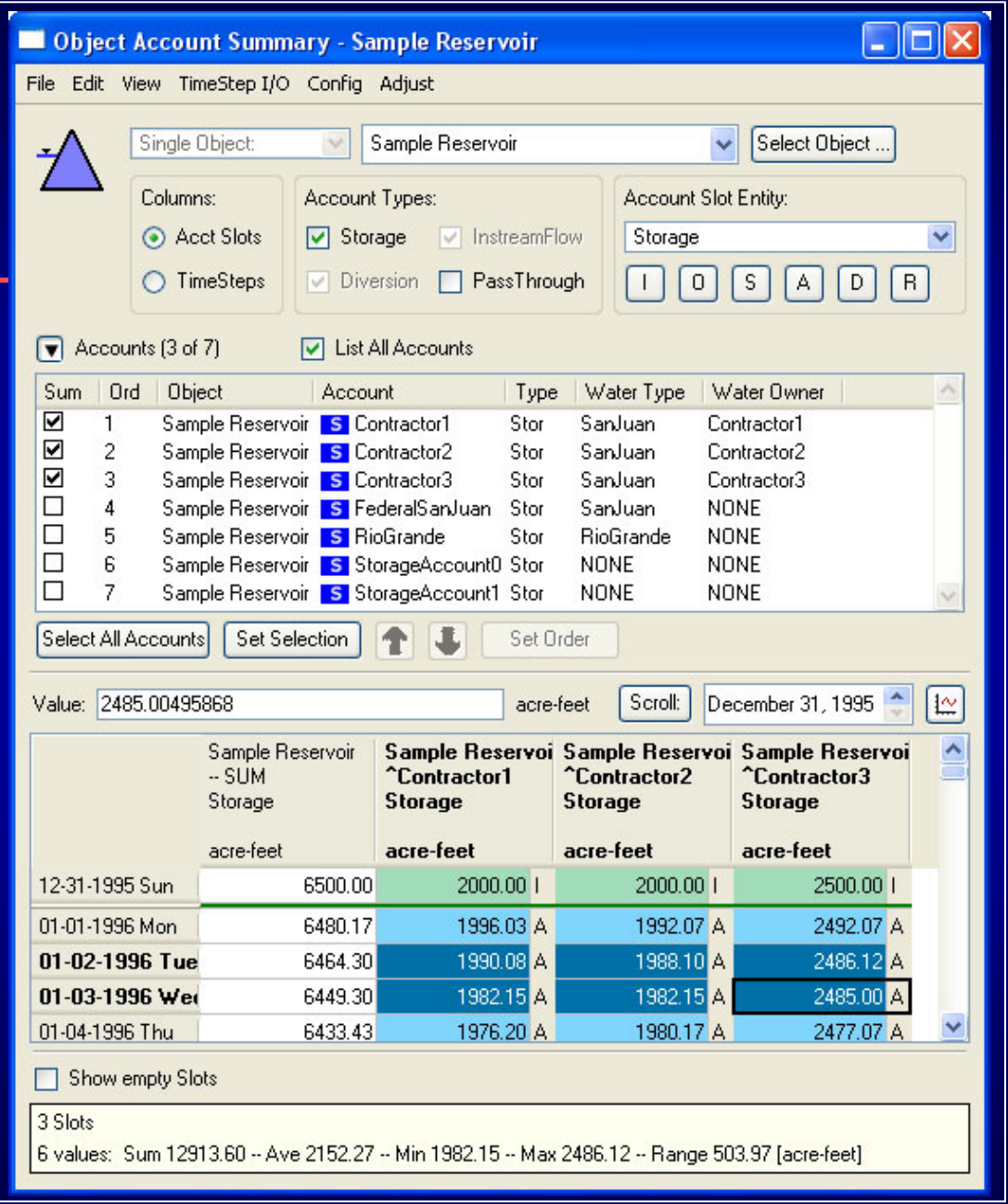


Presentation Outline

- Water Accounting Capabilities
  - Water Accounting Motivation and Definitions
  - Accounts
  - Supplies
  - Solution Algorithm
  - User Defined Accounting Methods
  - Exchanges
  - Demonstration as we go….
- Demonstration of Object Accounts Summary utility
- Accounting Class Outline
- Prioritized Water Rights Modeling and Allocation

Object Account Summary

- Ported to Qt
- Demo!



Presentation Outline

- Water Accounting Capabilities
  - Water Accounting Motivation and Definitions
  - Accounts
  - Supplies
  - Solution Algorithm
  - User Defined Accounting Methods
  - Exchanges
  - Demonstration as we go…
- Demonstration of Water Accounts Summary utility
- Accounting Class Outline
- Prioritized Water Rights Modeling and Allocation

Accounting Class Outline - Day 1

- Overview of accounting system and solution
- How accounting works
- Run and view an accounting model
  - Navigate accounting system using new accounting workspace (also through simulation workspace)
  - Become familiar with different elements of accounting system
  - Run, observe and analyze model output

Accounting Class Outline - Day 2

- Rules and Accounting – Strategies and features
- "After-the-fact" Accounting model
  - Use rules to specify accounting releases and diversions from simulated flows
- Advanced RBS: Accounting Driven Simulation
  - Rulebased Simulation dependence on Accounting System
- Water Rights Allocation
  - Allocation based on account water right priority date

Accounting Class Outline - Day 3

- Group/Interactive Exercise: Building an Accounting Model
  - Start with existing simulation model
  - Create accounting system according to specifications
  - Mix of white board session and work on the computer
  - Write rules to implement accounting policy

Presentation Outline

- Water Accounting Capabilities
  - Water Accounting Motivation and Definitions
  - Accounts
  - Supplies
  - Solution Algorithm
  - User Defined Accounting Methods
  - Exchanges
  - Demonstration as we go…
- Demonstration of Water Accounts Summary utility
- Accounting Class Outline
- Prioritized Water Rights Modeling and Allocation

Prioritized Water Rights Modeling and Allocation Outline

- Requirements
- Modeling Requests for Water
- Types of Water Rights Modeled
- General Description of Model
- Predefined Rule Function
- Solution Algorithm
- Status of Work
- Summary

Requirements

- Each right has a priority date
- No two accounts may have the same priority date
- A right's allowed quantity of water may be described in various ways on legal documents; the system must be flexible enough to model these ways of defining rights
- First in time, first in right
  - A junior right may not "short" a senior right

Requirements (cont'd)

- Three distinct types of rights must be accommodated:
  - The right to divert from a stream
  - The right to store water
  - The right to ensure minimum in-stream flow
- Some rights are subject to physical constraints that are defined in terms of the state of the network after senior rights have been satisfied
  - Physical capacity of diversion structure
  - Size of reservoir

Requirements (cont'd)

- Some rights are subject to legal constraints that are defined in terms of the state of the network after senior rights have been satisfied
  - Minimum bypass requirements, a form of subordination to other rights
  - Accrual-based maxima on the rights
- The effects of an upstream allocation on a downstream right must take into account lags and losses in the stream

Requirements (cont'd)

- Must improve performance over RPL-based implementations
- RiverWare must be able to replicate results of WAM (Texas WRAP model)

Modeling Requests for Water

- Each right makes an *Initial Request* based on the legal description of the right
- The state of the system at the time of allocation might restrict the initial request by
  - Physical constraints
  - Legal constraints

  to produce a net *Allocation Request*
- The difference between the *Initial Request* and the amount allocated is the *Shortage*

Types of Rights Modeled

- Diversion Rights
  - Right to divert from a stream (via a supply of type “diversion/return flow”)
  - Modeled with a diversion account
  - Initial allocation request may be user-input or disaggregated from an annual request
  - Subject to physical capacity of diversion structure
  - May be subject to other legal constraints

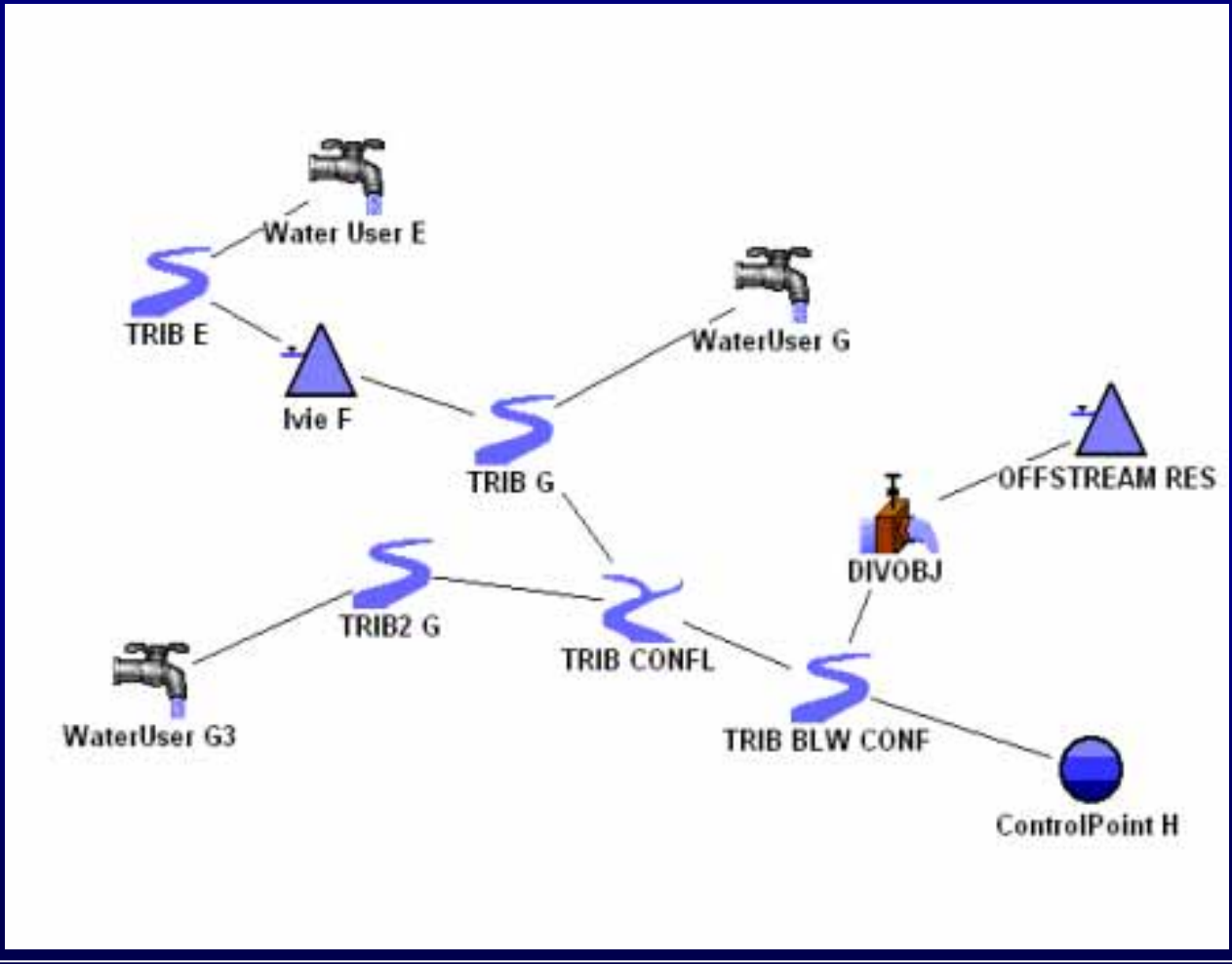
Types of Rights Modeled (cont'd)

- Storage Rights
  - Right to store water from a stream
  - Modeled with a storage account
    - On-stream: supplied by a passthrough account on a reservoir (via a supply of type "transfer")
    - Off-stream: supplied by a passthrough account on a diversion object (via a supply of type In/Out), which is in turn supplied by a passthrough account on a reach (supply type Div/Ret)
  - Subject to physical capacity of reservoir; off-stream rights also subject to capacity of diversion structure
  - Initial allocation request may be user-input or "fill conservation pool of reservoir"
  - May be subject to other legal constraints

Types of Rights Modeled (cont'd)

- Instream Flow Rights
  - Right to keep water in a stream
    - Irrespective of the "color" of that water (natural flow or project water)
    - Nevertheless, limited to the amount of natural flow that is in the stream after senior rights are satisfied
  - Modeled with an instream flow account on a control point object
  - No supply required; the supply is implicit
  - *Flow* slot shows total flow at control point
  - Initial allocation request may be user-input or based on a reference level and seasonal mapping of reference level to flow
    - Reference level is computed by the control point, based on sum of reservoir storages at end of a reference year

General Description of Model

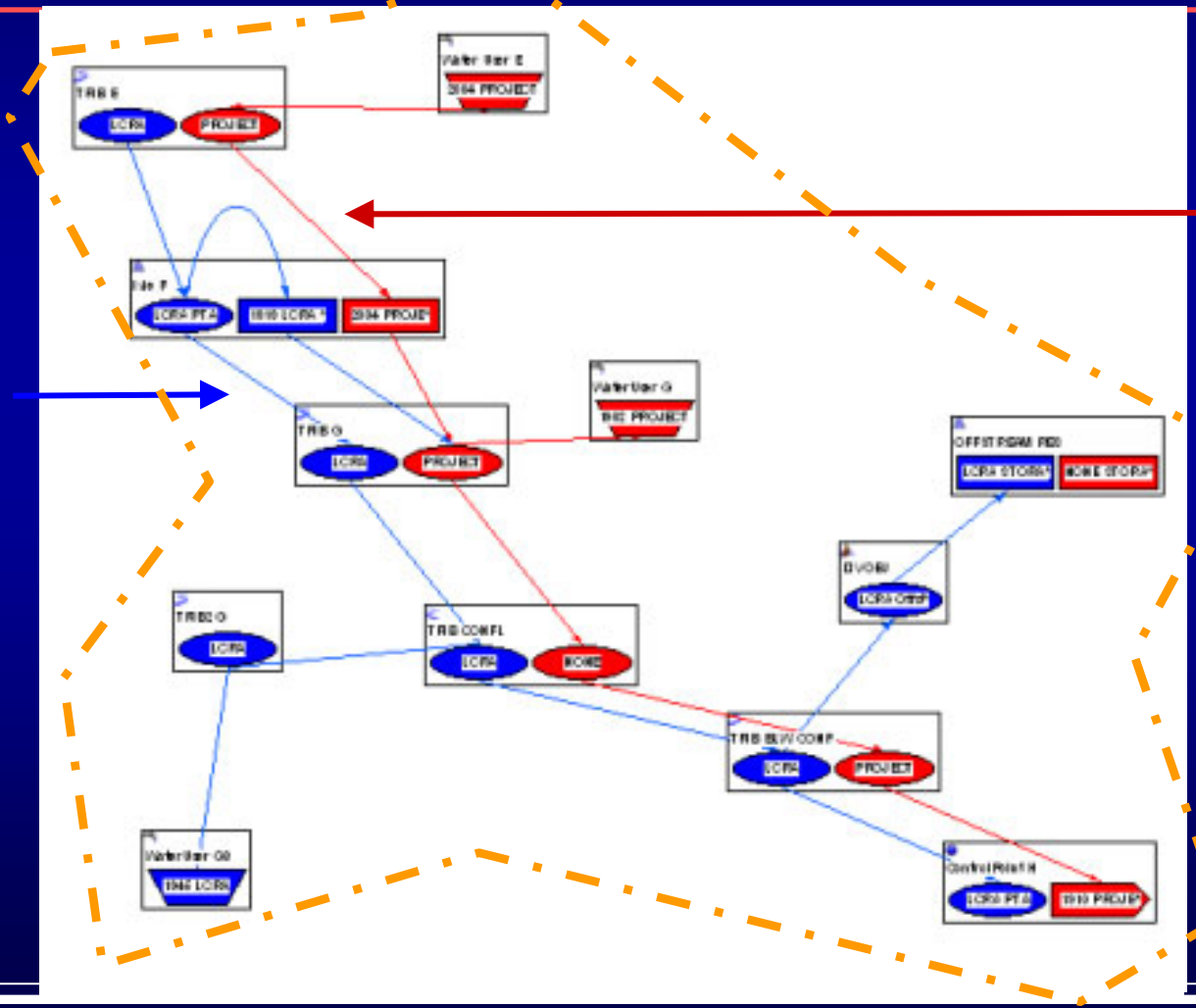


General Description of Model (cont'd)

- Natural-flow supply chain is identified by water type of the supplying accounts
- Objects containing the water-rights accounts are members of a Computational Subbasin for the purpose of computing appropriations
- Computational Subbasin clones the accounts; all computations for the solution are performed in clone-world

General Description of Model (cont'd)

Natural-Flow supply chain



Project-water supply chain

General Description of Model (cont'd)

- Rule-based model
- Accounting drives the simulation
- Requests for allocation are determined at beginning of run or at each timestep, depending on method selections on the accounts

General Description of Model (cont'd)

- At each timestep, a predefined rule function solves for the supplies that represent allocations from "natural flow" to prioritized rights
- Other rules implement policy decisions (e.g., releases from reservoirs to meet unmet allocation needs)
- Rules copy slot values (diversions, transfers, releases) from accounting world to physical world to drive the physical simulation

Predefined Rule Function

- SolveWaterRights is called in a rule:
  - SolveWaterRights(
    subbasin-name,
    water-type,
    range of dates for prioritized accounts)
  - Return value is a list of {slot name,value} pairs for:
    - Supplies that represent appropriations
    - Allocation Requests, Initial Requests
    - Current Maximum Call for instream flow accounts
  - Calling rule puts the returned values into their associated slots

Predefined Rule Function (cont'd)

- Current timesteps' values are copied to slots in cloned accounts
- Solver computes solution for current timestep
- Three passes:
  - Pass 1: clear all allocations
  - Pass 2: execute solution algorithm
  - Pass 3: transfer results as (slot, value) pairs to the calling rule

Solution Algorithm

- For each right in priority order:
- Compute its allocation request (taking into account state-of-system constraints)
- Propose the maximum allocation possible (make the allocation on the cloned account), up to the requested amount but not exceeding the right's shortage or the water available at point of allocation
- Re-solve the cloned accounting system
- Check for shortages at downstream seniors
  - If a shortage occurs make a "call": using that shortage, solve backwards for the amount we have to reduce the proposed allocation
  - Take into account loss (and lag, eventually) between the shortage point and the allocation point
  - If the proposed allocation is cut back, re-solve the cloned accounting system
- Continue with next right in priority

Solution Algorithm (cont'd)
Special Handling for Instream Flow Rights

- To model instream flow rights, we need to
  - establish the natural flows at control points at the time of allocation (*Current Maximum Call* slot values)
  - execute rules to operate the reservoirs, possibly releasing water to satisfy contracts with downstream diverters
  - allocate water from the natural flows but with instream flow rights now considering natural flows and project water

Solution Algorithm (cont'd)
Special Handling for Instream Flow Rights

- Variant of the rule function, SolveWaterRightsWithoutInstreamFlowCalls() treats instream flow accounts differently
  - When visiting instream flow accounts (in priority order), computes their allocation requests and sets the *Current Maximum Call* to lesser of (appropriation request, current flow)
  - Does not let instream flow rights place calls
- SolveWaterRights() allows instream flow accounts to make calls, treating *Current Maximum Call* as an upper bound

Status of Work

- Coded, in Beta test
  - New water account type - instream flow account
  - Dependent categories and slots on accounts
  - Methods and slots for accounts representing water rights
    - Priority dates
    - Requests, shortages - slots & methods on all legal water account types
    - Disaggregation of aggregated annual requests into daily/monthly requests for diversion rights
    - Minimum bypass constraints (legal) on diversion and storage rights
    - Reference level-based requests for instream flow rights
  - New solution methods on diversion accounts
    - Fractional return flow, variable fractional return flow

Status of Work (cont'd)

- Coded, in Beta test, cont'd
  - Methods and slots on physical objects to support water rights
    - Inflow needed to fill conservation pool, considering evap/precip
    - Capacity of physical diversion structure on reaches and reservoirs
  - Rule functions to allocate water
    - Cloning of accounts to provide side-effect-free computations
    - Topological analysis of subbasin to facilitate solution, performance
    - Variants to treat instream flow rights as required
- In Progress
  - On-line help
  - Continued testing

Status of Work

- Areas for Future Work
  - Lags
  - New controller (solves accounting first, followed by physical simulation)
    - Methods to move hydrologic inflow from simulation objects to accounts
    - Revisit reconciliation
  - Additional legal constraints (e.g., accrual-based)
  - Accrual methods to consider a subset of the supplies (e.g., natural flow only)
  - Offstream reservoir storage requests to include diversion demands on the reservoir at current timestep
  - New configuration possibilities for accounts on agg diversion sites
  - Improved diagnostics, detection of modeling errors
  - Additional performance analysis and improvements

Summary

- Models priority-based rights (“first in time, first in right”)
- Flexible mechanisms for modeling various legal entitlements and constraints
- Enforces physical constraints
- Simple calls to rule functions to solve for allocations
- Performance improvement realized
- Allows RiverWare model to replicate WAM
- Much is finished, more to do

Feedback

- Questions/Suggestions?