SCT enhancements to support TVA's Preschedule Editor functionality

Phil Weinstein, Tim Magee, Edie Zagona, CADSWES, 9-18-2014 (b) -- DRAFT

(0) Overview

This document describes RiverWare SCT enhancements to be developed to support TVA's Preschedule Editor -- previously implemented by TVA as an Excel application.

The TVA Preschedule Editor will be supported by the SCT's time-aggregated vertical timestep view (where slots are assigned to columns). Most new features will be implemented with new configuration options which expand the general usability of the SCT. However some Preschedule Editor SCT features will be too specific to be provided via general configuration options. Such configuration features would add unnecessary complexity for most SCT users. For this reason, the following special checkbox will be added to the SCT configuration to activate hard-coded provisions for the Preschedule application:

- [x] TVA Preschedule Mode

Features which are implemented as special "TVA Preschedule Mode" provisions are designated in this document with this symbol.

The following general enhancements will be developed to support the Preschedule Editor SCT:

1. Custom Time-Aggregation Summary Rows (only in the aggregated vertical timestep view)
2. Support for multiple Series Slots tabs
3. A second Series Data Table (on each Series Slots tab) to present non-scrolled slot columns (only in the vertical timestep views)
4. Automatic re-evaluations of RPL Expression series slots shown in the SCT at single timesteps.
5. Ability to swap different simulation objects into designated plot page curves based on the SCT's cell selection.

(0.1) Relevant SCT Preschedule Editor Details

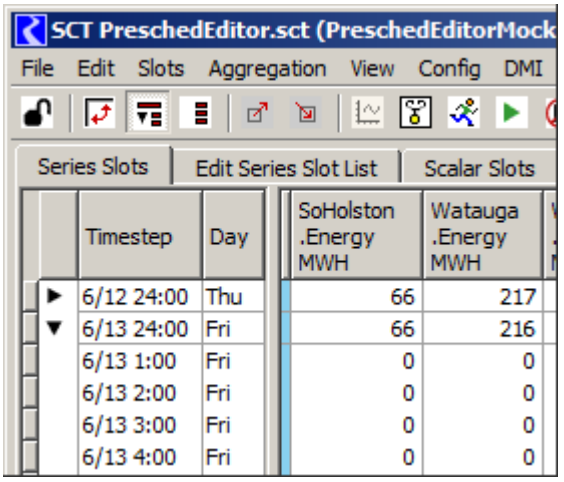
The TVA Preschedule editor presents Energy slots -- as columns -- for each reservoir in the system in an hourly timestep model. Each day's group of hourly timesteps (rows) are supported with additional rows for daily summary data. This daily summary data is either computed from the day's timesteps or data from other daily series slots or scalar slots.

A small number of days (of hourly series data) is supported -- currently three days, but the intention is to nicely support a week or so of data. (The new SCT features design doesn't impose an actual limit, but the formulation of features should exhibit optimal usability for this range of data).

(1) Feature Overview

(1.1) Application of current SCT features to the Preschedule SCT

SCT "Day Tree" Controls



Prior to this enhancement (e.g. in RiverWare 6.5) the SCT's Aggregated Vertical Timestep View -- when aggregating hourly timesteps into a daily aggregation -- shows a tree control (triangle arrow) on the summary row for each day of hourly data. When the tree control is open both the summary row *and* that day's 24 hour timestep rows are shown. When the tree control is closed, only the summary row is shown.

This functionality will be used within the Preschedule SCT for "navigation" to different days (among perhaps three to seven days of hourly data) by opening and closing each day's tree control. Probably these summary rows will be configured to show the daily *sums*. This will be redundant with a new "Total" custom summary row shown at the bottom of each aggregation (*see the next section*) -- if that total needs to be shown together with the other new summary rows.

A "TVA Preschedule Mode" customization for the legacy summary rows *hides* "24:00" in the row's Timestep column. In the accompanying image, the date/times shown to the right of the triangle arrows will be displayed as just "6/12" and "6/13" -- and not also "24:00".

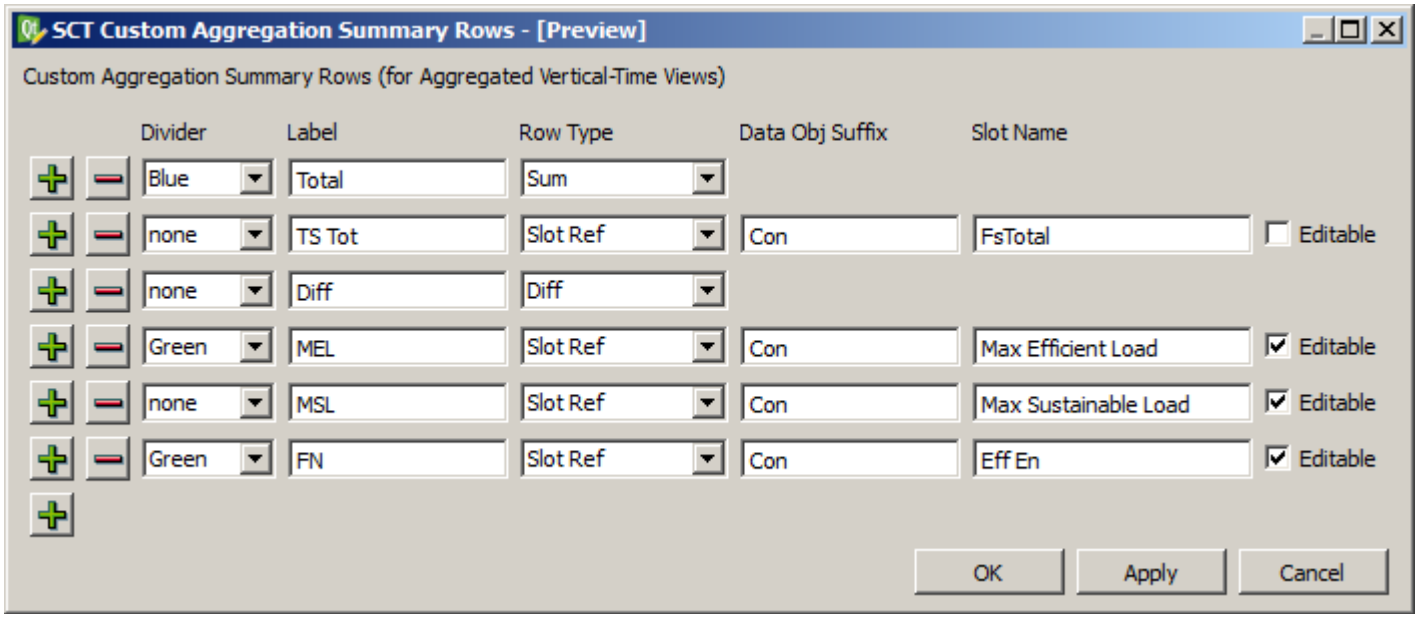
(1.2) Custom Time-Aggregation Summary Rows (only in the aggregated vertical timestep view)

This particular SCT View will support a variable number of new custom "Custom Aggregation Summary Rows" shown at the bottom of each time aggregation (in this case, for each day). Each such row can optionally be proceeded by a row divider with a selectable color. The following mockup image illustrates an example of six custom rows at the end of a one-day aggregation of hourly series data. (*Note: the initial column for aggregate tree controls is missing from this image*):

6/12 22:00	Thu	0	0	0	0	14	0	0	0
6/12 23:00	Thu	0	0	0	0	0	0	0	0
6/12 24:00	Thu	0	0	0	0	0	0	0	0
Total	Thu	0	13	0	0	28	0	148	256
FS Tot	Thu	7	0	0	0	14	19	109	256
Diff	Thu	0	0	0	0	0	0	32	158
MEL	Thu	7	0	0	0	14	19	109	256
MSL	Thu	0	0	0	0	0	0	32	158
FN	Thu	0	0	0	0	0	0	32	158

A new Custom Time-Aggregration Summary Row configuration dialog will be available when the aggregated vertical timestep view is current. This presents a variable number of rows of controls to define the set of these custom summary rows.

The following image is a mockup for this configuration editor dialog. See also field descriptions below.



Each row of controls contains at least these five columns. "Slot Reference" rows have additional controls in subsequent columns, *see below.*

1. A *plus*-icon button to insert a new row.
2. A *minus*-icon button to delete the row. This shows a confirmation dialog.

3. An optional preceding row divider type with a selectable color. This is presented as an option menu with these options:
    - NONE, Red, Orange, Yellow, Green, Cyan, Blue, Purple.

4. An editable label. This will generally be terse, about the length of a timestep date and hour specification (see image above). Preschedule SCT uses will probably be: "Total", "FS Tot", "Diff", "MEL", "MSL", "FN", "Sum6", "Sum12", "Sum18", "Sum24".

5. A Custom Summary Row Type column with an option menu for selecting among these different types:

    1. **Sum** -- of all timesteps in the time aggregation
    2. **Average** -- of all timesteps in the time aggregation
    3. **Maximum** -- of all timesteps in the time aggregation
    4. **Minimum** -- of all timesteps in the time aggregation
    5. **Difference** -- of the prior two rows.
    6. **Simulation Object Slot Reference.** This resolves to either a scalar slot or a series slot having a timestep size of the timestep aggregation. (In this example, this would be a daily timestep). The slot reference is composed from these pieces:
        - The name of the Simulation Object of the column's series slot.
        - An optional object name suffix to allow a DataObj associated with the Simulation Object to be used.
        - The local name of a slot.
    7. **Sub-range Sums.** Unlike the other custom row types, one instance of this item in the configuration creates multiple summary rows in the SCT: one row per time divider specified in the Vertical Time tab of the SCT configuration. Only time dividers for periods smaller than the time aggregation are considered. In a daily aggregation of hourly data, this applies to the following time divider options:

- Draw 4-Hour Divider Rows -- *this results in six Sub-Range Sum rows.*
- Draw 6-Hour Divider Rows -- *this results in four Sub-Range Sum rows.*

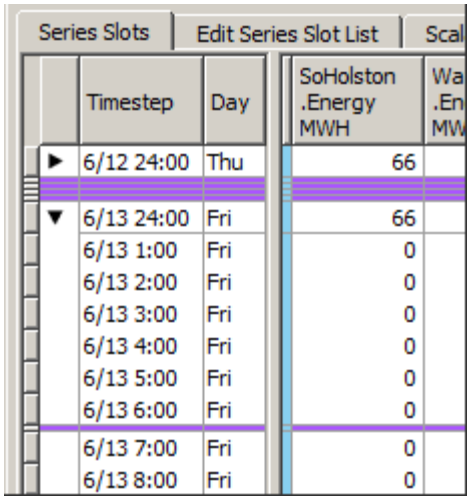
(note that only one of these two time divider types can be enabled at any given time).

Simulation Object Slot Reference items will contain the following additional control columns to identify a particular series slot (of the aggregation size ... in this case, daily) or scalar slot for each simulation object (reservoir) column. As described above, the primary reference simulation object will be the object containing the SCT column's associated series slot.

6. Optional Object Name Suffix (**"Data Obj Suffix"**) to specify a Data Object associated with the reference Simulation Object.
7. **Slot Name** -- the name of a slot on the reference Simulation Object or associated Data Object.
8. **"Editable"** checkbox to indicate if the resolved slot reference (daily series or scalar slot) is editable in the SCT.

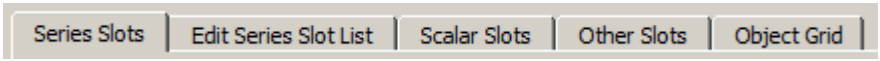
A "TVA Preschedule Mode" customization for the new Custom Summary Row Types *hides* a cell's value based on the particular summary row type and certain hard-coded slot names -- provisionally "Marginal Cost", "Load Forecast" and "System Total".

Normally, time dividers in the Aggregated Vertical Time view are shown even if the time aggregation in which they occur is closed. (*See the accompanying image showing 6-hour dividers*). But in "TVA Preschedule Mode", such time dividers will be hidden.



(1.3) Support for multiple Series Slots tabs

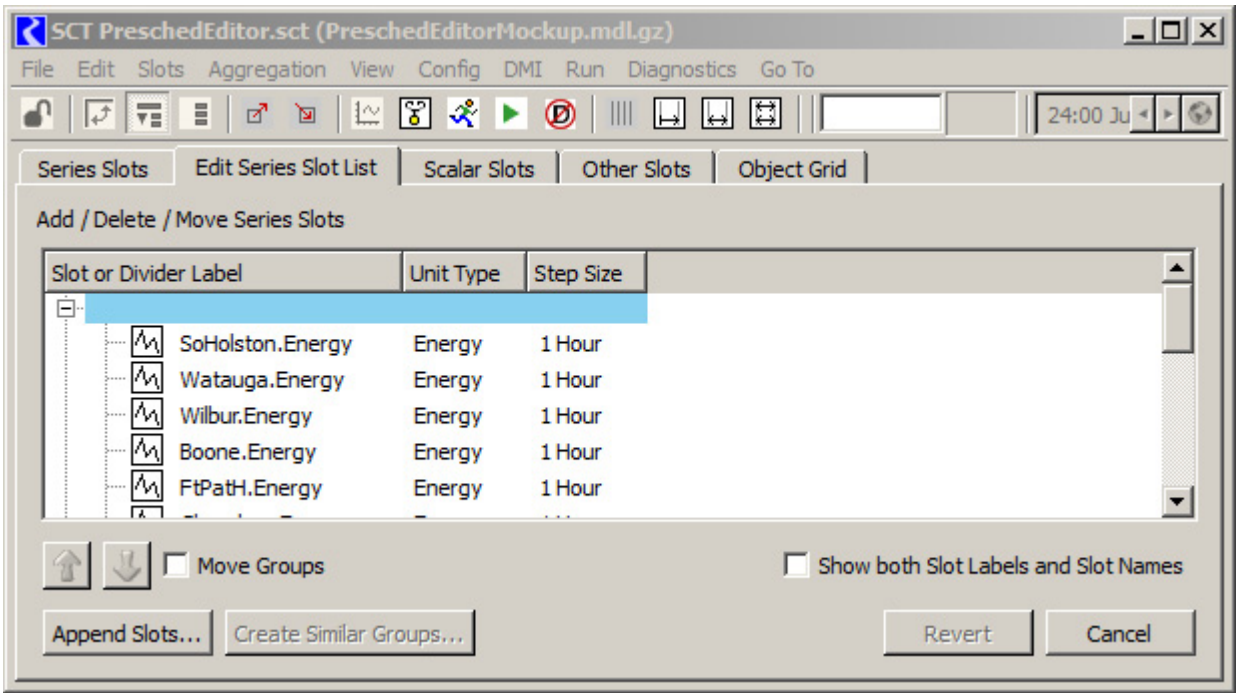
Prior to this enhancement (e.g. in RiverWare 6.5) the SCT supported these tabs:



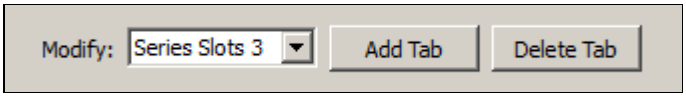
1. Series Slots
2. Edit Series Slot List
3. Scalar Slots
4. Other Slots
5. Object Grid

We will add support for multiple Series Slot tabs. These will just be numbered, starting at 1 (with labels "Series Slots 1", "Series Slots 2", ...). The Edit Series Slot List tab will be changed to **"Edit Series Slot Tabs"**.

Here is a screenshot of the RiverWare 6.5 SCT with the Edit Series Slot List tab shown:



In the change of this tab to **"Edit Series Slot Tabs"**, the following controls will be added above the "Add / Delete / Move Series Slots" text:



The **Add Tab** button appends a new Series Slots Tab and switches the option menu to that new tab.

The **Delete Tab** button shows a confirmation dialog box. Clicking "Delete Tab" in that dialog deletes the current Series Slots tab (indicated in the option menu) and renumbers the subsequent tabs.

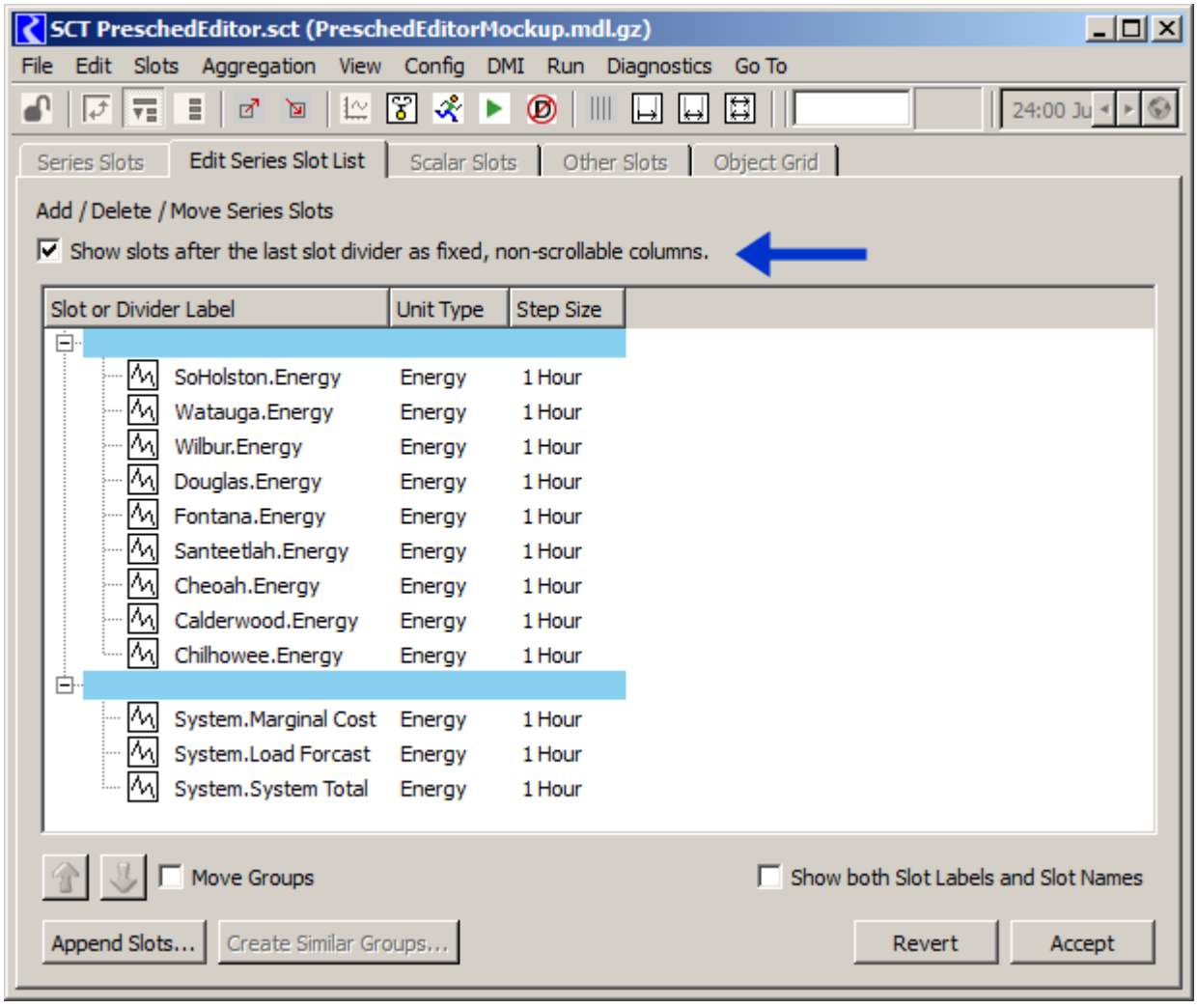
It will be easy to move slot list items from one Series Slots tab to another through the use of the currently supported Copy, Cut and Insert/Append slot item operations. (These can operate on a *multiple-item* selection). The user will be allowed to change the option menu to a different Series Slots tab even if changes have been

made to the slot list of the current tab.

The "Revert" and "Cancel/Accept" buttons will indicate whether changes to the slot list on any of the Slot List Tabs have been made. (If any changes have been made on any tabs, the Revert button is enabled and the "Cancel" button is changed to "Accept"). [Note: With these semantics, "Cancel" should instead be "Close"].

(1.4) A second Series Data Table (on each Series Slots tab) to present non-scrolled slot columns (only in the vertical timestep views)

In order to present a small number of fixed (non-horizontally-scrolled) slot columns, each Series Slot tab can support a 2nd Series Slot Data Table. Scrolling in the time dimension (vertically) is synchronized between those two series data tables and the row header table. This is supported only in the two vertical timestep views -- *aggregated* and *non-aggregated*. To support this feature, the indicated checkbox (*see below*) will be added to the "Edit Series Slot Tabs" tab. (*This is illustrated without the additional controls for multiple Series Slot tabs described in the prior section*).



(1.5) Automatic re-evaluations of RPL Expression series slots shown in the SCT at single timesteps.

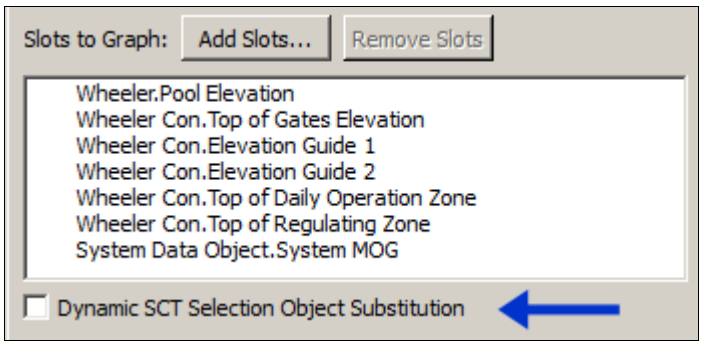
The SCT will support a new "Auto update expression slots on edit" option, presented as a new checkbox in the SCT's General Configuration tab. Operations within the SCT which result in changed slot values will cause RPL expression slots represented in the same SCT to be reevaluated at the effected timesteps.

Note: the alternative of triggering RPL expression slot reevaluation for any changes to the SCT's slots' values -- regardless of where those changes are initiated (i.e. also from outside of the SCT) -- was considered. There are some technical challenges to this approach, including being able to optimally limit the time range of the expression slot re-evaluations. Also, much attention would be required to preventing extra, unnecessary processing from external changes. Since the current requirement is for only data changes initiated from within the SCT, we will implement that more conservative approach.

(1.6) Ability to swap different simulation objects into designated plot page curves based on the SCT's cell selection.

Individual Plot Pages and slot curves within those pages will be designated as supporting "Dynamic SCT Selection Object Substitution". When the user makes a cell selection in the SCT (when that selection is limited to the slots of a single simulation object), the specially designated slot curves in open Plot Pages will show curves for the analogous slot on that reference simulation object -- and a heuristically identified companion Data Object. (This data object association is based on the name of a Data Object *starting with* the name of an existing simulation object -- excluding data objects).

The accompanying image shows the slot list of a RiverWare Plot Page, as viewed in the Plot Page configuration dialog. The indicated checkbox ("Dynamic SCT Object Selection Substitution") will be added to support this feature. When that checkbox is on, checkboxes will also be shown for each slot in the slot list. This indicates the slot curves to be changed when the cell selection changes within an SCT. (In this example, all but the last slot items would be checked).



(2) Feature Development Analysis

(2.1) Development of new Custom Time-Aggregation Summary Rows

See the description of this feature above in section (1.2).

The SCT's Aggregated Vertical Time view constructs a vector of row descriptors (RowTstepInfo records) given the SCT Configuration's time aggregation definition (SctAggregationDef) and the types of time dividers configured for vertical time views (e.g. the 6-hour dividers shown in examples above). [See method SctTableModel_TVertAgg::computeRowInfo()].

With the addition the new list of Custom Aggregation Summary Rows, the computed SCT row descriptor vector can also contain the new SctRowTypes defined in the following list. All of the new configuration items can result in at least one or two rows in the row descriptor vector: (1) an optional custom color divider row, and (2) a content row. A Sub-range Sums configuration item can result in multiple content rows, e.g. one for each of four 6-hour sub-ranges.

1. SCT_ROW_TYPE_DIV_TSTEP_CUST ... *custom time divider (with associated color)*
2. SCT_ROW_TYPE_DIFF ... *difference of prior two data rows*
3. SCT_ROW_TYPE_AGG_SUM
4. SCT_ROW_TYPE_AGG_AVE
5. SCT_ROW_TYPE_AGG_MAX
6. SCT_ROW_TYPE_AGG_MIN
7. SCT_ROW_TYPE_AGG_SLOT_REF
8. SCT_ROW_TYPE_AGG_SUBRANGE_SUM

Most of the new SctRowTypes show data which can be computed from slots available in the SCT. This computation will refer to the time-range of the particular containing time aggregation.

Development steps for this feature include:

1. Configuration Data Model, with persistence within an SctConfig, for a list of user-defined Custom Time-Aggregation Summary Rows.
2. Custom Summary Row editor. (See mockup screenshot in section 1.2).
3. Enhance computation of RowTstepInfo vector in SctTableModel_TVertAgg to include new custom summary rows.
4. Cell content computations for basic lookup summary types (Sum, Ave, Min, Max and Div).
    1. Basic computed aggregate summaries (Sum, Ave, Min, Max) -- done on the fly in user-units.
    2. Diff summary row (not technically dependent on aggregation).
    3. Sum-range Sums (based on configured time dividers).
5. Simulation Object Slot Reference rows.
    1. Data source slot reference computation -- various parts of the full slot name come from different pieces, including the cell's column.
    2. Maintenance of callbacks (for value changes and deletions) of referenced slots (distinct for each column).
    3. Enhance cell/timestep mapping algorithm for these rows.
    4. Maintenance of "editability" of these cells, including SctTableDelegate provisions.
    5. Upon editor completion, assignment of values to referenced slot (series slot or scalar slot).

(2.2) Development of support for multiple Series Slot tabs

Prior to this enhancement (e.g. in RiverWare 6.5), the SCT implementation works off of a single list of slot names (at the SctConfig level) and a single list of resolved slot pointers (at the SctModelData level). Clients of these modules (e.g. the various SctView instances and their contained SctTableModels and SctTableViews) will retain an index which identifies a particular Series Slot tab -- that index will, for the most part, be transparent to those clients.

Within SctConfig and SctModelData classes, the slot lists and other state information (e.g. the axis orientation and use of time aggregation) will be moved into new encapsulations for individual series slot tabs. SctConfig Serialization, however, will still maintain a single list of slot items, delimited with special markers. (The main consideration here is that SctConfig serialization uses a Flex/Bison-implemented grammar which should be modified as little as possible).

As described above in section (1.3), SCT configuration editing to support multiple Series Slot tabs is supported primarily by enhancements to the "Edit Series Slot Tabs" tab.

Development steps for this feature include:

1. SctConfig and SctModelData enhancements to support distinct Series Slot tabs.
2. Edit Series Slot Tabs operations:
    1. Add Tab and Delete Tab
    2. Switch to edit other Series Slot tab (option menu)
    3. Direction of the EditSctSlotListTreeView to the appropriate tab, for tree construction and user edits.
3. Redirection of SctDialog operations which operate on Series Slot presentation *and* data to the appropriate tab.

(2.3) Development of a second Series Data Table (on each Series Slots tab) to present non-scrolled columns (only in the vertical timestep views).

The concept of "slot index" is fundamental to much of the SCT implementation. A single vector of slot items will still be used across both SctTableViews and associated abstract item models and delegates. This will require a "base slot index" for that table implementation, but will minimize the impact on most of the SCT application code, e.g. the cell selection representation. An example of this is the minimal impact this enhancement has on the Edit Series Slot List (or "Tabs") tab described in section (1.4) -- the only change there is the addition of a new checkbox which turns out to be completely transparent to the operation of that panel.

There are two general areas of development required for this feature:

1. Interactions of the SCT application code with the SctTableView (and model and delegate) need to be directed to the correct instance.
2. SCT geometry maintenance needs to take into account three tables -- the Row Header table and the two Series Data tables.

The fact that the width of the second Series Data table will depend only the columns within that table helps (i.e. since the idea is that the second data table is intended for "fixed" columns). The current horizontal splitter between the Row Header Table and the Series Data Table will remain as it is -- with just a single splitter divider control; the new data table will be together with the first one with respect to this splitter.

Development steps for this feature include:

1. Add "show fixed slot column table" flag to SCT config (relative to a particular Series Data Table if we DO implement multiple), including serialization, and reflect that state in the new Edit Series Slot List (or "Tabs") tab.
2. Mechanism to provide a distinct slot index range for each of the two data tables. This involves both SctConfig and SctModelData provisions.
3. Conditionally deploy TWO data tables in an SCT View (for only the vertical timestep views); including:
    1. Confirm correct slot index mappings.
    2. Synchronize vertical (time) scrolling between all three tables.
4. Enhance SCT application interactions with the SctTableView (e.g. in selection processing) to support two table instances.
5. Horizontal geometry management issues involving the 2nd data table.

(2.4) Development of automatic re-evaluations of RPL Expression series slots shown in the SCT at single timesteps.

As mentioned in section (1.5), we are choosing to trigger RPL Expression slot re-evaluations only for data changes originated in the SCT.

With one minor exception*, all relevant value-setting operations operate on the current slot/timestep cell selection. Note that some *display* cells correspond to a set of contiguous timesteps on a single slot. It is latter type of "cells" which are significant here. (This cell selection is represented with the SctSlotTstepSet class which has efficient representations of "all timesteps" of a slot, and also "all slots" at a given timestep). At the point of initiation of value-change operations, the current slot/timestep cell selection can be examined to determine at which timesteps the expression series slots in the SCT need to be recomputed. If that cell selection contains an "all timesteps" indication (for any particular series slot, excluding expression series slots), then the SCT will initiate a full recomputation of the relevant expression slots (across all timesteps) instead of using the new algorithm to recompute expression slots only at specific timesteps.

Recomputation of the SCT's expression series slots will be initiated from the following methods:

1. SctDialog::setValueOnSlotTstepSet (SctSlotTstepSet* stSet, double val)
2. SctDialog::clearOutputsOnSlotTstepSet (SctSlotTstepSet* stSet)
3. SctDialog::interpolateSelection (bool testOnly)
4. SctDialog::adjustValuesOnSlotTstepSet (SctSlotTstepSet* stSet, ..)
5. SctManager::pasteCopyCellSet (SctDialog* dlg, SctSlotTstepSet* sel, bool asInput)
6. SctView::clipboardImportDlg_Paste (ClipboardImportDlg* dlg)

*The SctView::clipboardImportDlg_Paste() method can optionally set values at timesteps beyond the current cell selection.

As these are high level "hooks" for user-initiated operations, it will not be necessary to defer the actual expression series slot recomputation (i.e. using a QTimer, to condense multiple "trigger" events into a single recomputation event). This recomputation will be implemented with the following new method:

- SctDialog::recomputeRplExpressionSeriesSlots (SctSlotTstepSet* stSet=NULL).
  ... if stSet is NULL, the recomputation occurs at all timesteps within the relevant expression series slots.

RPL Expression Slot Manager initiates the recomputation of expression slots in the model. This is currently done using this method:

- RplExprSlotMgr::evaluateSlots (EvalTime now) const

where EvalTime is one of the following values:

1. ALWAYS
2. ON_DEMAND
3. BEGIN_RUN
4. END_RUN
5. BEGIN_BLOCK
6. END_BLOCK
7. NEVER

The "now" EvalTime is reconciled with each expression slot's EvalTime to determine if the slot should be evaluated. If "now" is ON_DEMAND, then the evaluation occurs regardless of the slot's EvalTime (with the exception of slots having the NEVER EvalTime). The RplExprSlotMgr::evaluateSlots() method iteratively reevaluates all expression slots needing to be evaluated (depending on the "now" parameter value) until all mutual dependencies have been satisfied.

Without significantly impacting on the run-time performance of this method -- and with care to insure that its behavior in its current uses are not changed -- *we need a way of providing the set of expression slots to be evaluated.*

The RplExprSlotMgr::evaluateSlots() method calls this SeriesSlot method to evaluate an expression series slot:

- SeriesSlot::evaluateExpr (bool evalOnly)

This method recomputes values at timesteps based on the following criteria. If active within a "timestep controller" run, evaluation occurs only at the run controller's current timestep. Otherwise, evaluation occurs at the expression slot's full configured time range.

Again, without significantly impacting on the run-time performance of this method -- and with care to insure that its behavior in its current uses are not changed -- we need ways of specifying the following:

- A single specific timestep (or, alternatively, a range of timesteps) at which to evaluate the expression slot, OR
- *that* the expression slot should be re-evaluated at its full configured time range -- regardless of any run controller state*. (This latter requirement could be satisfied by allowing a range to timesteps to be specified, as suggested as an alternative to the first requirement).

*It would be unusual (and likely, problematic) for the user to apply edits to series data in an SCT during a run, but that is not actually prevented.

Development steps for this feature include:

1. SCT: Add configuration option: [x] Auto update expression slots on edit. (a) Sct configuration data model and flex/bison persistence, (b) Checkbox in the "General" SCT Configuration tab.
2. SCT: Implement recomputeRplExpressionSeriesSlots() method to evaluate the SCT's expression series

slots at the appropriate timesteps.
3. SCT: Call that method from the six identified high-level value edit methods. In one special case, this involves constructing a SctSlotTstepSet instance to represent the time range of the modified series slot timesteps.
4. RplExprSlotMgr::evaluateSlots() modifications: devise an efficient way of *optionally* specifying: (a) a set of expression slots to be evaluated, and (b) a timestep range at which those slots should be evaluated.
5. SeriesSlot::evaluateExpr() modifications: devise an efficient way of *optionally* specifying a range at which the slot should be evaluated.

(2.5) Development of the ability to swap different simulation objects into designated plot page curves based on the SCT's cell selection.

The major two functions for this feature are:

1. Routing of notifications from SCTs to open Plot Dialogs of "Object Context" changes based on an SCT's cell selection.
2. Substitutions of the SimObj component of plot curve slot references, according to the Object Context notification, for designated curves in designated plots. (See details in section 1.6).

"Object Context" notifications will be implemented as RiverWare callbacks routed through the Output Device Manager (Sim/cwOutputDeviceMgr). Curve slot reference substitutions will take place only if the correct "live" slot is found on the new object, or as appropriate, on a related data object.

Management of plot curve state information is generally tricky because the "current copy" of that information lives in two different sorts of places at different times. When a Plot Page is active in a dialog, the curves' state information is represented within a QwtPlotCurve object -- i.e. of the Qwt library. But since use of the "Dynamic SCT Selection Object Substitution" feature is fundamentally *ephemeral,* it may not be important to get those substitutions back into the corresponding RiverWare data structures if that isn't easy to do.

A special algorithm is required for support of slots on data objects, given their implicit relationship to other non-data object simulation objects (e.g. reservoirs). A slot curve which is indicated as supporting this feature which is on a data object will be processed in this way:

1. We search for a non-data object SimObj whose name (in its entirety) matches the first part of the data object's name.
2. The remaining part of the data object's name is used.
3. If there exists another data object in the workspace whose name is the new "Object Context" object PLUS the remaining part of the original data object's name, then the substitution continues with that identified data object.
4. If that data object also contains a slot with the local name of the original curve's slot, then the substitution is applied.

(That last step applies also to normal non-data object SimObj subtitutions).

Development steps for this feature include:

1. Enhancement of the Plot Page data model to indicate that this feature is active for a Plot Page, and which curves within the various (up to 9) plots contained in that Plot Page.
2. Enhancement of Plot Page (Output Device) configuration dialog with the new checkbox and conditionally

shown checkbox on the individual slot items (QTreeWidgetItems). Note that this configuration dialog is an application of the SlotSetOutputConfigDlg class.

3. Generation of object context notifications from the SCT to Plot Pages via Sim/cwOutputDeviceMgr. We'll define a public method on that manager class which will be called from the SCT. That method will create and issue a callback which will be processed by Q3GUI/PlotDialog instances.
4. Make substitutions in the live curves, including the algorithm outlined above for slots on data objects; and force a redraw.

--- (end) ---