

RiverWare Series Display Compression

High Level Design Options

Author: Phil Weinstein, CADSWES
 Edit Date: 9-15-2007

The one requirement for Series Display Compression which influences the options outlined here is **the ability to apply Series Display Compression (in particular, Repeated Value compression) to an arbitrarily composition of multiple series.** [[See use case 3b](#)].

The current method of composing multiple series into a single display is the RiverWare **SCT**. While this nicely supports multiple series composition, there are some disadvantage of use of the SCT for anticipated Series Display Compression uses -- in particular:

- The SCT does not support Accounting SeriesSlot Flags (with cell background colors).
- The SCT's support for SeriesSlot Flags uses only background color. The other primary RiverWare series display -- the Open SeriesSlot dialog and related Accounting Series applications of the SlotQtDlg class -- support a Text Character flag (as well as Accounting SeriesSlot colors).
- The SCT is not optimal for the display of a single series, or a small set of series. This case is represented by [three of the four use cases](#) provided by Shane Coors.

With these considerations in mind, we have identified these four high level development options:

1. **SCT-only** (vertical timestep / non-aggregated view only), enhanced to support Accounting SeriesSlot Flags (background colors plus optional Text Character flags).
2. **SlotQtDlg applications only** / with development of **Named, Persistent SlotColRefLists**.
3. **SlotQtDlg applications only** / with development of **ListSlots of SlotColRefs**.
4. **SlotQtDlg** (as is) AND the **SCT** (vertical timestep / non-aggregated view only) (as is).

These options are discussed in the following sections. **My current recommendation is Option 4.** [Phil, 9-15-2007].

(1) SCT-only (vertical timestep / non-aggregated view only), enhanced to support Accounting SeriesSlot Flags (background colors plus optional Text Character flags).

[[See the use case screenshot using the SCT](#)]. With enhancements to the SCT's series flag ornamentation, the SCT could be used for all needs for the Series Display Compression feature.

From any [Open SeriesSlot](#) or related Accounting Series display ([Edit Account](#), [Object Account Summary](#) or [Exchange Balance](#) dialogs), the user can show the displayed series in a New SCT or single Open SCT with operations under the File menu. [Note: the linked screenshots show the prototype Display Compression Control Panel, but those would not be added to those dialogs under this development option. Rather, that Panel would be shown only within the SCT].

This option has the additional development requirements and disadvantage summarized in the first section, above. Adding support for Accounting Flags is, unfortunately, somewhat complex because there are persistent custom-color configurations associated with each flag. This involves SCT configuration GUI dialogs and SctConfig class support, including SCT File persistence using Flex/Bison parsing. (So, Sct Configuration configuration mechanisms would need to be enhanced, albeit only with additional items supported in the same way as existing items).

The user interface disadvantage to this approach is the awkwardness of using an SCT for only a single, or low number of series. The SCT requires a wide window display -- a sub-optimal use of screen real estate. Note that

adding additional series to the SCT would interfere with the display compression because all columns must conform to the compression algorithm (assuming the [current design](#)).

(2) SlotQtDlg applications only / with development of Named, Persistent SlotColRefLists.

If the [SlotQtDlg / SlotQDlgTable applications -- including applications which display related series](#) -- were to be enhanced to allow the user to pick an arbitrary set of SeriesSlots (and SeriesSlot Columns and TableSeriesSlot Columns) -- then all requirements could be met by implementing Series Display Compression only there.

This approach has been documented as the "[RiverWare Multiple SeriesSlot Viewer](#)" (see the linked document).

That document briefly mentions support for a global set of SlotColRefLists maintained as named, persistent objects within the RiverWare Model (and model file). The SlotQtDlg and SlotQDlgTable classes would be enhanced with a fifth "application" -- the RiverWare Multiple SeriesSlot Viewer -- which would configure the various GUI columns from elements within a SlotColRefList.

This approach entails a **significant development effort** (both GUI and data model) in the maintenance of named SlotColRefLists, and a non-trivial enhancement to SlotQDlgTable.

A more fundamental disadvantage is the difficult-to-justify redundancy of **having two distinct ways** of configuring and displaying an arbitrary composition of series (i.e. this new viewer *and* the SCT).

(3) SlotQtDlg applications only / with development of ListSlots of SlotColRefs.

A variation of the previous development option is organizing lists of SlotColRefs into ListSlots -- i.e. ListSlots of SlotColRefs. [We currently support ListSlots of SimObjs and ListSlots of Accounts].

This has the advantage of leveraging the maintenance of sets of things (Slots) on SimObjs. In particular, the user would create SlotColRef ListSlots on Data Objects. The SlotQtDlg / SlotQDlgTable classes would be enhanced to support the display of SlotColRef ListSlots -- with the result being very similar to the draft design of the [RiverWare Multiple SeriesSlot Viewer](#).

This is potentially more coherent than the previous option. However, a broader development effort would be required to insure that SlotColRef ListSlots were supported reasonably in all mechanisms which handle Slots in general. A good amount of analysis and development would be warranted to insure that the implementation was sufficiently complete.

Perhaps there will be future requirements for which SlotColRef ListSlots would be great. But this may be more than what we want to take on in the context of the Series Display Compression development project.

(4) SlotQtDlg (as is) AND the SCT (vertical timestep / non-aggregated view only) (as is).

I recommend that we just implement Series Display Compression in both areas, with shared support provisions (e.g. a common Display Compression Control Panel) [Phil, 9-15-2007] ...

Through the use of actual shared components (e.g. a [Display Compression Control Panel](#)) and analogous design approaches (e.g. "static" -- if not actual -- polymorphism), it really shouldn't be too difficult to implement Series Display Compression in both the SCT and the SlotQtDlg / SlotQDlgTable applications. Series Display Compression is a natural feature for both of these series displays.

Of the four SCT view modes, we would implement Series Display Compression in only the first:

- Vertical Timestep / Non-Aggregated View -- *SUPPORTS compression*
- Vertical Timestep / Aggregated View -- *no compression supported*

- Horizontal Timestep / Non-Aggregated View -- *no compression supported*
- Horizontal Timestep / Aggregated View -- *no compression supported*

The implementation of the four SCT view modes are well encapsulated in distinct SctView subclasses.

--- (end) ---