RiverWare Run Analysis Dialog Qt Port Development Analysis

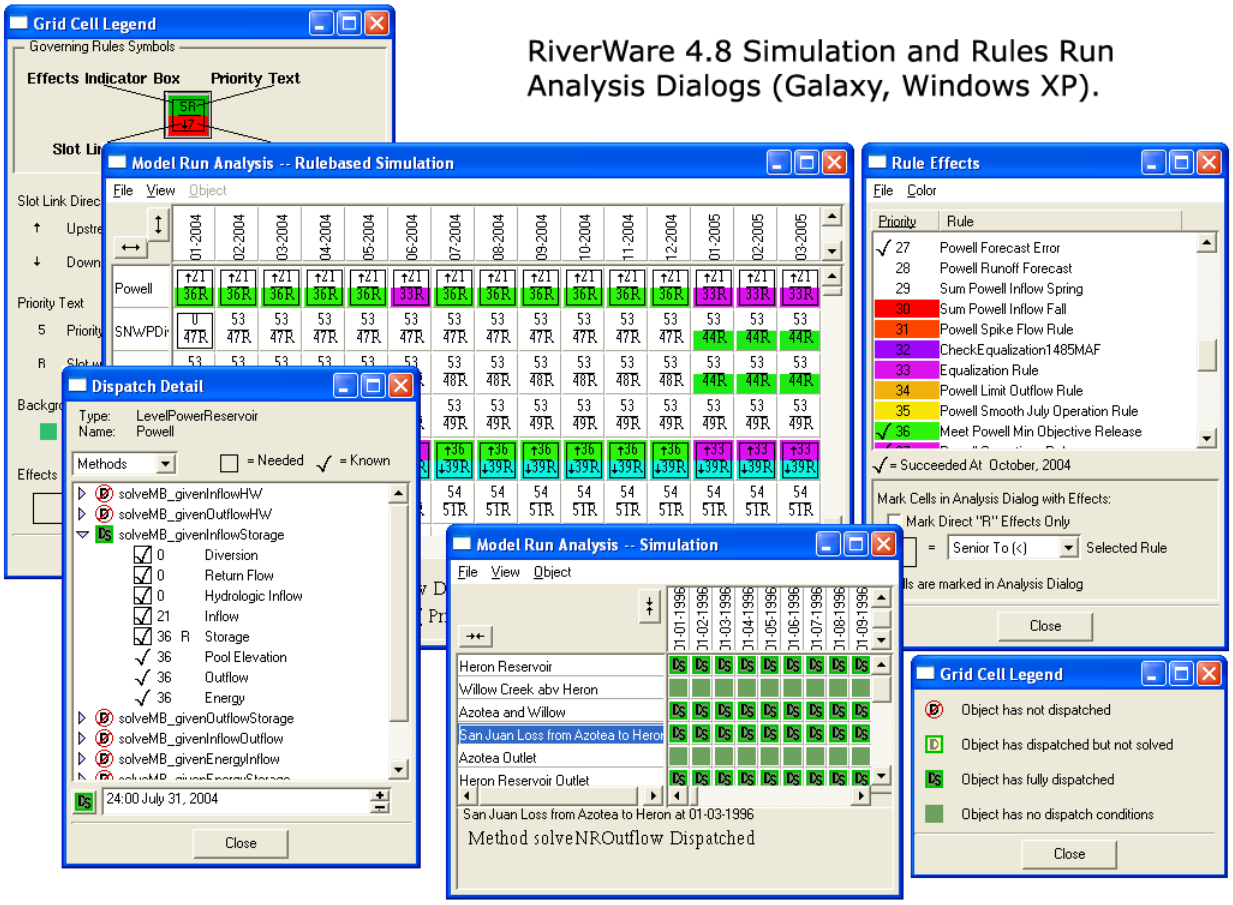
Author: Phil Weinstein

This document outlines the GUI design and development issues related to porting the Galaxy-implemented Simulation and Rules Run Analysis dialogs to Qt.

Not covered in this document is the analysis for the porting of the Optimization Run Analysis dialogs.

Document Status

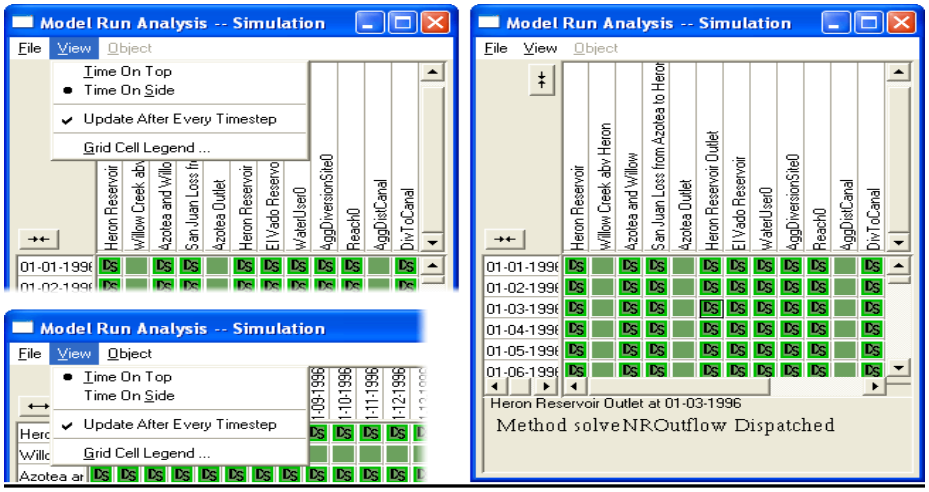
- [11-28/29-2006]: Initial writing; Alternative / Intermediate Dev. Options for shorter development time.
- [1-20-2007]: Development Task Progress updated. See last two pages.



1.0 Contents

1.1 Related Documents

- Design for Dispatch Information Interface, Paddy McCarthy, January 21, 1998
- Dispatch Information for Rulebased Simulation / GUI Design, Paddy McCarthy, June 29, 1998
- Functionality for Dispatch Information, Paddy McCarthy, January 29, 1997
- Dispatch Information for Rulebased Simulation / Functional Requirements, Paddy McCarthy, July 2, 1998



View menu: *Time On Side* vs. *Time On Top* — *Time On Side mode (non-default)*

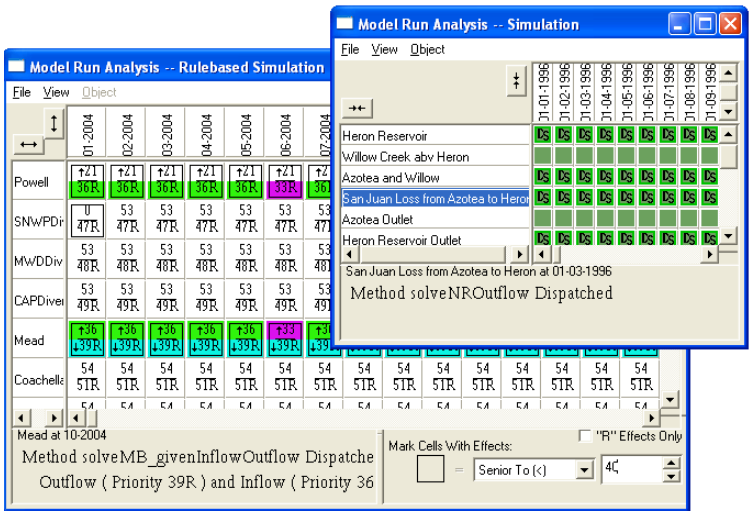
2.0 Dialog Boxes / Development Analysis

The Qt port of the Simulation and Rules Run Analysis dialogs will provide the functionality of the older Galaxy dialogs. For the most part, enhancements to be introduced are those which are inherently or readily provided by the Qt widgets, e.g. context menus, where appropriate. Also, the following enhancements will be implemented:

1. Add a **TimeStep Date/Time Spinner for Timestep navigation** to the **Model Run Analysis** dialog. When the **Rule Effects** dialog is displayed stand-alone (as opposed to being shown as an integrated panel in the Model Run Analysis dialog), it also will display its own Date/Time spinner. The existing (Galaxy) Dispatch Detail dialog already has a TimeStep Date/Time spinner, and that will be true also of the Qt implementation of that dialog.

2.1 Model Run Analysis Dialog

The two Model Run Analysis dialog displays addressed in this port -- i.e. for Simulation and Rules -- share the same basic structure and will be implemented as distinct display modes of a single Model Run Analysis dialog in order to allow switching between the two modes within a single instance of the dialog (i.e. in the case of a Rules model).



The two display modes have similar menubar operations, and the following panels:

1. An **Object/TimeStep Dispatch Info Table** with simulation timesteps along one dimension (generally, columns), and Simulation Objects (SimObj) along the other dimension (generally, rows). The table cells provide a terse summary of whether, and how, a SimObj **dispatched** and **solved** at each timestep. The information shown in this table is not editable by the user.

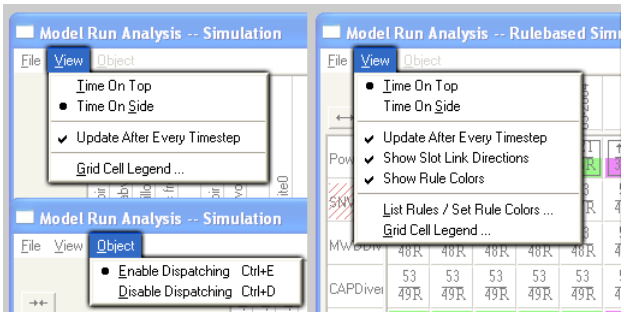
   Two toggle push buttons, for independently **expanding or condensing the row and column headers** are displayed in the upper left corner of the table (above, and to the left of the row and column headers).

One or two panels below the table:

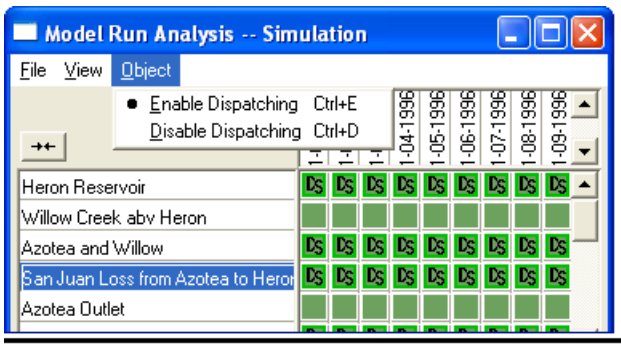
2. A **Selection Status Panel** providing a textual description of the Dispatch Info for the selected cell in the Object/Timestep Dispatch Info Table.
3. For the Rules Model Run Analysis dialog only, an **Effects Ornamentation Panel** to define the properties of Object/TimeStep Dispatch states for which cells in the Table are highlighted (with an extra cell border).

The **File Menu** contains only the Close Window (Ctrl+W) operation.

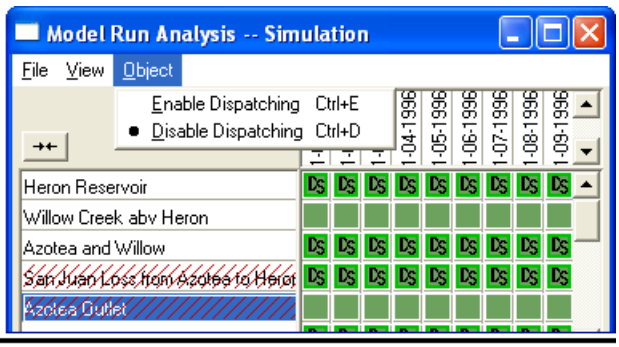
The **View Menu** contains operations to choose the axis orientation (Time on Top vs. Time on Side), a toggle to indicate whether or not the table should be updated after every model run timestep, dispatch info cell drawing options, and operations to show auxiliary Dispatch Info dialog boxes. See image to the right.



The **Object Menu** is active only if a single SimObj is indicated within the table selection. It currently contains only controls to disable or enable the dispatching of the indicated SimObj. Note that SimObjs for which **dispatching is disabled** are indicated with a **red crosshatch over the object name.** See the following screenshot:



Object menu: Enable / Disable Dispatching



Disable Dispatching crosshatch (red) on two SimObjs

The upper-left corner may be able to handle one or two GUI controls in addition to the **expand / condense header controls.** A good candidate would be some sort of **toggle to switch between Simulation and Rules views** in Rules models. Recent prior versions of RiverWare has an invisible control for this purpose, and may have been eliminated because of some confusion about what it was for.

2.1.1 Object / TimeStep Dispatch Info Table Architecture

The **table cells** will be presented within a subclass of QTable (or actually, of our hacked copy of QTable -- **rwQTable** -- to overcome some implementation limitations with Trolltech's QTable class). QTable virtual methods will be redefined to prevent instantiation of per-cell C++ objects. The table will support custom cell drawing. In the case of Simulation mode, drawing will be limited to the painting of one of a small number of predrawn icons. In Rules mode, the individual symbols and digits will be drawn using low-level drawing primitives, similar to the Galaxy implementation.

In order to make use of the top-left rectangular area between the row and column headers, at least one of the headers need to be implemented as distinct synchronized-scroll rwQTable widgets, rather than as the central table's horizontal or vertical QHeader.

2.1.2 Object / TimeStep Dispatch Info Table - Column Headers

Recommendation: The **Column Header** should be implemented as a **distinct synchronized-scroll rwQTable.** This is necessary for the drawing of rotated header text. Some specific properties and issues include:

- Non-adjustable column width. Column width will be fixed based on the horizontal space needed for the dispatch info cell images.
- No horizontal scroll bar is shown. Horizontal scrolling is synchronized with the horizontal scroll position of the main table.
- **Change Possibility (TO BE CONSIDERED):** Rotate text 90 degrees clockwise instead of counter-clockwise. This is a more standard representation, and aligns the beginning of the text label with the top of the drawing area, instead of with the bottom.

2.1.3 Object / TimeStep Dispatch Info Table - Row Headers

Alternative A: Use the central **rwQTable's vertical QHeader.** The user would be able to re-order SimObj rows by dragging row labels. So, development to support user-controlled ordering would be minimal. However, since we can't readily provide custom drawing in QHeader cells, the "dispatch disabled" (red-crosshatch) ornamentation would either have to be done elsewhere (e.g. on the central table cells), or we would have to implement that feature in rwQHeader, which currently has only very minor, absolutely essential, changes from the QHeader class.

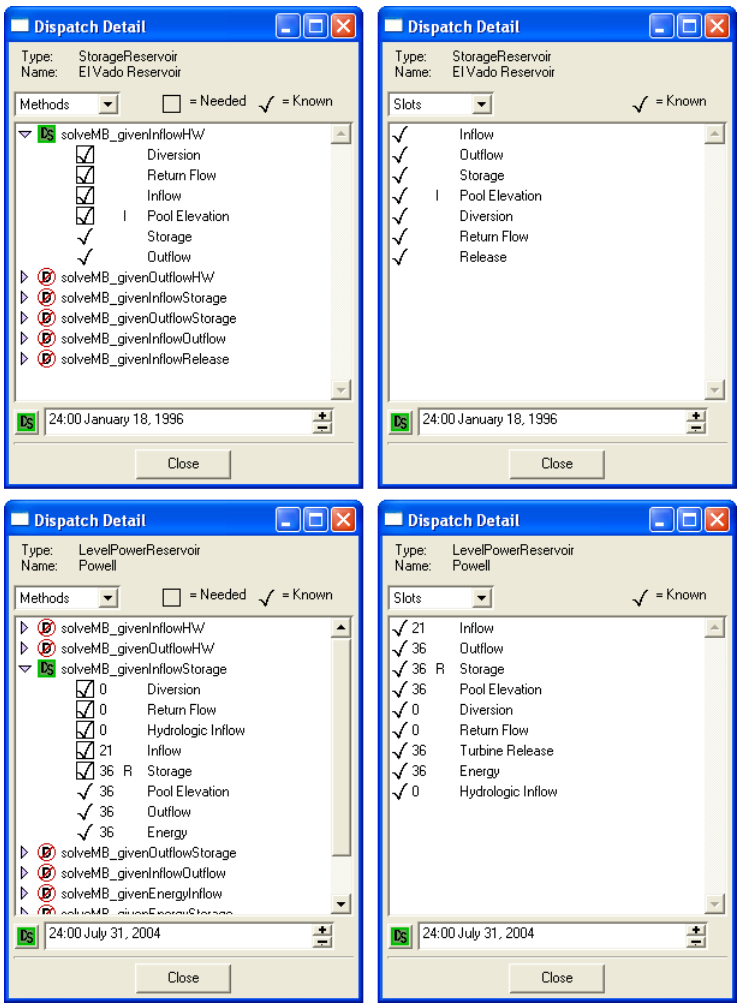
Alternative B (recommended, see below): Use a **distinct synchronized-scroll rwQTable** (single column, no vertical QHeader). We would have to implement our own method of reordering SimObj items, e.g. with the puffy up-arrow and down-arrow which move the selected rows up or down one position (e.g. See the Object Accounts Summary dialog's Account panel).

Alternative C: Hybrid. Use a **distinct synchronized-scroll rwQTable WITH its own minimal vertical QHeader.** SimObj names would be drawn in this rwQTable's data cells. Custom re-ordering would be implemented with dragging the narrow, textless QHeader cells.

Recommendation (Phil, 11-28-2006): Alternative B. Although synchronizing QTable scrolling operations is non-trivial, we've done this before (e.g. in the SCT), and will be doing it for the column header in this application. This is actually a more internally-consistent approach. Also, there are usability problems with re-ordering by dragging: (a) The availability of that feature is not apparent to the user based on visible GUI elements, and (b) Move-by-dragging is inconsistent with multiple row selection by dragging. Even if the latter is disabled, the user will inadvertently start a drag when intending to affect a multiple row selection. Bad.

2.2 Dispatch Detail Dialog

Double-clicking within a cell in the Object / TimeStep Dispatch Info Table causes a Dispatch Detail Dialog to be shown for the indicated SimObj and TimeStep. The screenshots to the right show examples of the Dispatch Detail Dialog's two modes (Methods vs. Slots) for Simulation vs. Rules.



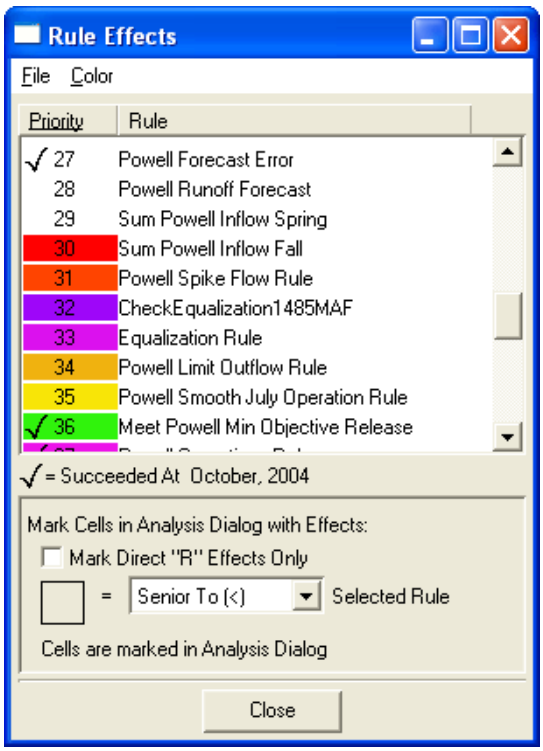
For the Qt implementation, the contents of the Dispatch Detail dialog will be implemented as a **distinct panel** (probably a QFrame) so that it can be managed standalone (within a QMainWindow, possibly with a menubar) or as an "auxiliary panel" within the Model Run Analysis dialog.

A multiple-level (tree-view mode) QListView subclass will be used with only a single column, and with the QHeader hidden. The multiple columns of the sub-elements in the "Methods" views will be accomplished using custom drawing of the QListViewItem (including the painting of the box and/or check icons). This is necessary because QListView doesn't support distinct column schemes for different treeview levels.

2.3 Rules Effects Dialog

The Rules Effects dialog is a singleton. It shows:

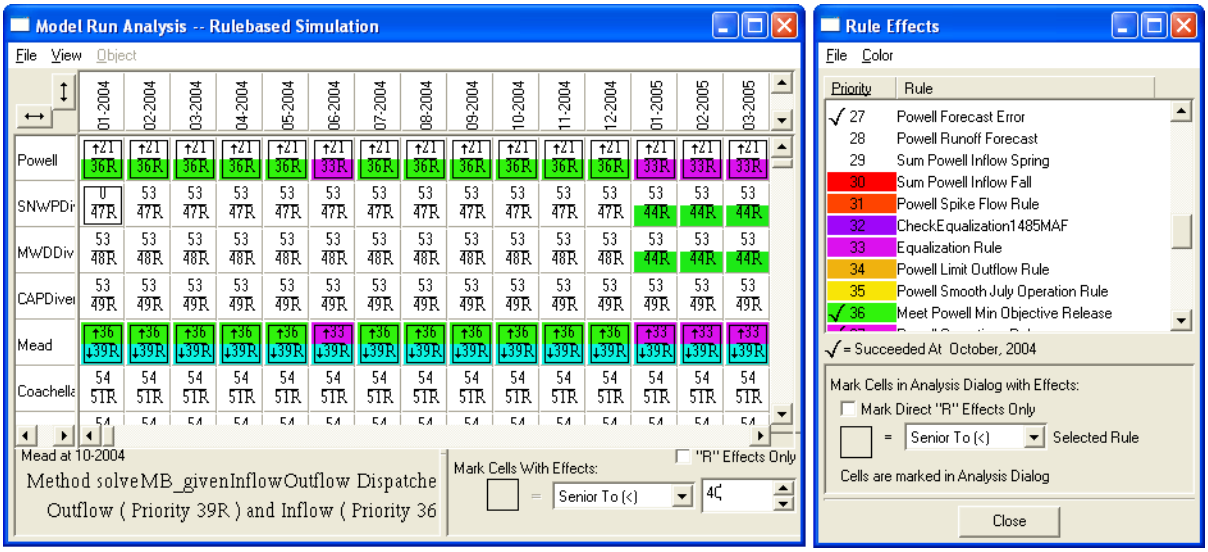
- Information related to a single TimeStep across all SimObjs.
- User settable color assignments for the various Rules.



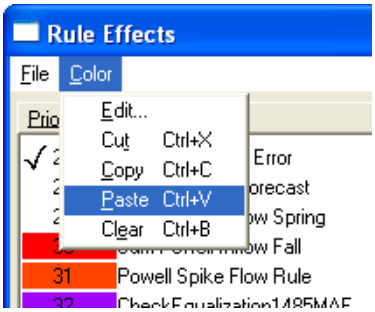
In the current Galaxy implementation, this dialog is shown from the Model Run Analysis / Rules mode dialog menu: “View >> List Rules / Set Rule Colors ...”.

This dialog maintains a redundant Effects Ornamentation Panel. It’s state is maintained in parallel with the similar panel within the Model Run Analysis / Rules mode dialog box. One different of the presentation is that the Rule number parameter is represented as the selected Rule row instead of in an integer spinner. Selecting a row in the Rules Effects dialog sets the value of the integer spinner. Also, selecting a time in the Model Run Analysis (e.g. by clicking a cell) changes the TimeStep of the Rules Effects dialog.

Recommendation: Instead of showing the Rule Effects dialog as a distinct dialog, show it as an optional “auxiliary” side panel in the Model Run Analysis / Rules mode dialog (which is also a singleton) -- with the ability to show either or both of the table and the Rules Effects panel, and to manually redistribute the horizontal space shared by these panels. This has the advantage of needing to show only a single Effects Ornamentation Panel among the two dialogs. Below is an illustration of the current two Galaxy dialogs side-by-side:

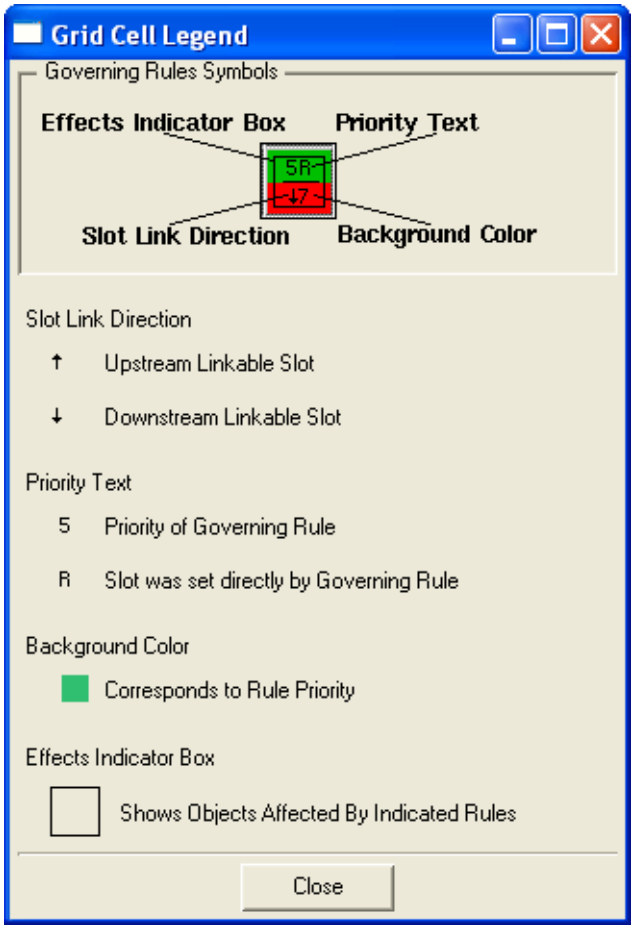


Color associations with Rules (for background cell color in the Model Run Analysis dialog) are edited using the Rules Effects dialog. See the Color menu diagram to the right. The “Color >> Edit ...” function brings up the respective platform’s color chooser to assign a new color to the currently selected Rule row. (i.e. the Windows XP color chooser when running on Windows, and some other native color chooser on other platforms).



2.4 Grid Cell Legends

Two distinct Grid Cell Legend dialogs are managed from the Model Run Analysis dialog, depending on the display mode: Simulation or Rules. These are static displays. They are drawn dynamically using the same low-level utilities used elsewhere in these dialogs so as to exactly match the depicted ornamentations.



3.0 High Level GUI Design

The set of Qt Widgets which compose each dialog box (with the possible exception of the very simple support dialogs) will be contained an independently manageable “panel” widget (generally, a QFrame widget) so that those GUI controls can be managed either as a stand-alone dialog box (generally a QMainWindow, so that a menubar can be accommodated) or as an integrated “auxiliary” panel. In the latter case, the containing dialog box would need to provide menubar menus specifically for the auxiliary panel. This applies to the following dialog boxes:

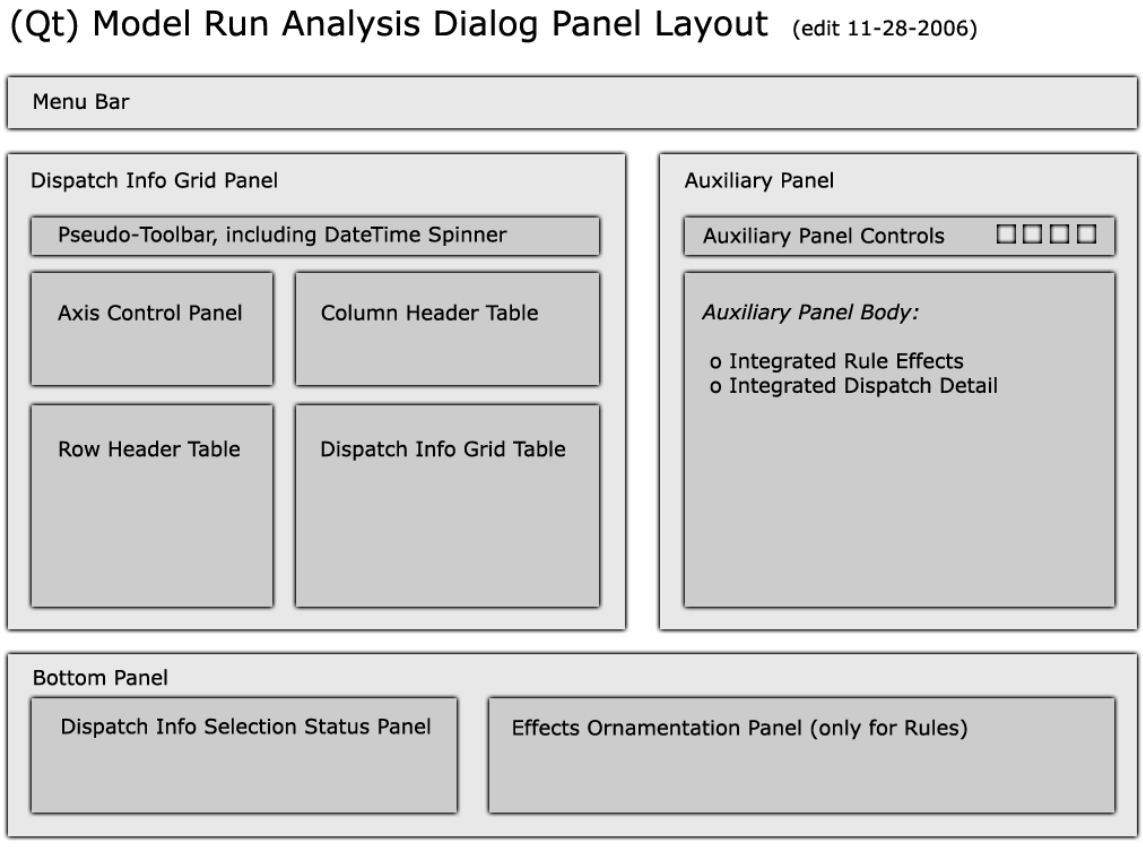
Dialogs which can be managed as an auxiliary panel:

- Rules Effects Dialog / Panel
- Dispatch Detail Dialog / Panel

Dialogs which can manage an auxiliary panel:

- Model Run Analysis Dialog

3.1 Model Run Analysis Dialog - High Level Design



The Model Run Analysis dialog will be a QMainWindow with a menubar. See the schematic layout diagram above. Two top-level internal panels will be managed vertically:

```
QSplitter* _centerPanel;       // horizontal splitter containing:
RunAnalDispatchInfoGridPanel*; // a QFrame
RunAnalAuxiliaryPanel*;        // a QFrame, conditionally shown

QFrame* _bottomPanel;          // horizontal layout containing:
RunAnalDispatchInfoSelectionPanel*;
RunAnalEffectsOrnamentPanel*;
```

The Display Info Grid Panel (class RunAnalDispatchInfoGridPanel) contains:

1. a pseudo-toolbar frame along the top, including a Date/Time TimeStep spinner (for navigation).
2. a 2x2 grid supporting a central grid table (lower right) with two synchronized-scroll header tables for the row and column headers, and an **Axis Control Panel** (upper left) containing different sets of widgets, depending on a mode setting.

The **Axis Control Panel** (class RunAnalAxisControlPanel, probably a QWidgetStack subclass) will have at least two modes having the indicated widgets:

1. Default mode: (a) Two Expand/Collapse toggle pushbuttons for expanding or collapsing the row and column headers (similar to the Galaxy implementation); (b) a swap-axis button; (c) a “Rules” toggle to switch between Rules mode and Simulation mode (for Rule models only).
2. Edit SimObj Order mode: Depending on the axis orientation, a pair of arrow buttons (either up/down or left/right) to move the selected SimObj rows or columns earlier or later in the SimObj list.

The **Auxiliary Panel** (class RunAnalAuxiliaryPanel) will contain a row of controls to manage the contents of the panel. It will include icon push buttons for:

- **Expand:** Repositioning the Splitter divider to give the panel it’s required width.
- **Condense:** Repositioning the Splitter divider to give the main panel the majority of the horizontal space.
- **Stand-Alone:** Show the panel contents as a stand-alone dialog (and give the main panel the entire available width of the dialog box).
- **Close** (“X”): Close the panel (and give the main panel the entire available width of the dialog box).

4.0 Intermediate Development Options

In case we don’t have time to complete the full design, the following simpler alternatives will first be developed to provide a reasonably functional subset.

4.1 Row and column headers implemented with QHeaders instead of with distinct QTables.

The inability to specialize the behavior of the QTable’s QHeaders (e.g. for customized drawing) motivated the specification of distinct synchronized-scrolled QTables to implement the Dispatch Info Grid’s row and column headers. Using at least one synchronized-scrolled QTable also makes possible the use of the top-left area between the headers for other widgets. (QTable doesn’t support use of that top-left area).

However, we will first implement the Dispatch Info Grid Panel with a single QTable. This has the following implications:

1. Text in the column header cannot be rotated. So, text in those columns (either SimObj names or TimeStep Date/Times) will be drawn using multiple lines, with one character per line.
2. Disabled Dispatching of SimObjs will be indicated with a red cross-hatch within the Dispatch Info Grid Table cells rather than on the SimObj names in the row or column headers.
3. Re-ordering of SimObjs *could* be implemented with drag-and-drop of SimObj names within a QHeader. (QHeader does provide some native support for this). If this approach is taken, then the Dispatch Info Grid Table will not support multiple cell selection (since multiple selection interferes with the drag-and-drop operation). The original recommended approach could still be used (i.e. blue puffy up and down arrows to move the selected SimObj rows up or down one position), but those controls would have to be located in a different place (i.e. not in the top-right area between the headers) -- probably in the Pseudo-Toolbar with the DateTime Spinner.

4.2 Non-swapable Axes: Support only SimObj Rows and Timestep Columns

~~4.3 No Auxiliary Panel: Support only a stand-alone Rule Effects dialog.~~

UPDATE [Phil, 12-11-2006]: Aux Panel / Stand Alone configuration controls, including "Detach" and "Dock" operations, WERE IMPLEMENTED for the Dispatch Detail display (under development). The Aux Panel is already instantiated as a two-widget Widget Stack; the second one will be used for the Rule Effects display.

UPDATE [Phil, 1-20-2007]: The Rules Effects (Rule / Priority List) display has been implemented ONLY AS an integrated panel -- and not as a stand alone dialog box.

Various considerations motivated the addition of an "Auxiliary Panel" to the right of the Dispatch Info Grid Panel within the Model Run Analysis dialog. The Rule Effects dialog is a singleton (as is the Model Run Analysis dialog), and is generally used in conjunction with the Dispatch Info Grid Table. So, it would be convenient to the user to manage those two panels (the Grid Table and the Rules Effect display) together within one dialog box (and support the showing of either one OR both simultaneously).

Also, in the current Galaxy implementation, both of these contain their own Effects Ornamentation Panels, which need to be dynamically synchronized. It seemed reasonable to eliminate that panel in the latter display if the two displays were generally shown together. In the context of a development shortcut, it's not unreasonable to eliminate the Effects Ornamentation Panel from the Rules Effects display even if the Rules Effects display remains a stand-alone dialog. (The presence of the Effects Ornamentation Panel within the Rules Effects display is not fundamental to the operation of that display).

The original design anticipated provisions for managing a Dispatch Detail panel within the Auxiliary Panel, with the capability of detaching the Dispatch Detail display as one or more instances of stand-alone dialogs since, sometimes multiple Dispatch Detail displays are needed.

As a development shortcut, we will deploy the Rules Effects panel only within a stand-along dialog, and not include an Auxiliary Panel in the Model Run Analysis Dialog. We will still omit the Effects Ornamentation Panel from the Rules Effects panel to avoid the need to dynamically synchronize two Rules Effects panel displays.

5.0 Development Tasks

TABLE 1. Run Analysis Dialog Qt Port Development Tasks

Task	Partial Design Days (est).	Compl.	Full Design Days (est)	Compl.	Description
Model Run Analysis Dialog / Basic Operation / Simulation Mode					
1.0	2.0	**12-01**	2.0	**12-01**	Panel Layout / Dynamic showing of Aux Panel. (Size adjustment ops developed later)
1.1	0.5	**12-01**	1.0	**Done**	Menu Bar with conditional dynamic menus for Aux Panel
1.2a	2.0	**12-03**	2.0	**12-03**	Dispatch Info Grid Panel management (class RunAnalDispatchInfo-GridPanel); create and manage the five sub-panels; design and implement row and column data structures; define simulation dispatch info icons (four); populate Grid Table with dummy content.
1.3a	3.0	**12-06**	3.0	**12-06**	Basic Simulation Dispatch Info display in the Dispatch Info Grid Table. (Paint Row and Column headers; Paint one of four basic icons in each table cell). Respond to appropriate callbacks for cell refresh.
1.3b			4.0		Composite QTable -- Row and Column header as Aux. QTables.
1.4	1.0	**12-06**	1.0	**12-06**	DateTime spinner implementation; Dispatch Info Grid Table timestep scroll.
1.5			3.0		Swappable Axis / Axis Control Panel controls
1.6a	0.5	**12-14**	0.5	**12-14**	SimObj sorting.
1.6b			3.0		Custom SimObj Sort Order editing
1.7	2.0	**12-07**	2.0	**12-07**	Dispatch Info Selection Status Panel (for Simulation Dispatch Information)
Dispatch Detail Panel / Dialog					
2.0	2.0	**12-10**	2.0	**12-10**	Widget layout and management, including QWidgetStack with two QListViews for Methods and Slots views.
2.1			1.0	**12-10**	Management of Dispatch Detail Panel within the Model Run Analysis “auxiliary” panel.
2.2	1.0	**12-10**	1.0	**12-10**	Management of Dispatch Detail Panel as a stand-alone dialog.
2.3	1.5	**12-13**	1.5	**12-13**	QListViewItem subclasses; Custom pseudo-column drawing
2.4	3.5	**12-19**	3.5	**12-19**	Dispatch Methods tab / listview (with actual Dispatch data)
2.5	1.0	**12-13**	1.0	**12-13**	Dispatch Slots tab / listview (with actual Dispatch data)
Model Run Analysis Dialog / Rules Mode					
3.0	4.0	**12-29**	4.0	**12-29**	Rule Dispatch Grid Cell Drawing
3.1	1.0	**1-2**	1.0	**1-2**	Dispatch Info Selection Status Panel (for Rule Dispatch Information)

TABLE 1. Run Analysis Dialog Qt Port Development Tasks

3.2	2.0	**1-5**	2.0	**1-5**	Effects Ornamentation Control Panel / Grid Cell Ornamentation.
Rules Effects Dialog					
4.0	1.5	**Done**	2.0		Widget layout and management: **DONE:** (a) panel, **NOT DONE:** (b) stand-alone dialog with menubar; Manage within the Model Run Analysis “auxiliary” panel, or standalone.
4.1	2.0	**Done**	2.0	**Done**	QListViewItem subclass; custom drawing with color backgrounds and check icon.
4.2	2.0	**Done**	2.0	**Done**	Selection Interactions with (to/from) the Model Run Analysis panel
4.3	2.0	**Done**	1.0	**Done**	Color Edit operations
Minor Support Dialogs					
5.0	1.0	**12-19**	1.0	**12-19**	Simulation Grid Cell Legend
5.1	2.0	**Done**	2.0	**Done**	Rules Grid Cell Legend. (Actual custom drawing using the same code used to draw actual grid table cells).
Finishing					
6.0	1.0		3.0		Automatic and manual geometry management functions / size adjustments
6.1	2.0		2.0		Completeness and Usability testing and fixing
Total	40.5		53.5		

5.1 Deferred Development Issues

5.1.1 QHeaders: Can’t shrink in response to shorter longest text string

[12-6-2006, Task 1.3, Partial Design]. Failed attempt to shrink the height of the TimeStep column QHeader (e.g. when changing them model timestep from hourly to daily, which doesn’t show hours). We’ve never succeeded in doing this. **WORKAROUND:** Re-open the Model Run Analysis dialog. This will need to be addressed especially if and when we implement explicit "condense" header functions. We may have to destroy and recreate the table -- or else use synchronized-scrolled header QTables instead of native QHeaders ... See use of _timeDispStrRefLenSave in RunAnalGridQTable::rebuildTable();

--- (end) ---