Commonality Analysis for Header Table Tool

June 9, 1999

“Do you know where your cursor is?”

Mark Ardis, Dan Hoffman, Harvey Siy

INTRODUCTION

We want to produce a family of tools that support storing and processing of student grades. Users are instructors and teaching assistants. Every user must be able to enter and maintain information about students, assignments, and scores. In addition, users need to analyze, calculate, and report grades. We want the tool to be easy to learn, use, and adapt to user preferences; and to be portable across Unix, PC, and Mac platforms. While the main focus is grading, this tool can be used for simple editing and analysis of any data that fits well into the header table format.

OVERVIEW

Figure 1 shows the rough appearance of the application window of Grades, a typical member of the Header Table Tool family. Each column represents an assignment. The column header cells provide information about the assignment: points for a perfect solution, percentage of the course grade, and name. Each row represents a student; with the row header cells storing the student’s first and last name. At the intersection of each assignment column and student row is the score the student earned on the assignment. Rows and columns may be inserted and removed. At any given time, one cell has the cursor; only the cell with the cursor can be modified. Because there may be many more students and assignments than can fit on the screen, vertical and horizontal scrolling is supported. The header table can be stored and retrieved from disk.

The family variabilities fall into four main categories:
1. Static presentation, especially the window shape: the number of columns, rows, column headers, and row headers.
2. Cursor window repositioning: the rules determining how the cursor and window are repositioned as a side-effect of other commands. These rules answer questions such as: how should the window be repositioned when the last visible row is removed?
3. Cell validation policy: the rules determining what values may be entered in each cell and what happens when illegal values are entered.
4. Computation and reporting: the calculations provided, e.g., the course average for each student, and the format for reports, e.g., an HTML table.

We expect that most variabilities will be bound at application engineering time, with a few at domain engineering time (e.g., type family) or run-time (e.g., scroll delta).
DEFINITIONS
In this section and in the following sections the terms defined will be italicized. These definitions would be easier to use if in alphabetical order

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application window</td>
<td>the GUI window containing the application, including the window table and command menus</td>
</tr>
<tr>
<td>base cursor position</td>
<td>the position of the cell in the base table corresponding to the window cursor position and the window position.</td>
</tr>
<tr>
<td>base table</td>
<td>header table containing the underlying data being edited</td>
</tr>
<tr>
<td>column header</td>
<td>a row in a column header matrix</td>
</tr>
<tr>
<td>column header label</td>
<td>a string that appears to the left of the leftmost column header text cell</td>
</tr>
<tr>
<td>column-header matrix</td>
<td>see Figure 1.</td>
</tr>
<tr>
<td>core matrix</td>
<td>see Figure 1.</td>
</tr>
<tr>
<td>disk table</td>
<td>header table that is on disk</td>
</tr>
</tbody>
</table>

Figure 1: Example header table
enabled/disabled cell

A window table cell is enabled if and only if it has an underlying base table cell.

header table

Stores 3 closely-linked matrices: core matrix, column-header matrix, row-header matrix. The cells in each matrix are addressed relative to (0,0), with the origin in the upper-left corner. The first coordinate is the horizontal displacement and the second is the vertical displacement within the matrix.

row header

A column in a row header matrix.

row header label

A string that appears above the top row header text cell.

row-header matrix

See Figure 1.

scroll delta

The amount the window gets shifted if the window has to be repositioned after a row or column deletion.

shape

A four-tuple of (number of rows in the core matrix, number of columns in the core matrix, number of row headers, number of column headers); see TableShape class.

type

Set of string values legal for all cells in a type region.

type family

Set of available values for CVP type.

type region

Any of a row header matrix, a column header matrix, or a core matrix.

validation trigger event

An event that causes cell validation to be performed.

window cursor position

A (m,x,y) triple indicating the window table cell with the keyboard focus, where m is column header matrix, row header matrix, or core matrix, and x and y indicate the cell within the corresponding matrix; see FieldId class.

window position

An (x,y) pair indicating the position of the window table in the base table, where x is the number of base table columns to the left of the window table and y is the number of base table rows above the window table; see WindowPosition class.

window table

Header table containing the data that is currently visible.

ACRONYMS AND ABBREVIATIONS

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application programming interface</td>
</tr>
<tr>
<td>CWR</td>
<td>Cursor window repositioning</td>
</tr>
<tr>
<td>CVP</td>
<td>Cell validation policy</td>
</tr>
</tbody>
</table>

ASSUMPTIONS

The following statements are basic assumptions about the domain of header tables.

We assume that with the shape chosen for the window table, the entire window table will be visible in the application window.
COMMONALITIES

Static Presentation
When you bring up the application, what do you see? For example, all the colors, borders, fonts, column header labels are part of the static presentation.

C1. There is always a single application window containing a single window table.
C2. The application window may not be resized.
C3. The number of row headers and number of column headers are the same in the window table and the base table. (See V2)
C4. The window table will always fit inside the application window, i.e., the entire window table will be visible in the application window.
C5. The window table is a window into the base table. The enabled cells in the window table display a subset of cells in the base table with the origin of the window table given by the window position. (See V5)
C5.1. There will be a way to invoke each of the file (open, save, exit) and edit (insert/delete row/column) commands mentioned in the Overview. (See V4.1)

Cursor Window Repositioning
The commonalities here express limits on the positions of the cursor and window.

C6. Whenever the window table is the active window, there is always exactly one window table cell with the window cursor. There is one exception: if the base table is empty, then no window table cell has the window cursor. (See V6)
C7. There are four unique Unicode characters for moving the window position left, right, up and down. It will not be possible to enter these characters as table cell values. (See V7)
C8. The column header matrix of the window table always has the same horizontal displacement as its core matrix.
C9. The row header matrix of the window table always has the same vertical displacement as its core matrix.
C10. Frequently, the cursor or window has to be repositioned as a side effect of another operation. There will be an explicit cursor/window repositioning policy. (See V8)

Cell Validation Policy
There are three components of validation: legal values, check time (when do you check?), and failure action (what do you do on failure?). In some cases, the user will be forced to fix an error before continuing.

C11. Each type region has the same type. (See V9, V10, V11)
C12. All cells have the same granularity, error action, error signal, and error highlighting. (See V12)
C13. The cell values are strings. (See V11)
C14. Each type has a unique name.
C15. If CVP granularity (P4) is "cell" and CVP error action (P5) is "reject", every cell in the window table and base table is valid, except possibly the window table cell with the
window cursor position. If CVP error action is "reject", every disk table cell is valid.

C16. There is an additional edit command (see V4.1) called "next invalid cell" unless the CVP granularity (P4) is "cell" and the CVP error action (P5) is "reject".

C17. The validation trigger events for CVP granularity = "cell" are: lost focus, insert column, insert row, scrolling, save and save as. (See Issue 6)

C18. The validation trigger events for CVP granularity = "form" are: save and save as, except that, for error highlighting (P7) and unhighlighting, the validation trigger events for CVP granularity = "cell" are used.

Computation/Reporting
What do you compute and what is the formula? For example the course grade is computed.

C19. The computation engine can load any base table and manipulate it with the base table API and store it as another base table.

C20. The reporting engine can load any base table and manipulate and display it using the base table API.

VARIABILITIES
The following statements describe how header tables may vary.

Static Presentation
When you bring up the application, what do you see? For example, all the colors, borders, fonts, column and row header labels are part of the static presentation.

V1. The number of rows and columns in the window table may vary from 1 to a small integer constrained by the size of the application window. (See P10)

V2. The number of row headers and number of column headers in the window table and base table may vary from 1 to a small integer. (See P10)

V3. The number of rows and columns in the base table may vary from 0 to a large integer. (See P11, P12)

V4. The row header labels and column header labels may vary over strings. (See P13, P14)

V4.1. The invocation mechanism for the file and edit commands may vary, for example, pull-down menus, keystrokes, toolbars.

Cursor Window Repositioning
The variabilities here express limits on the positions of the cursor and window.

V5. The window may not move above the first row or below the last row, i.e., y in [0..r-1], where y is the number of base table rows above the window and r is the number of rows in the base table. Similarly, x in [0..c-1] where x is the number of base table columns to the left of the window and c is the number of columns in the base table. (See P15)

V6. The window cursor position may vary over any of the enabled cells in the window table. (See P16)

V7. The keys for moving the window position may vary among any Unicode characters as long
as they are distinct and available from the keyboard. (See P17)

V8. When repositioning the cursor or window after any operation aside from using the mouse, the cursor or window can be repositioned along the following dimensions: *cursor/table binding, scroll delta, post-insertion point*. (See P1, P2, P3)

**Cell Validation Policy**

There are three components of validation: legal values, check time (when do you check?), and failure action (what do you do on failure?). In some cases, the user will be forced to fix an error before continuing.

V9. The CVP *type family* varies at domain engineering time. Each member of the family comprises an arbitrary boolean function on strings. (see P9)

V10. The CVP *type* associated with a given *type region* varies at application engineering time. (See P8)

V11. The table cell values may vary over strings, within the constraints of the CVP *type*. (See P8)

V12. The cell validation policies will vary along the following dimensions: type, granularity, error action, error signal, error highlighting. (See P4, P5, P6, P7, P8, P9)

**Computation/Reporting**

What do you compute and what is the formula? For example the course grade is computed.

V13. We expect most reports to simply display the *base table* plus 0 or more computed rows, columns or separate cells.

**PARAMETERS OF VARIATION**

The following parameters characterize the variabilities allowed in the domain of header tables.

<table>
<thead>
<tr>
<th>#</th>
<th>Parameter &amp; Variability</th>
<th>Meaning</th>
<th>Domain</th>
<th>Decision Time</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1.</td>
<td>CWR: scroll delta (V8)</td>
<td>(see definition)</td>
<td>{min, half, max}</td>
<td>run-time</td>
<td>max</td>
</tr>
<tr>
<td>P2.</td>
<td>CWR: cursor/table binding (V8)</td>
<td>attempt to minimize cursor movement in base or window table</td>
<td>{base, window}</td>
<td>application engineering</td>
<td>window</td>
</tr>
<tr>
<td>P3.</td>
<td>CWR: post-insertion point (V8)</td>
<td>cursor position after insertion</td>
<td>{no change, new row or column}</td>
<td>application engineering</td>
<td>new row or column</td>
</tr>
<tr>
<td>P4.</td>
<td>CVP: granularity (V12)</td>
<td>class of events that triggers the validation</td>
<td>(cell, form)</td>
<td>application engineering</td>
<td>form</td>
</tr>
<tr>
<td>P5.</td>
<td>CVP: error action (V12)</td>
<td>when validation fails, accept or reject the event that triggered the validation</td>
<td>(accept, reject)</td>
<td>application engineering</td>
<td>accept</td>
</tr>
<tr>
<td>P6.</td>
<td>CVP: error signal (V12)</td>
<td>immediate and transient notification of error</td>
<td>(beep, message, both, none)</td>
<td>application engineering</td>
<td>none</td>
</tr>
<tr>
<td>P7.</td>
<td>CVP: error highlighting (V12)</td>
<td>persistent notification of error</td>
<td>(on, off)</td>
<td>application engineering</td>
<td>off</td>
</tr>
<tr>
<td>P8.</td>
<td>CVP: type (V10, V11, V12)</td>
<td>legal values for the cell</td>
<td>type family</td>
<td>application engineering</td>
<td>any</td>
</tr>
<tr>
<td>P9.</td>
<td>CVP: type family (V9, V12)</td>
<td>available values for P8</td>
<td>handcoded boolean functions on strings</td>
<td>domain engineering</td>
<td>{alphabetic, alphanumeric, integer, fixed point, any}</td>
</tr>
<tr>
<td>P10.</td>
<td>Window table shape (V1, V2)</td>
<td></td>
<td>&lt;1..20, 1..10, 1..10, 1..20&gt;</td>
<td>application engineering</td>
<td>&lt;6, 4, 3, 3&gt;</td>
</tr>
<tr>
<td>P11.</td>
<td>Base table number of rows (V3)</td>
<td></td>
<td>0..1000</td>
<td>run-time</td>
<td>0</td>
</tr>
<tr>
<td>P12.</td>
<td>Base table number of columns (V3)</td>
<td></td>
<td>0..1000</td>
<td>run-time</td>
<td>0</td>
</tr>
<tr>
<td>P13.</td>
<td>Row header labels (V4)</td>
<td>string of printable characters of length 0..15</td>
<td>application engineering</td>
<td>no default</td>
<td></td>
</tr>
<tr>
<td>P14.</td>
<td>Column header labels (V4)</td>
<td>string of printable characters of length 0..15</td>
<td>application engineering</td>
<td>no default</td>
<td></td>
</tr>
<tr>
<td>P15.</td>
<td>Window position (V5)</td>
<td></td>
<td>&lt;0..c-1, 0..r-1&gt;, where c and r are number of columns and rows, respectively, in the base table</td>
<td>run-time</td>
<td>&lt;0, 0&gt;</td>
</tr>
</tbody>
</table>
Table 6. Window cursor position (V6)

| m, x, y where m is one of {null, core, row header, column header}, x and y are the offset into the window table |
| run-time <null, 0, 0> |

Table 7. Window movement keys (V7)

| l, r, u, d where they are any Unicode characters that are distinct and available from the keyboard |
| application engineering shift-arrow keys |

ACKNOWLEDGMENTS

The authors wish to thank the following people for their valuable contributions: Nigel Daley, Dave Weiss.

ISSUES

The following are key issues considered during the commonality analysis.

Issue: 1. What to do with the display resolution: will this determine the bounds on (number of row headers + number of columns) and (number of column headers + number of rows), i.e., the width and height of the window table. (See V1 and V2)-- Dan

Resolution: Assume that with the shape chosen for the window table, the entire window table will be visible in the application window.

Issue: 2. Locking cells (read-only cells) might be useful. Do we want to provide this feature? -- Mark

Resolution: Added to wish list.

Issue: 3. Should CVP granularity and error action be allowed to vary a) cell by cell, b) bound to the type, or c) uniform across all cells? -- Dan

Resolution: b) because it seems most adequate -- a) is too much freedom and c) is too restrictive.

Issue: 4. Which CVP combinations can be allowed? -- All

Resolution: All combinations are permitted, but two are considered unacceptably hard to use: (1) reject with error signalling and highlighting off, and (2) reject with form granularity.

Issue: 5. What special values should be permitted, e.g., null values in numeric fields? How should they be handled? -- All

Resolution: The domain engineer has to create a new type which allows for nulls and the
application engineer must accommodate nulls in his computation.

Issue: 6. Should Open/Exit be a validation trigger? -- All

Resolution: No. We should not constrain the user from aborting. If they want to save, they will be forced to validate.

WISH LIST
1. There should be a way to make selected cells read-only.
2. Scrollbars might be better than arrow keys.
3. A diff-like feature would be useful.
4. Provide keyboard shortcuts for moving the window cursor position left, right, up and down. (Issues: how to move between matrices? should the window be dragged?)
5. Provide character level CVP granularity.
6. Provide computed rows or columns in the window table, e.g., a course average column.
7. The core header table functionality might make a nice applet or AWT component.

TO DO
Give out to reviewers: Dave Cuka, Lloyd Nakatani