ISO/IEC JTC 1/SC34 N0471

ISO/IEC JTC1/SC34/WG2 N133

Information Technology --
Document Description and Processing Languages

TITLE:	ISO/IEC TR 19758/PDAM1: Extensions to Basic Composition Styles and Tables
SOURCE:	Project editors
PROJECT ID:	tr19758a1
PROJECT EDITOR:	Gen Nagamura and Yushi Komachi
STATUS:	PDAM text
ACTION:	PDAM Processing
DATE:	2003-12-07
DISTRIBUTION:	SC34, SC34/WG2 and Liaisons
REFER TO:	
REPLY TO:	Dr. James David Mason (ISO/IEC JTC 1/SC 34 Chairman) Network: masonjd@y12.doe.gov Mr. G. Ken Holman (ISO/IEC JTC 1/SC 34 Secretariat - Standards Council of Canada) Network: jtc1sc34@scc.ca

---

DSSSL library for complex compositions – Amendment 1: Extensions to Basic Composition Styles and Tables

Introduction

This Amendment specifies additional DSSSL specifications for basic composition styles and table compositions.

Page 5, Clause 4.4

Add the following clause 4.4.1 before 4.5

4.4.1 Graphical method using Unwin module concept)

Content-driven specification can be determined by using graphical method based on Unwin module concept.

Following procedure and image are how all four margins are specified by graphical method referring Unwin module concept.

a) Specify S point
b) Draw diagonal line from the top-right edge through S point to bottom left edge.
c) Draw line from left page of bottom-left edge to right page of top-right edge
d) Draw line from left page of top-left edge to right page of bottom-right edge
e) Draw line from left page of top-left edge to left page of bottom right edge
f) Draw line from right page of top-left edge to right page of bottom right edge
g) Draw horizontal line from S point until reaching line which was drawn on procedure 4
h) Draw vertical line from point A until reaching line which was drawn on procedure 2
i) Draw horizontal line from point B and vertical line from S to set point C

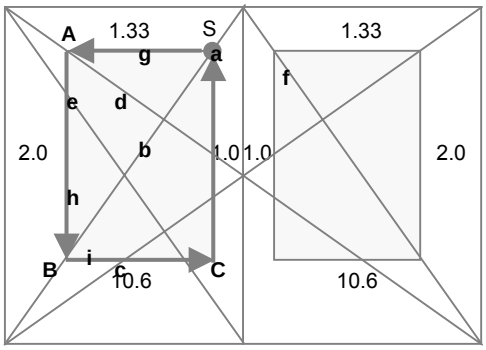

Figure A1.1 procedure of graphical method

However, this method has problems to be content-driven.

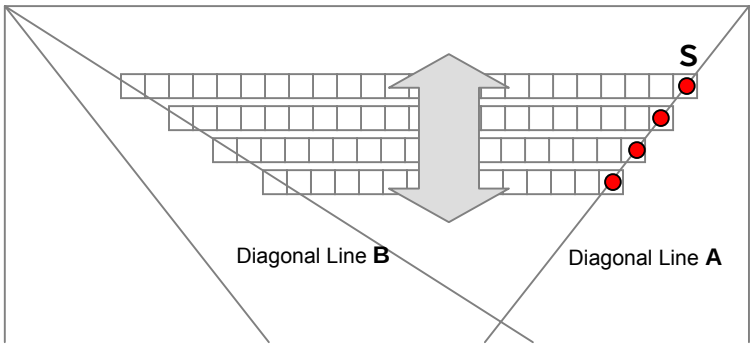

Figure A1.2 Specification of starting point

The number of em characters in a line and the number of lines in a page must be specified in content-driven specification. Therefore, starting point S is determined by such a closest starting point on diagonal line A as has closest number of designated the number of em characters in a line between diagonal line A and B.

Following equation is to calculate length of line floating direction

Length of line direction = (# of Line -1) * line spacing + character size

By inserting this equation, graphical method is applied to content-driven method.

In case of using above content-driven method as DSSSL library, specification of right margin, font size, number of lines, value of line spacing.

However, four inputting data is not efficient as DSSSL library, accordingly, limit the inputting data to values of **gutter margin** (*gutter-margin*) and **font size** (*default-font-size*).
Default value of line-spacing is designated as 1.5em.

Page34, Clause 6

Add the following specification before the specification of position of image data

```
;;;;;;;;;;; graphical method using Unwin module concept
;;
;;
(define *gutter-margin* 10mm)
(define *default-font-size* 15pt)

(define *a4-width* 210.0mm)
(define *a4-height* 297.0mm)

(define-unit okuri (* *default-font-size* 1.5))
(declare-initial-value font-size *default-font-size*)
(declare-initial-value line-spacing 1okuri)

(define _PH_ *a4-height*)
(define _PW_ *a4-width*)
(define _TM_ (* *guttert-margin* (/ _PH_ _PW_)))
(define _BM_ (* 2 _TM_))
(define _LM_ (* 2 _RM_))
(define _RM_ *gutter-margin*)
(define _RW_ (* *default-font-size*
             (ceiling (/ (- *a4-width* (* 3 _RM_)) *default-font-size*))))
```

```
(define *gyosu* (ceiling
              (/ (- (+ (- *a4-height* (* 3 _TM_)) 1okuri) *default-font-size*)
                 1okuri)))
(define _RH_ (+ (* (- *gyosu* 1) 1okuri) *default-font-size*))

(define-page-model unwin_module_a4_left
  (filling-direction 'top-to-bottom)
  (width _PW_)
  (height _PH_)
  (region
    (width _RW_)
    (height _RH_)
    (x-origin (- *a4-width* _RW_ _RM_))
    (y-origin _BM_))
  )

(define-page-model unwin_module_a4_right
  (filling-direction 'top-to-bottom)
  (width _PW_)
  (height _PH_)
  (region
    (width _RW_)
    (height _RH_)
    (x-origin _RM_)
    (y-origin _BM_))
  )
```

Page 14, Clause 4.18

Add the following clauses after 4.18.2

4.18.3 rounded corner table

Shape of table is rectangle, though corner is allowed to be rounded.

- If you wish to round the corner of table, use *corner-rounded-table-style* under the specification of element table
- Radius of corner is specified on (define *base-table-corner-radius* *3pt*). By changing the number of above **define**, the radius of table corner is specified as the value of it. Default value of table corner radius is 3pt.

4.18.4 Table header column and row

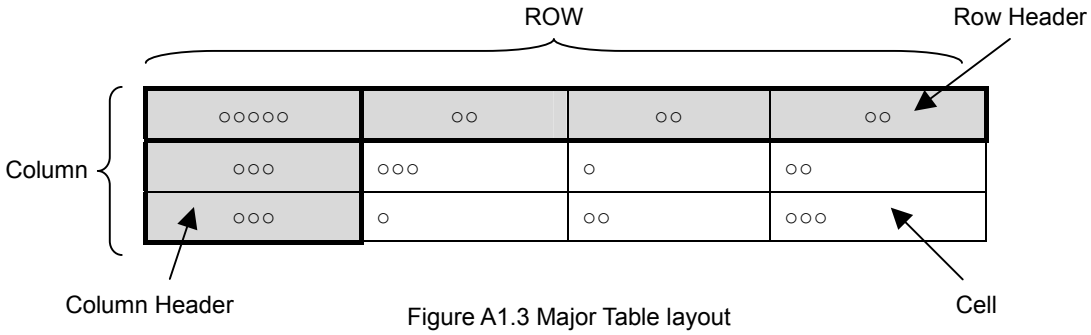

Figure A1.3 Major Table layout

Add <THC> as table header column and <THR> as table header row to regular column and row definition. Both <THC> and <THR> are initially specified back-ground color as light-gray. <THC> is siblings of table-column<TC> and <THR> is sibling of table-row<TR>, and they have same children. Therefore, these two specifications are treated as same as usual table-column and table-row.

4.18.5 Multiple string in table cell

alpha / num	○○	○○	○○
○○○	○○○	○	○○
○○○	○	○○	○○○

Figure A1.4 Multiple strings in table cell

Multiple table cell <MTD> is the specification of above shape of table cell as a sibling of

table-cell <TD>. This element has Row String <RSTR> and Column String <CSTR> as children. RSTR is the #PCDATA which is related to row data, CSTR is the #PCDATA which is related to column data.

When following cell is specified, layout of this table cell is shown upper left cell of above table.

```
<MTD>
   <RSTR IMAGESRC="">alpha</RSTR>
   <CSTR IMAGESRC="">num</CSTR>
 </MTD>
```

4.18.5.1 Diagonal line through table

If you fish to specify to draw diagonal line through whole table like league match table, specify diagonal line as the attribute of element TABLE. One restrict rule of this designation is number of columns and rows must be same number.

4.18.6 Word wrapping

Descend font size if string in the cell is exceeded to the width of table cell. To designate this attribute, DesFSize=”#t” means font is descended if the string is exceeded, “#f” means linefeed.

Page56, Clause 9

Replace existing Table specification with the following specification

```
;; ============================== TABLES ================================
;; Specification of table corner rounded style
(declare-characteristic table-corner-rounded
  "UNREGISTERED::Next Solution//Characteristic::table-corner-rounded" "#f")

;; Specification of actual table element
(element TABLE
        (let ((number-of-columns
                (node-list-reduce (node-list-rest (children (current-node)))
                      (lambda (cols nd)
                        (max cols
                             (node-list-length (children nd))))
                      0)))
        (make display-group
       space-before: %block-sep%
       space-after: %block-sep%
       start-indent: %body-start-indent%
        (with-mode table-caption-mode (process-first-descendant "CAPTION"))
        (make table
```

```
                use: *table-style*
;; Replace below designation to above if you wish to round table corner
;; use: *corner-rounded-table-style*

                (process-children)))))

(mode table-caption-mode
  (element CAPTION
         (make paragraph
              ;; use: para-style
                  quadding: 'center
              font-weight: 'bold
              space-before: %block-sep%
              space-after: %para-sep%
              start-indent: (inherited-start-indent);
              (literal
               (string-append
                "Table "
                (format-number
                 (element-number) "1") ". "))
              (process-children-trim))))

(element CAPTION (empty-sosofo)) ; don't show caption inside the table


(element TC (make-column "COLUMN_WIDTH"))
(element THC (make-background-column "COLUMN_WIDTH"))


(element THR
  (make table-row
         use: *background-cell-style*
       (process-children-trim)))

(element TR
  (make table-row
       (process-children-trim)))

(element MTD
  (make-multiple-cell
   (make paragraph (process-matching-children "RSTR"))
   (make paragraph (process-matching-children "CSTR"))))

(element TH
  (make table-cell
       ;n-rows-spanned: (string->number (attribute-string "COLSPAN"))
         quadding: 'center
         use: *cell-style*
       (make paragraph
            font-weight: 'bold
            space-before: 0.25em
            space-after: 0.25em
            start-indent: 0.25em
            end-indent: 0.25em
```

```
           quadding: 'start
           (process-children-trim))))

(element TD
  (make table-cell
       ;n-rows-spanned: (string->number (attribute-string "COLSPAN"))
         quadding: 'center
         use: *cell-style*
       (make paragraph
           space-before: 0.25em
           space-after: 0.25em
           start-indent: 0.25em
           end-indent: 0.25em
           quadding: 'start
           (process-children-trim))))

;; All definition toward table designation
(define *rgb-color-space*
  (color-space "ISO/IEC 10179:1996//Color-Space Family::Device RGB"))
(define (rgb R G B) (color *rgb-color-space* (/ R 255) (/ G 255) (/ B 255)))

(define *lightgray* (rgb 211 211 211))
(define *base-backgroud-color* *lightgray*)
(define *base-table-border* #t)
(define *base-table-corner-radius* 3pt)
(define *base-display-alignment* 'center)
(define *base-cell-margin* 1mm)
(define *base-cell-border* #t)
(define *base-cell-row-alignment* 'center)

(define *table-style*
  (style table-border: *base-table-border*
         display-alignment: *base-display-alignment*))

(define *corner-rounded-table-style*
  (style use: *table-style*
         table-corner-radius: *base-table-corner-radius*
         table-corner-rounded: "#t"))

(define *cell-margin-style*
  (style cell-before-row-margin: *base-cell-margin*
         cell-after-row-margin: *base-cell-margin*
         cell-before-column-margin: *base-cell-margin*
         cell-after-column-margin: *base-cell-margin*
         ))

(define *cell-border-style*
  (style cell-before-row-border: *base-cell-border*
         cell-before-column-border: *base-cell-border*
         ))

(define *cell-style*
  (style use: (merge-style *cell-border-style* *cell-margin-style*)
         cell-row-alignment: *base-cell-row-alignment*))
```

```
(define *background-cell-style*
  (style cell-background?: #t
         background-color: *base-backgroud-color*
         use: *cell-style*
         ))

(define (make-column attribute)
  (make table-column width: (* 1mm (string->number (attribute-string attribute)))))
(define (make-background-column attribute)
  (make table-column
    use: *background-cell-style*
    width: (* 1mm (string->number (attribute-string attribute)))))

(define *nonborder-style*
  (style cell-before-row-border: #f
         cell-before-column-border: #f))

(define *nonmargin-style*
  (style cell-before-row-margin: 0mm
         cell-after-row-margin: 0mm
         cell-before-column-margin: 0mm
         cell-after-column-margin: 0mm))

(define (unit-columns #!rest ws)
  (apply-map sosofo-append (lambda (w) (make table-column width: (table-unit w)))
ws))


(define *multiple-column-position* 13.5mm)
(define *multiple-row-position* 'center)
(define (make-multiple-pos position)
  (cond ((symbol? position)
         (style quadding: position))
        ((and (quantity? position) (not (real? position)))
         (style quadding: 'start
                start-indent: position))
        (else (error "invalid value for ¥"make-multiple-pos¥" characteristic"))))

(define (make-multiple-cell row-sosofo column-sosofo)
  (make table-cell
    cell-crossed: 'with
    use: (merge-style *cell-border-style* *nonmargin-style*)
    (make table
      cell-crossed: #f
      table-border: #f
      use: (merge-style *nonborder-style* *cell-margin-style*)
      (unit-columns 1 1 1)
      (make table-row
        (make table-cell (make paragraph (empty-sosofo)))
        (make table-cell
          use: (make-multiple-pos *multiple-row-position*)
          n-columns-spanned: 2
          row-sosofo))
```

```
    (make table-row
     (make table-cell
      use: (make-multiple-pos *multiple-column-position*)
      n-columns-spanned: 2
      column-sosofo)
     (make table-cell (make paragraph (empty-sosofo)))
     ))))
```