```
print.xtable <- function(x,
  type = getOption("xtable.type", "latex"),
  file = getOption("xtable.file", ""),
  append = getOption("xtable.append", FALSE),
  floating = getOption("xtable.floating", TRUE),
  floating.environment = getOption("xtable.floating.environment", "table"),
  table.placement = getOption("xtable.table.placement", "ht"),
  caption.placement = getOption("xtable.caption.placement", "bottom"),
  caption.width = getOption("xtable.caption.width", NULL),
  latex.environments = getOption("xtable.latex.environments", c("center")),
  tabular.environment = getOption("xtable.tabular.environment", "tabular"),
  size = getOption("xtable.size", NULL),
  hline.after = getOption("xtable.hline.after", c(-1,0,nrow(x))),
  NA.string = getOption("xtable.NA.string", ""),
  include.rownames = getOption("xtable.include.rownames", TRUE),
  include.colnames = getOption("xtable.include.colnames", TRUE),
  only.contents = getOption("xtable.only.contents", FALSE),
  add.to.row = getOption("xtable.add.to.row", NULL),
  sanitize.text.function = getOption("xtable.sanitize.text.function", NULL),
  sanitize.rownames.function = getOption("xtable.sanitize.rownames.function",
                                         sanitize.text.function),
  sanitize.colnames.function = getOption("xtable.sanitize.colnames.function",
                                         sanitize.text.function),
  math.style.negative = getOption("xtable.math.style.negative", FALSE),
  html.table.attributes = getOption("xtable.html.table.attributes", "border=1"),
  print.results = getOption("xtable.print.results", TRUE),
  format.args = getOption("xtable.format.args", NULL),
  rotate.rownames = getOption("xtable.rotate.rownames", FALSE),
  rotate.colnames = getOption("xtable.rotate.colnames", FALSE),
  booktabs = getOption("xtable.booktabs", FALSE),
  scalebox = getOption("xtable.scalebox", NULL),
  width = getOption("xtable.width", NULL),
  comment = getOption("xtable.comment", TRUE),
  timestamp = getOption("xtable.timestamp", date()),
  ...)
{
    ## If caption is length 2, treat the second value as the "short caption"
    caption <- attr(x, "caption", exact = TRUE)
    short.caption <- NULL
    if (!is.null(caption) && length(caption) > 1){
        short.caption <- caption[2]
  caption <- caption[1]
    }

    ## Claudio Agostinelli <claudio@unive.it> dated 2006-07-28 hline.after
    ## By default it print an \hline before and after the columns names
    ## independently they are printed or not and at the end of the table
    ## Old code that set hline.after should include c(-1, 0, nrow(x)) in the
    ## hline.after vector
    ## If you do not want any \hline inside the data, set hline.after to NULL
    ## PHEADER instead the string '\\hline\n' is used in the code
    ## Now hline.after counts how many time a position appear
    ## I left an automatic PHEADER in the longtable is this correct?

    ## Claudio Agostinelli <claudio@unive.it> dated 2006-07-28 include.rownames,
    ## include.colnames
    pos <- 0
    if (include.rownames) pos <- 1

    ## Claudio Agostinelli <claudio@unive.it> dated 2006-07-28
    ## hline.after checks
    if (any(hline.after < -1) | any(hline.after > nrow(x))) {
        stop("'hline.after' must be inside [-1, nrow(x)]")
    }

    ## Claudio Agostinelli <claudio@unive.it> dated 2006-07-28
    ## add.to.row checks
    if (!is.null(add.to.row)) {
        if (is.list(add.to.row) && length(add.to.row) == 2) {
            if (is.null(names(add.to.row))) {
                names(add.to.row) <- c('pos', 'command')
            } else if (any(sort(names(add.to.row))!= c('command', 'pos'))) {
                stop("the names of the elements of 'add.to.row' must be 'pos' and 'command'")
            }
            if (is.list(add.to.row$pos) && is.vector(add.to.row$command,
                                                     mode = 'character')) {
                if ((npos <- length(add.to.row$pos)) !=
                    length(add.to.row$command)) {
                    stop("the length of 'add.to.row$pos' must be equal to the length of 'add.to.row$command'")
                }
                if (any(unlist(add.to.row$pos) < -1) |
                    any(unlist(add.to.row$pos) > nrow(x))) {
                    stop("the values in add.to.row$pos must be inside the interval [-1, nrow(x)]")
                }
            } else {
                stop("the first argument ('pos') of 'add.to.row' must be a list, the second argument ('command') must be a vector of mode character")
            }
        } else {
            stop("'add.to.row' argument must be a list of length 2")
        }
    } else {
        add.to.row <- list(pos = list(),
                           command = vector(length = 0, mode = "character"))
        npos <- 0
    }

    ## Claudio Agostinelli <claudio@unive.it> dated 2006-07-28 add.to.row
    ## Add further commands at the end of rows
    if (type == "latex") {
        ## Original code before changes in version 1.6-1
        ## PHEADER <- "\\hline\n"

        ## booktabs code from Matthieu Stigler <matthieu.stigler@gmail.com>,
        ## 1 Feb 2012
        if(!booktabs){
            PHEADER <- "\\hline\n"
        } else {
            ## This code replaced to fix bug #2309, David Scott, 8 Jan 2014
            ## PHEADER <- ifelse(-1%in%hline.after, "\\toprule\n", "")
            ## if(0%in%hline.after) {
            ##     PHEADER <- c(PHEADER, "\\midrule\n")
            ## }
            ## if(nrow(x)%in%hline.after) {
            ##     PHEADER <- c(PHEADER, "\\bottomrule\n")
            ## }
            if (is.null(hline.after)){
                PHEADER <- ""
            } else {
                hline.after <- sort(hline.after)
                PHEADER <- rep("\\midrule\n", length(hline.after))
                if (hline.after[1] == -1) {
                    PHEADER[1] <- "\\toprule\n"
                }
                if (hline.after[length(hline.after)] == nrow(x)) {
                    PHEADER[length(hline.after)] <- "\\bottomrule\n"
                }
            }
        }
    } else {
        PHEADER <- ""
    }

        lastcol <- rep(" ", nrow(x)+2)
    if (!is.null(hline.after)) {
        ## booktabs change - Matthieu Stigler: fill the hline arguments
        ## separately, 1 Feb 2012
        ##
        ## Code before booktabs change was:
        ##    add.to.row$pos[[npos+1]] <- hline.after

        if (!booktabs){
            add.to.row$pos[[npos+1]] <- hline.after
            } else {
            for(i in 1:length(hline.after)) {
                add.to.row$pos[[npos+i]] <- hline.after[i]
            }
        }
        add.to.row$command <- c(add.to.row$command, PHEADER)
    }

    if ( length(add.to.row$command) > 0 ) {
        for (i in 1:length(add.to.row$command)) {
            addpos <- add.to.row$pos[[i]]
            freq <- table(addpos)
            addpos <- unique(addpos)
            for (j in 1:length(addpos)) {
                lastcol[addpos[j]+2] <- paste(lastcol[addpos[j]+2],
                                              paste(rep(add.to.row$command[i],
                                                        freq[j]),
                                                sep = "", collapse = ""),
                                              sep = " ")
            }
        }
    }

    if (length(type)>1) stop("\"type\" must have length 1")
    type <- tolower(type)
    if (!all(!is.na(match(type, c("latex","html"))))) {
        stop("\"type\" must be in {\"latex\", \"html\"}")
    }
    ## Disabling the check on known floating environments as many users
    ## want to use additional environments.
    #    if (!all(!is.na(match(floating.environment,
    #                          c("table","table*","sidewaystable",
    #                            "margintable"))))) {
    #        stop("\"type\" must be in {\"table\", \"table*\", \"sidewaystable\", \"margintable\"}")
    #    }
    if (("margintable" %in% floating.environment)
        & (!is.null(table.placement))) {
        warning("margintable does not allow for table placement; setting table.placement to NULL")
        table.placement <- NULL
    }
    if (!is.null(table.placement) &&
        !all(!is.na(match(unlist(strsplit(table.placement,  split = "")),
                          c("H","h","t","b","p","!"))))) {
        stop("\"table.placement\" must contain only elements of {\"h\",\"t\",\"b\",\"p\",\"!\"}")
    }
    if (!all(!is.na(match(caption.placement, c("bottom","top"))))) {
        stop("\"caption.placement\" must be either {\"bottom\",\"top\"}")
    }

    if (type == "latex") {
        BCOMMENT <- "% "
        ECOMMENT <- "\n"
        ## See e-mail from "John S. Walker <jsw9c@uic.edu>" dated 5-19-2003
        ## regarding "texfloat"
        ## See e-mail form "Fernando Henrique Ferraz P. da Rosa"
        ## <academic@feferraz.net>" dated 10-28-2005 regarding "longtable"
        if ( tabular.environment == "longtable" & floating == TRUE ) {
            warning("Attempt to use \"longtable\" with floating = TRUE. Changing to FALSE.")
            floating <- FALSE
        }
        if ( floating == TRUE ) {
            ## See e-mail from "Pfaff, Bernhard <Bernhard.Pfaff@drkw.com>"
            ## dated 7-09-2003 regarding "suggestion for an amendment of
            ## the source"
            ## See e-mail from "Mitchell, David"
            ## <David.Mitchell@dotars.gov.au>" dated 2003-07-09 regarding
            ## "Additions to R xtable package"
            ## See e-mail from "Garbade, Sven"
            ## <Sven.Garbade@med.uni-heidelberg.de> dated 2006-05-22
            ## regarding the floating environment.
            BTABLE <- paste("\\begin{", floating.environment, "}",
                            ifelse(!is.null(table.placement),
                                   paste("[", table.placement, "]", sep = ""),
                                   ""), "\n", sep = "")
            if ( is.null(latex.environments) ||
                (length(latex.environments) == 0) ) {
                BENVIRONMENT <- ""
                EENVIRONMENT <- ""
            } else {
                BENVIRONMENT <- ""
                EENVIRONMENT <- ""
                if ("center" %in% latex.environments){
                    BENVIRONMENT <- paste(BENVIRONMENT, "\\centering\n",
                                          sep = "")
                }
                for (i in 1:length(latex.environments)) {
                    if (latex.environments[i] == "") next
                    if (latex.environments[i] != "center"){
                        BENVIRONMENT <- paste(BENVIRONMENT,
                                              "\\begin{", latex.environments[i],
                                              "}\n", sep = "")
                        EENVIRONMENT <- paste("\\end{", latex.environments[i],
                                              "}\n", EENVIRONMENT, sep = "")
                    }
                }
            }
            ETABLE <- paste("\\end{", floating.environment, "}\n", sep = "")
        } else {
            BTABLE <- ""
            ETABLE <- ""
            BENVIRONMENT <- ""
            EENVIRONMENT <- ""
        }

        tmp.index.start <- 1
        if ( ! include.rownames ) {
            while ( attr(x, "align", exact = TRUE)[tmp.index.start] == '|' )
                tmp.index.start <- tmp.index.start + 1
            tmp.index.start <- tmp.index.start + 1
        }
        ## Added "width" argument for use with "tabular*" or
        ## "tabularx" environments - CR, 7/2/12
        if (is.null(width)){
            WIDTH <-""
        } else if (is.element(tabular.environment,
                              c("tabular", "longtable"))){
            warning("Ignoring 'width' argument.  The 'tabular' and 'longtable'
                environments do not support a width specification.  Use another
                environment such as 'tabular*' or 'tabularx' to specify the width.")
            WIDTH <- ""
        } else {
            WIDTH <- paste("{", width, "}", sep = "")
        }

        BTABULAR <-
            paste("\\begin{", tabular.environment, "}",
                  WIDTH, "{",
                  paste(c(attr(x, "align",
                               exact = TRUE)[
                               tmp.index.start:length(attr(x, "align",
                                                           exact = TRUE))],
                          "}\n"),
                        sep = "", collapse = ""),
                  sep = "")

        ## fix 10-26-09 (robert.castelo@upf.edu) the following
        ## 'if' condition is added here to support
        ## a caption on the top of a longtable
        if (tabular.environment == "longtable" && caption.placement == "top") {
            if (is.null(short.caption)){
                BCAPTION <- "\\caption{"
            } else {
                BCAPTION <- paste("\\caption[", short.caption, "]{", sep = "")
            }
            ECAPTION <- "} \\\\ \n"
            if ((!is.null(caption)) && (type == "latex")) {
                BTABULAR <- paste(BTABULAR,  BCAPTION, caption, ECAPTION,
                                  sep = "")
            }
        }
        ## Claudio Agostinelli <claudio@unive.it> dated 2006-07-28
        ## add.to.row position -1
        BTABULAR <- paste(BTABULAR, lastcol[1], sep = "")
        ## the \hline at the end, if present, is set in full matrix
        ETABULAR <- paste("\\end{", tabular.environment, "}\n", sep = "")

        ## Add scalebox - CR, 7/2/12
        if (!is.null(scalebox)){
            BTABULAR <- paste("\\scalebox{", scalebox, "}{\n", BTABULAR,
                              sep = "")
            ETABULAR <- paste(ETABULAR, "}\n", sep = "")
        }

        ## BSIZE contributed by Benno <puetz@mpipsykl.mpg.de> in e-mail
        ## dated Wednesday, December 01, 2004
        if (is.null(size) || !is.character(size)) {
            BSIZE <- ""
            ESIZE <- ""
        } else {
            if(length(grep("^\\\\", size)) == 0){
                size <- paste("\\", size, sep = "")
            }
            BSIZE <- paste("{", size, "\n", sep = "")
            ESIZE <- "}\n"
        }
        BLABEL <- "\\label{"
        ELABEL <- "}\n"
        ## Added caption width (jeff.laake@nooa.gov)
            if(!is.null(caption.width)){
                BCAPTION <- paste("\\parbox{",caption.width,"}{",sep="")
                ECAPTION <- "}"
            } else {
                BCAPTION <- NULL
                ECAPTION <- NULL
            }
            if (is.null(short.caption)){
                   BCAPTION <- paste(BCAPTION,"\\caption{",sep="")
            } else {
                   BCAPTION <- paste(BCAPTION,"\\caption[", short.caption, "]{", sep="")
            }
        ECAPTION <- paste(ECAPTION,"} \n",sep="")
        BROW <- ""
        EROW <- " \\\\ \n"
        BTH <- ""
        ETH <- ""
        STH <- " & "
        BTD1 <- " & "
        BTD2 <- ""
        BTD3 <- ""
        ETD  <- ""
        ## Based on contribution from Jonathan Swinton
        ## <jonathan@swintons.net> in e-mail dated Wednesday, January 17, 2007
        sanitize <- function(str) {
            result <- str
            result <- gsub("\\\\", "SANITIZE.BACKSLASH", result)
            result <- gsub("$", "\\$", result, fixed = TRUE)
            result <- gsub(">", "$>$", result, fixed = TRUE)
            result <- gsub("<", "$<$", result, fixed = TRUE)
            result <- gsub("|", "$|$", result, fixed = TRUE)
            result <- gsub("{", "\\{", result, fixed = TRUE)
            result <- gsub("}", "\\}", result, fixed = TRUE)
            result <- gsub("%", "\\%", result, fixed = TRUE)
            result <- gsub("&", "\\&", result, fixed = TRUE)
            result <- gsub("_", "\\_", result, fixed = TRUE)
            result <- gsub("#", "\\#", result, fixed = TRUE)
            result <- gsub("^", "\\verb|^|", result, fixed = TRUE)
            result <- gsub("~", "\\~{}", result, fixed = TRUE)
            result <- gsub("SANITIZE.BACKSLASH", "$\\backslash$",
                           result, fixed = TRUE)
            return(result)
        }
        sanitize.numbers <- function(x) {
            result <- x
            if ( math.style.negative ) {
                ## Jake Bowers <jwbowers@illinois.edu> in e-mail
                ## from 2008-08-20 suggested disabling this feature to avoid
                ## problems with LaTeX's dcolumn package.
                ## by Florian Wickelmaier <florian.wickelmaier@uni-tuebingen.de>
                ## in e-mail from 2008-10-03 requested the ability to use the
                ## old behavior.
                for(i in 1:length(x)) {
                    result[i] <- gsub("-", "$-$", result[i], fixed = TRUE)
                }
            }
            return(result)
        }
        sanitize.final <- function(result) {
            return(result)
        }
    } else {
        BCOMMENT <- "<!-- "
        ECOMMENT <- " -->\n"
        BTABLE <- paste("<TABLE ", html.table.attributes, ">\n", sep = "")
        ETABLE <- "</TABLE>\n"
        BENVIRONMENT <- ""
        EENVIRONMENT <- ""
        BTABULAR <- ""
        ETABULAR <- ""
        BSIZE <- ""
        ESIZE <- ""
        BLABEL <- "<A NAME="
        ELABEL <- "></A>\n"
        BCAPTION <- paste("<CAPTION ALIGN=\"", caption.placement, "\"> ",
                          sep = "")
        ECAPTION <- " </CAPTION>\n"
        BROW <- "<TR>"
        EROW <- " </TR>\n"
        BTH <- " <TH> "
        ETH <- " </TH> "
        STH <- " </TH> <TH> "
        BTD1 <- " <TD align=\""
        align.tmp <- attr(x, "align", exact = TRUE)
        align.tmp <- align.tmp[align.tmp!="|"]
        BTD2 <- matrix(align.tmp[(2-pos):(ncol(x)+1)],
                       nrow = nrow(x), ncol = ncol(x)+pos, byrow = TRUE)
        ## Based on contribution from Jonathan Swinton <jonathan@swintons.net>
        ## in e-mail dated Wednesday, January 17, 2007
        BTD2[regexpr("^p", BTD2)>0] <- "left"
        BTD2[BTD2 == "r"] <- "right"
        BTD2[BTD2 == "l"] <- "left"
        BTD2[BTD2 == "c"] <- "center"
        BTD3 <- "\"> "
        ETD  <- " </TD>"
        sanitize <- function(str) {
            result <- str
            ## Changed as suggested in bug report #2795
            ## That is replacement of "&" is "&amp;"
            ## instead of previous "&amp" etc
            ## result <- gsub("&", "&amp ", result, fixed = TRUE)
            ## result <- gsub(">", "&gt ", result, fixed = TRUE)
            ## result <- gsub("<", "&lt ", result, fixed = TRUE)
            result <- gsub("&", "&amp;", result, fixed = TRUE)
            result <- gsub(">", "&gt;", result, fixed = TRUE)
            result <- gsub("<", "&lt;", result, fixed = TRUE)
            ## Kurt Hornik <Kurt.Hornik@wu-wien.ac.at> on 2006/10/05
            ## recommended not escaping underscores.
            ## result <- gsub("_", "\\_", result, fixed=TRUE)
            return(result)
        }
        sanitize.numbers <- function(x) {
            return(x)
        }
        sanitize.final <- function(result) {
            ## Suggested by Uwe Ligges <ligges@statistik.uni-dortmund.de>
            ## in e-mail dated 2005-07-30.
            result$text <- gsub("  *", " ",  result$text, fixed = TRUE)
            result$text <- gsub(' align="left"',  "", result$text,
                                fixed = TRUE)
            return(result)
        }
    }

    result <- string("", file = file, append = append)
    info <- R.Version()
    ## modified Claudio Agostinelli <claudio@unive.it> dated 2006-07-28
    ## to set automatically the package version
    if (comment){
        result <- result + BCOMMENT + type + " table generated in " +
              info$language + " " + info$major + "." + info$minor +
              " by xtable " +  packageDescription('xtable')$Version +
              " package" + ECOMMENT
        if (!is.null(timestamp)){
            result <- result + BCOMMENT + timestamp + ECOMMENT
        }
    }
    ## Claudio Agostinelli <claudio@unive.it> dated 2006-07-28 only.contents
    if (!only.contents) {
        result <- result + BTABLE
        result <- result + BENVIRONMENT
        if ( floating == TRUE ) {
            if ((!is.null(caption)) &&
                (type == "html" ||caption.placement == "top")) {
                result <- result + BCAPTION + caption + ECAPTION
            }
            if (!is.null(attr(x, "label", exact = TRUE)) &&
                (type == "latex" && caption.placement == "top")) {
                result <- result + BLABEL +
                          attr(x, "label", exact = TRUE) + ELABEL
            }
        }
        result <- result + BSIZE
        result <- result + BTABULAR
    }
    ## Claudio Agostinelli <claudio@unive.it> dated 2006-07-28
    ## include.colnames, include.rownames
    if (include.colnames) {
        result <- result + BROW + BTH
        if (include.rownames) {
            result <- result + STH
    }
        ## David G. Whiting in e-mail 2007-10-09
        if (is.null(sanitize.colnames.function)) {
            CNAMES <- sanitize(names(x))
    } else {
            CNAMES <- sanitize.colnames.function(names(x))
    }
        if (rotate.colnames) {
            ##added by Markus Loecher, 2009-11-16
            CNAMES <- paste("\\begin{sideways}", CNAMES, "\\end{sideways}")
    }
        result <- result + paste(CNAMES, collapse = STH)

        result <- result + ETH + EROW
    }

    cols <- matrix("", nrow = nrow(x), ncol = ncol(x)+pos)
    if (include.rownames) {
        ## David G. Whiting in e-mail 2007-10-09
        if (is.null(sanitize.rownames.function)) {
            RNAMES <- sanitize(row.names(x))
        } else {
            RNAMES <- sanitize.rownames.function(row.names(x))
        }
        if (rotate.rownames) {
            ##added by Markus Loecher, 2009-11-16
            RNAMES <- paste("\\begin{sideways}", RNAMES, "\\end{sideways}")
    }
    cols[, 1] <- RNAMES
    }

## Begin vectorizing the formatting code by Ian Fellows [ian@fellstat.com]
## 06 Dec 2011
##
##  disp <- function(y) {
##    if (is.factor(y)) {
##      y <- levels(y)[y]
##    }
##    if (is.list(y)) {
##      y <- unlist(y)
##    }
##    return(y)
##  }

    varying.digits <- is.matrix( attr( x, "digits", exact = TRUE ) )
    ## Code for letting "digits" be a matrix was provided by
    ## Arne Henningsen <ahenningsen@agric-econ.uni-kiel.de>
    ## in e-mail dated 2005-06-04.
    ##if( !varying.digits ) {
    ## modified Claudio Agostinelli <claudio@unive.it> dated 2006-07-28
    ##  attr(x,"digits") <- matrix( attr( x, "digits",exact=TRUE ),
    ## nrow = nrow(x), ncol = ncol(x)+1, byrow = TRUE )
    ##}
    for(i in 1:ncol(x)) {
    xcol <- x[, i]
    if(is.factor(xcol))
            xcol <- as.character(xcol)
    if(is.list(xcol))
            xcol <- sapply(xcol, unlist)
        ina <- is.na(xcol)
        is.numeric.column <- is.numeric(xcol)

    if(is.character(xcol)) {
            cols[, i+pos] <- xcol
    } else {
            if (is.null(format.args)){
                format.args <- list()
            }
            if (is.null(format.args$decimal.mark)){
                format.args$decimal.mark <- options()$OutDec
            }
            if(!varying.digits){
        curFormatArgs <-
                    c(list(
                      x = xcol,
                      format =
                      ifelse(attr(x, "digits", exact = TRUE )[i+1] < 0, "E",
                                   attr(x, "display", exact = TRUE )[i+1]),
                      digits = abs(attr(x, "digits", exact = TRUE )[i+1])),
                      format.args)
                cols[, i+pos] <- do.call("formatC", curFormatArgs)
            }else{
        for( j in 1:nrow( cols ) ) {
                    curFormatArgs <-
                        c(list(
                          x = xcol[j],
                          format =
                          ifelse(attr(x, "digits", exact = TRUE )[j, i+1] < 0,
                                 "E", attr(x, "display", exact = TRUE )[i+1]),
                          digits =
                          abs(attr(x, "digits", exact = TRUE )[j, i+1])),
                          format.args)
                    cols[j, i+pos] <- do.call("formatC", curFormatArgs)
        }
            }
    }
    ## End Ian Fellows changes

        if ( any(ina) ) cols[ina, i+pos] <- NA.string
        ## Based on contribution from Jonathan Swinton <jonathan@swintons.net>
        ## in e-mail dated Wednesday, January 17, 2007
        if ( is.numeric.column ) {
            cols[, i+pos] <- sanitize.numbers(cols[, i+pos])
        } else {
            if (is.null(sanitize.text.function)) {
                cols[, i+pos] <- sanitize(cols[, i+pos])
            } else {
                cols[, i+pos] <- sanitize.text.function(cols[, i+pos])
            }
        }
    }

    multiplier <- 5
    full <- matrix("", nrow = nrow(x), ncol = multiplier*(ncol(x)+pos)+2)
    full[, 1] <- BROW
    full[, multiplier*(0:(ncol(x)+pos-1))+2] <- BTD1
    full[, multiplier*(0:(ncol(x)+pos-1))+3] <- BTD2
    full[, multiplier*(0:(ncol(x)+pos-1))+4] <- BTD3
    full[, multiplier*(0:(ncol(x)+pos-1))+5] <- cols
    full[, multiplier*(0:(ncol(x)+pos-1))+6] <- ETD

    full[, multiplier*(ncol(x)+pos)+2] <- paste(EROW, lastcol[-(1:2)],
                                                sep = " ")

    if (type == "latex") full[, 2] <- ""
    result <- result + lastcol[2] + paste(t(full), collapse = "")
    if (!only.contents) {
        if (tabular.environment == "longtable") {
            ## booktabs change added the if() - 1 Feb 2012
            if(!booktabs) {
                result <- result + PHEADER
            }

            ## fix 10-27-09 Liviu Andronic (landronimirc@gmail.com) the
            ## following 'if' condition is inserted in order to avoid
            ## that bottom caption interferes with a top caption of a longtable
            if(caption.placement == "bottom"){
                if ((!is.null(caption)) && (type == "latex")) {
                    result <- result + BCAPTION + caption + ECAPTION
                }
            }
            if (!is.null(attr(x, "label", exact = TRUE))) {
                result <- result + BLABEL + attr(x, "label", exact = TRUE) +
                    ELABEL
            }
            ETABULAR <- "\\end{longtable}\n"
        }
        result <- result + ETABULAR
        result <- result + ESIZE
        if ( floating == TRUE ) {
            if ((!is.null(caption)) &&
                (type == "latex" && caption.placement == "bottom")) {
                result <- result + BCAPTION + caption + ECAPTION
            }
            if (!is.null(attr(x, "label", exact = TRUE)) &&
                caption.placement == "bottom") {
                result <- result + BLABEL + attr(x, "label", exact = TRUE) +
                    ELABEL
            }
        }
        result <- result + EENVIRONMENT
        result <- result + ETABLE
    }
    result <- sanitize.final(result)

    if (print.results){
    print(result)
    }

    return(invisible(result$text))
}
```