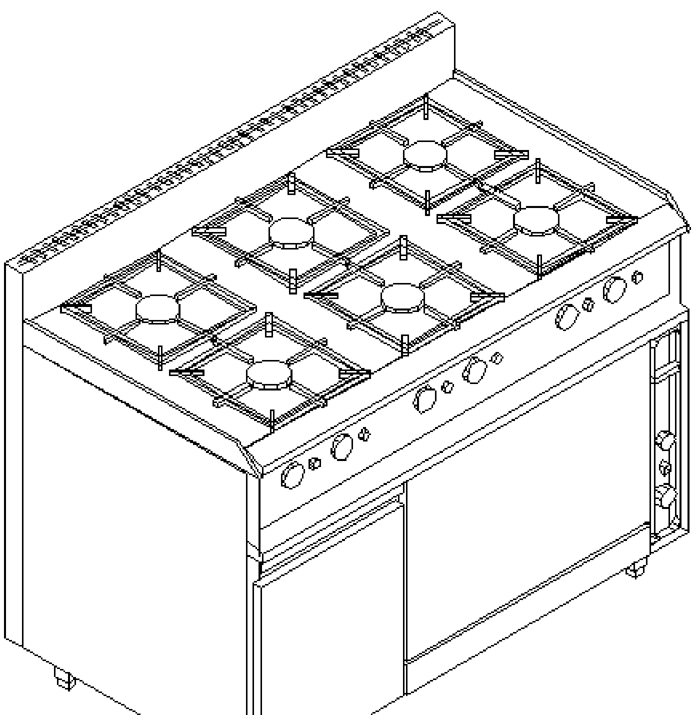
KCAD

USER MANUAL

ver. 7.0 for Windows 2000-XP

and AutoCAD 2007-2008-2009



REGISTERED MARKS

AutoCAD	Autodesk
Windows	Microsoft
Word	Microsoft
Excel	Microsoft
Access	Microsoft

GENERAL CONCEPTS

KCAD is an application suite made up of different application that exchange data, the first for layout design in AutoCAD environment, the second for managing discounts and changes in Windows environment, and others customized for each user, for the automated generation of documents such as offers or commercial reports in Microsoft Word or Microsoft Excel; besides, Microsoft Access is required to manage database files in MDB format.

The application has been conceived to facilitate the design of environments which must be furnished with items (kitchens, refrigerators, tables, etc.), chosen among different manufacturing houses lists. Providing all the necessary data, once the design is completed, the application can automatically compose the offer and other documents that depend on design, like commercial reports. Moreover, the application can automatically generate on the drawing the schema of all the necessary feed ears and discharge connections, with an automatic computing of respective consumptions.

To work correctly, KCAD needs to access all the data of objects used in the design. The objects are divided into lists, like the ordinary paper lists; in computer folders, they consist of a database containing all the alphanumeric information and a set of graphic files, collected in subfolders with other optional documents.

At installation time, a KCAD folder is created on disk, containing all the necessary files to make application suite work. Inside this folder, are created many subfolders, one for each available sample list, with corresponding database files and all associated graphic files.

To add your own lists, it is necessary to create a subfolder for each list, containing the database file and further subfolders representing the list managing divisions, containing graphic files and other support files, as described in detail in “SYSTEM MAINTENANCE”, later in this manual.

For the automatic compilation of documents related to a specific drawing, it is possible to use either Microsoft Word and Microsoft Excel, and every other application in Windows environment, that support Visual Basic Application edition or other development environment able to recall functions available in KCAD32.DLL library, that allow you to read all the data of all the objects placed in design drawing. These functions are documented in the appendix of this manual..

To modify and update the database files, it is necessary to use Microsoft Access.

The KCAD application suite contains the following applications:

1) An AutoCAD add-in application to aid kitchen and catering rooms design, named Kcad.arx, that allows to compose the design layout through the visual selection of the manufactors’ lists and their objects, and to export all the necessary data to other applications to generate all the documents related to the drawing.

2) A Windows application, named KCADWIN.exe, that allows to create files of data like those created by the AutoCAD add-in application, without the need of any drawing creation, through a similar visual-driven selection of lists and objects. This application can also manage existing file of data exported from drawings, in order to allow objects insertion, deletion or substitution, as long as the setting of discounts on objects and accessories through many logical criteria.

3) One or more applications for automated generation of documents related to the design, like offers or commercial reports, in Word or Excel format, that eventually integrate the data of a specific design with other data concerning the offer (date, customer name, etc.) and create the documents basing on one or more files extracted from the AutoCAD add-in or created by KcadWin Windows application. Inside the box of KCAD application suite, is provided a sample offer template that can easy customized by the user; generally these applications are customized to fit every user demand, but all of them are based on the use of functionalities available in KCAD32.DLL library.

INSTALLATION

Before to install the application, it’s necessary that AutoCAD is installed and started at least once on the target computer, making available to the Kcad setup application some settings that are completed only after the first AutoCAD start. If there are multiple installations of AutoCAD, it’s necessary to run and close the AutoCAD version corresponding to Kcad version before starting Kcad setup.

To install Kcad, you need to have administration right on the target computer. After ypu have start and closed the corresponding AutoCAD version, you must run the SETUP.EXE application on installation CD, or in the folder where you copied all the installation files. The setup application detects the AutoCAD presence, checks its version for Kcad version compatibility, and asks you to confirm the destination folder name, usually “C:\Program files\Disicad\Kcad2007”. Then asks you if you want to install the sample generical furniture list. If some files result already present on disk, the application display a message and let the user avoid to overwrite it, as they maybe customized by the user, if you’re repeating the Kcad installation.

Once the installation is over, it will be available a new menu voice under “Start”->”All programs”->”Disicad”, containing the two items “Kcad 2007” and “Kcadwin 2007”.

Selecting the first item, AutoCAD is started with all the Kcad features enabled. At the first execution, the application displays a form with the application authorization request and other registration data, and an email address where to send the request. We suggest to copy all the window’s content, and paste it in the sending email. Once received back the authorization code, you must copy it in the corresponding edit box and close the window pressing OK. You must have administration right to allow the application write the authorization code in a part of Windows registry accessible to all the users.

Selecting the second item, the Windows application “Kcadwin” is started, that allows to manage the file of type “Kcad report” with “KCR” extension, that contains the data exported from AutoCAD drawings, or may be created by this application without the need for any existing drawing file. This application allows you to edit all the data in that files, and to set any kind of discount, using various criteria.

If you install your own lists, different from the sample generical one given with Kcad suite, you should keep them in a dedicated folder, other than that used in KCAD installation; in this case, you must set the environment variable “KCADDATA” with the full path to the lists dedicated folder. The environment variables are set in the dialog box accessible through the Control Panel, selecting the “System” item, then the “Advanced” tab, and then click the “Environment variables” command button. In the appearing dialog box, it’s possible to create both “user variables” and “system variables”; because the system variables are available to all users, you must have administration rights on computer to add “KCADDATA” on this variables group, but in this case the setting will be valid for all computer users.

For sake of simplicity, it’s possible to create and use the folder “C:\KCADDATA”, without the need of setting any environment variable. The application logic to search for lists folder is the following: If the environment variable “KCADDATA” is set, the specified folder is used. If the environment variable does not exists, the “C:\KCADDATA” folder is used. If the folder does not exists, the Kcad application installation folder is searched for data lists.

Note: Unlike the previous versions, it is no longer used the environment variable "KCADPATH" to specify an installation folder other than the default. Now the path to the application files is stored in the system registry when you install Kcad.

Among the application files is present an Excel template file with “.xlt” extension, to automatically produce a sample offer document, selecting a data file extracted from the AutoCAD drawing or created with Kcadwin application. In all the Excel documents created with this model, there is a command button that allows you to select one or more Kcad data files, and compile the offer template reporting all the contained data. This file is provided for demonstration purposes, but can be easily customized, and the contained Visual Basic code can be taken as an example to create your own automated templates, using Excel, Word or any other application that supports Visual Basic for Application.

AUTOCAD ADD-IN COMMANDS DESCRIPTION

The menu "KCAD" added to AutoCAD with Kcad.arx when loading the application contains all the commands described below. The commands contained also in the KCAD toolbar KCAD have the button image on the right.

KCAD COMMAND

Besides being the application name, KCAD is also the name of the command that displays the list of all commands added in AutoCAD, with a brief description of their use; it also allows you to set some configuration variables that are stored in KCAD.CFG file to be loaded automatically each time the program runs.

The list of the commands available, with their short description, is similar to the one reproduced below; if you do not remember the exact name or function of a command, simply call this command and read the displayed list. The name KCAD for this command has been chosen to be easily remembered by the operator. The listed names are the effective command names (what you should write at AutoCAD prompt to call them).

KC_LISTS	Select the current object list
KC_OBJECT	Place an object from current list
KC_GLANCE	Display view of selected object
KC_ACCESSORIES	Set object accessories
KC_MODIFICATIONS	Modify object features
KC_MARK	Object numeration and grouping
KC_NUMBER	Modify object number
KC_REPORT	Make KCR report
KC_TABLES	Place tables of inserted objects
KC_PLUGS	Display plug symbols
KC_LABEL	Label displayed plug symbols
KC_SOUND	Enable / disable feedback sound
KCAD	Program setting and commands list
KCAD3D	Trasform objects from 2d to 3d
KCAD2D	Trasform objects from 3d to 2d
KC_BULKHEAD	Calculates mq of bulkheads and cold stores
KC_DISCOUNT	Load discounts from existing KCR report
KC_CONSUMPREVIEW	Consumption totals preview
KCADINFO	Informations about KCAD current version

After printing the commands list, the program goes on asking if you want to edit the configuration parameters; if so, asks for all settings of configuration variables, displaying the current value, that is proposed as the default answer; if you do not want to change any setting, simply answer pressing ENTER to their requests (thus accepting the default answer) or leave the command pressing ESC. In this case no modification eventually set in the previous questions will be saved (this can be useful if you have answered incorrectly to a previous question).

The configuration variables are the following:

Name of the text style used in the production of the labels and the tables: It must be the name of an AutoCAD style. No matter if that the style exists at the moment of the setting; if it does not exist at the time of use, the program displays a warning message and uses the STANDARD style. This style is also used if the configuration has not been made or the KCAD.CFG file has not been found.

Texts height (in centimetres): Height used in the production of labels and tables texts, that must be a positive number. The default value is 0,25 cm, if the configuration has not been made or the KCAD.CFG file has not been found.

Use uppercase: The labels and tables texts are usually placed in the drawing with the same capitalization used when they were stored in the database files. However, setting this variable to 1, you can force the conversion in uppercase of all the texts (the columns header text in the tables are not affected by this setting).

Absolute text height: Usually, the texts height mentioned above is sensitive to the scale as all the KCAD symbology; however, if you want to specify the height in an absolute way that should not be multiplied by the current scale, you can set this variable to 1.

Name of the layer used for summary tables. The summary tables can be positioned in a dedicated layer. The default name is "TABLES".

Name of the block used for the marks: As the balloons style can change depending on the density of the objects in the drawing, the block used by the program can be modified according to the different needs, simply setting the name of the block in this variable.

The name of the shown block should not be preceded by the path (it must necessarily be present in the SYMBOLS subfolder of KCAD) nor should it be followed by the extension (which is always DWG). The default name is MARCA_1S and the corresponding block is supplied with KCAD.

The balloon dimension necessarily depends by the scale, like all the other symbols; but the text style and height of the number inside the balloon are not affected by the previous settings, as they are defined at the creation time of the block.

If you want to create a block to use with the balloons, remember the following specifications:

1) The block name must have “1S” as last characters, and must not have more than eight characters.

2) The block must be drawn using the centimeter as units (one unit in AutoCAD will be one centimeter on the final plotted drawing).

3) The block must have only one attribute, not constant, not invisible, without verify, and with a preset value as null string; that necessary because KCAD puts first the block without specifying any attribute value, and afterwards it changes the attribute value with the correct balloon number: the block insertion must not be interrupted by the request for an attribute value.

Default x distance of the balloons: The balloons are automatically inserted near the marked object, in a default position based on object insertion point and inclination; with this variable and the next one, you can set a different relative baloon position from insertion point of the object.

Default y distance of the balloons: With this variable and the previous one, you can set the relative balloon position from the insertion point of the object.

Mark additional scale: The balloons blocks are scaled as all other symbols; in some drawings, however, it is handly to be able to set their size indipendently. Setting this value differently from 1.0, you can easily obtain the desired size.

Plugs additional scale: As the mark additional scale, this command lets you change the size of plugs symbols. Only the plugs represented with symbols blocks respond to this scale; the plugs represented with real blocks are not affected by this scale (neither by the general symbols scale; they are affected only to the absolute objects' scale; see the explanation of the PLUGS command for further information).

Language: The summary table reproduced in the drawing are in English, but they can be in Italian or in French language, according to this variable.

Tables translation: If the language is different from English, is possible to produce the summary table with a double description for each item, showing the foreign and the English descriptions.

Using blocks as table header. The summary tables can use as column header a block for each column provided in the language set for the table, and possibly also with double language. If this flag is set to 0, column headers are generated as text, otherwise use they use the "HDR *" blocks in the "Symbols" subfolder of the Kcad installation folder.

Consumption calculation prompt: If some technological plug is present, the summary tables are produced with the consumption calculation. Setting this variable to 1 will cause you to be prompted each time if you want or not the consumpion calculation on the table, if any technological plug is displayed in the drawing.

Accessories prompt: When you place objects in the drawing, if an object can have some accessories, you'll be prompted to choice someone. You can use this setting to enable or disable this feature.

Enable prices display. This flag can be set to zero so that, during the selection of objects, the information displayed will not show the list price. Once the display of prices has been disabled, to set it again to 1 requires a password, which corresponds to the same authorization code of Kcad.

Enable the display of prices. This flag can be set to zero so that, during the selection of objects, the information displayed to be shown the list price. Once the display of prices has been disabled, by setting it to 1 again requires a password, which corresponds to the same authorization code of Kcad.

Once at this point, the previous settings are stored in the KCAD.CFG configuration file. The following settings are not stored in the configuration file, but are stored only in the current drawing:

Relative blocks scale: As sometimes you have to complete a drawing already done with a certain scale, you can act on this variable to change the scale of the objects yet to be placed. This setting is stored in the AutoCAD system variable USERR2.

Relative symbol scale: Similarly to previous one, this variable affect the symbology scale (balloons and plugs) to be placed ina drawing already done. This setting is stored in the AutoCAD system variable USERR1.

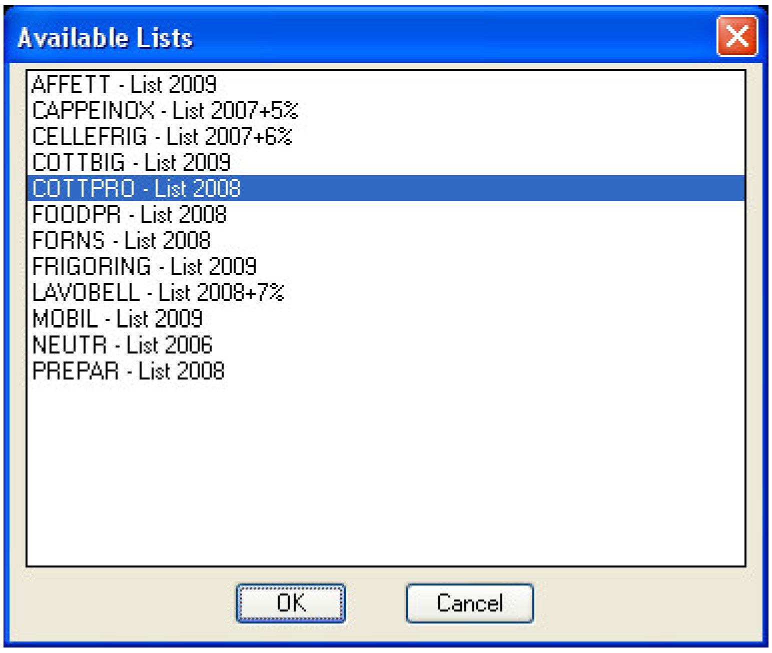
Measure unit: Millimeters, Centimeters, meTers. If the drawing is not in centimeters, it is necessary to set this variable in the right way to calculate the bulkheads' size. This setting is stored in the AutoCAD system variable USERI1.

LISTS COMMAND

This command allows you to set the furniture list from where you want to choose items to be placed with the command OBJECT described below. KCAD is supplied together with only a generic list, but is designed so you can add and use more furniture lists in a single drawing.

The way to create a new list and add it to the database is described in SYSTEM MAINTENANCE. Once added, the list name will appear listed along with the names of existing lists.

The command displays a dialogue box with a complete list of the available forniture lists; selecting the name of the wanted list and pressing ENTER (or clicking the **OK** button), you set the list as current list, the one from which items are selected when using the OBJECT command.


Available Lists

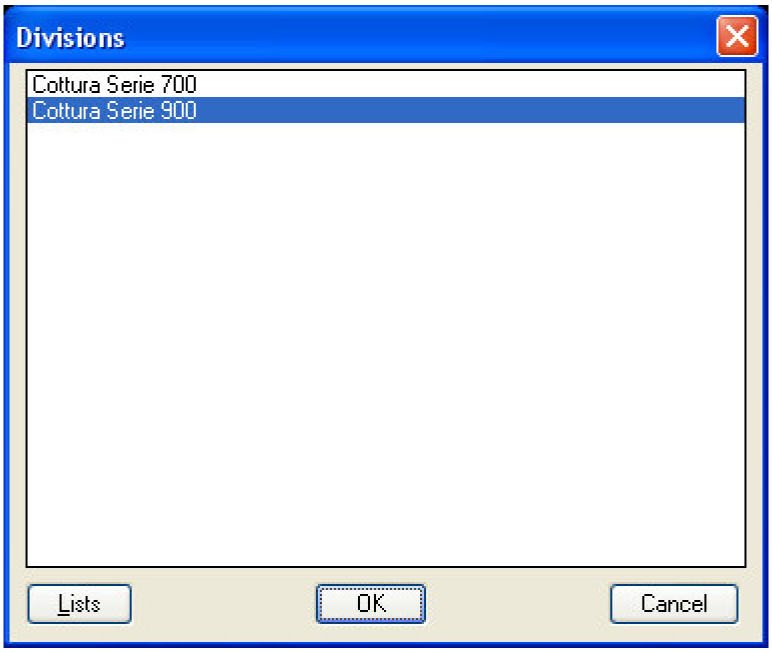
AFFETT - List 2009
CAPPEINOX - List 2007+5%
CELLEFRIG - List 2007+6%
COTTBIG - List 2009
COTTPRO - List 2008
FOODPR - List 2008
FORNS - List 2008
FRIGORING - List 2009
LAVOBELL - List 2008+7%
MOBIL - List 2009
NEUTR - List 2006
PREPAR - List 2008

This command can be also invoked through a button in the dialog box of the division choice of furniture list (see the OBJECT command below). Moreover, it is automatically invoked if the OBJECT command is invoked without having chosen the current list yet.

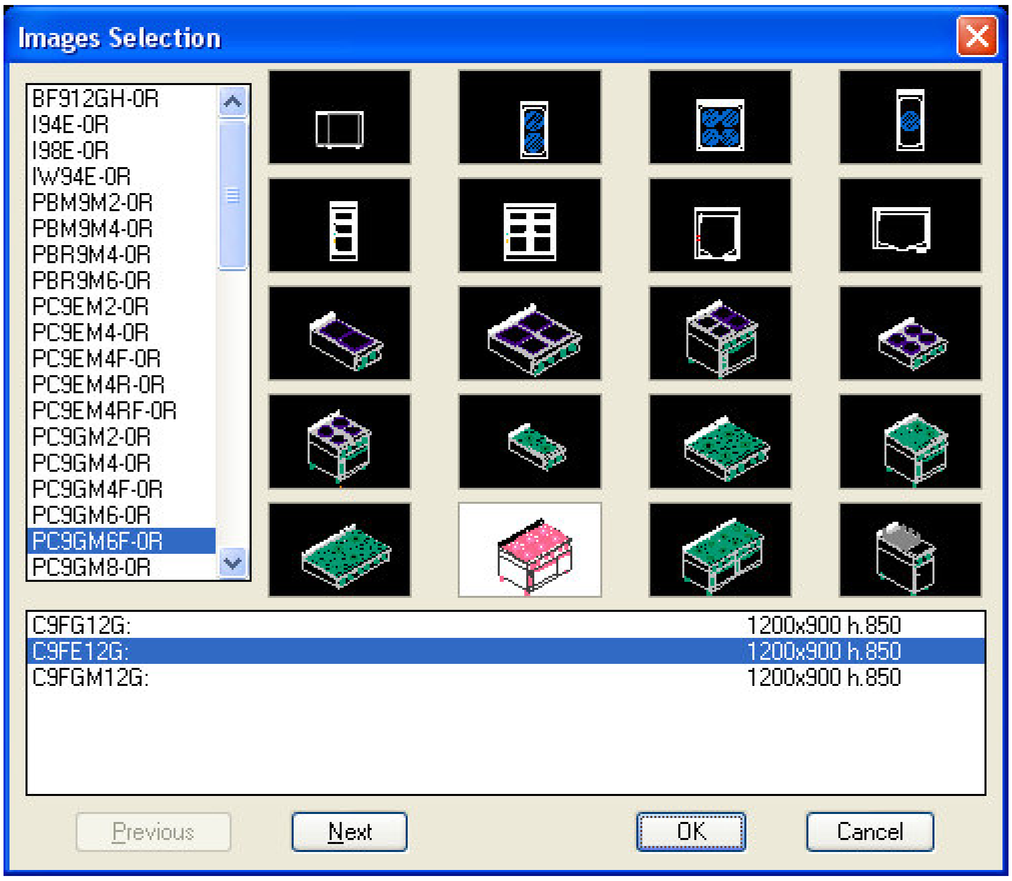
OBJECT COMMAND

With this command, you can choose an object among the ones contained in the current list and insert it in the drawing, eventually with its accessories. This choice is guided through a series of dialog boxes, whose number is varying according to the characteristics of the object.

First of all, if no list has been set yet as the current list, the previous command is automatically invoked; then the command goes on displaying the dialog box for choosing the desired list division. The divisions correspond to those you can usually find in the paper lists, with generic names such as REFRIGERATORS, REFRIGERATED CABINET or divisions of products for series. These divisions have been made because they allow you to manage smaller groups of objects than a complete list, and help to keep order during the system maintenance process. In this dialog box you can find a button to invoke directly the LISTS command described above.



Once the division is chosen, is displayed a dialog box with the images of the objects and their names. The images are not always represent a unique object; in many cases, the same image references more objects with similar appearance but different in other characteristics. In this dialog box, are represented all the different images in the current the division objects, displayed, when necessary, over several consecutive pages. This step corresponds to the visual search of an object on a paper list.



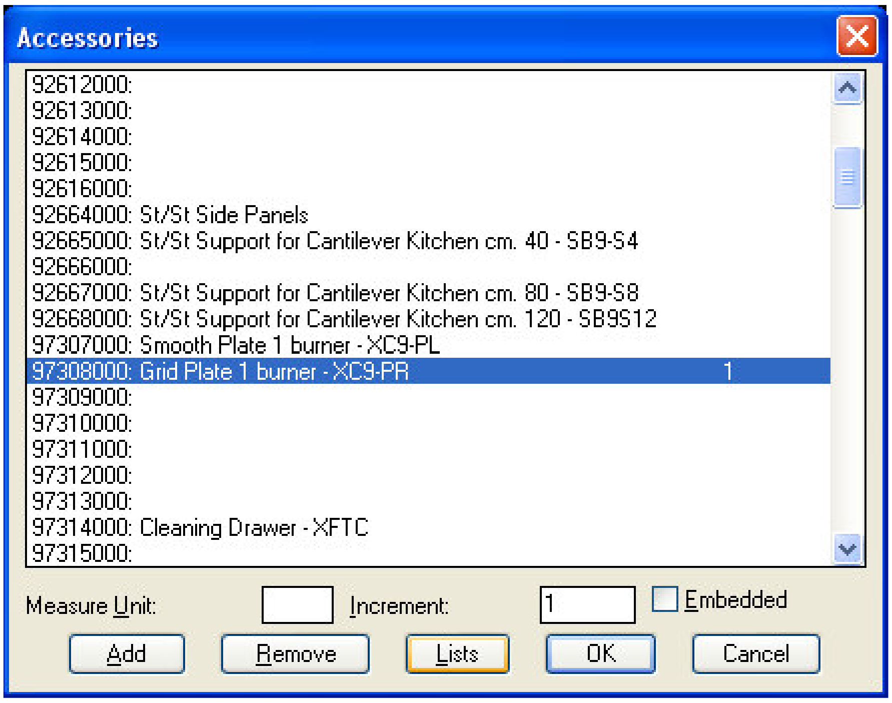
BF912GH-0R
I94E-0R
I98E-0R
IW94E-0R
PBM9M2-0R
PBM9M4-0R
PBR9M4-0R
PBR9M6-0R
PC9EM2-0R
PC9EM4-0R
PC9EM4F-0R
PC9EM4R-0R
PC9EM4RF-0R
PC9GM2-0R
PC9GM4-0R
PC9GM4F-0R
PC9GM6-0R
PC9GM6F-0R
PC9GM8-0R

C9FG12G:	1200x900 h.850
C9FE12G:	1200x900 h.850
C9FGM12G:	1200x900 h.850

After selecting the desired image of an object, one o more short descriptions are displayed in the bottom part of the dialog box. These allow you to see the differences among items that refer to the same image, with the corresponding item code. Through this selection, you can make the final choice of the object to be inserted in the drawing.

Some objects can have one or more accessories collected in accessories lists. These lists have a different structure than objects lists, as they have to contain a smaller amount of data. Each list of accessories is made of a table with code, description and price of each accessory, without any division. The list of accessories can be contained in the same database of a list of objects, or can be contained in a dedicated database stored in a subfolder named "Accessories" of KCAD data folder.

If an object can have accessories, among its data is also specified a default accessories list name. In this case, after you made the final choice of the object, a further dialog box is displayed to let you choose the accessories to be added to the object. The dialog box lists all the accessories available in the default accessories list, so you can choose which accessor you want to add or remove to the object. This feature of automatic displaying of the accessories dialog box can be disabled by setting properly the parameter “Accessories prompt” among the configuration variables displayed with the “KCAD” command. In any case, this window remains accessible through the command “ACCESSORIES” described below.


Accessories

```
92612000:
92613000:
92614000:
92615000:
92616000:
92664000: St/St Side Panels
92665000: St/St Support for Cantilever Kitchen cm. 40 - SB9-S4
92666000:
92667000: St/St Support for Cantilever Kitchen cm. 80 - SB9-S8
92668000: St/St Support for Cantilever Kitchen cm. 120 - SB9S12
97307000: Smooth Plate 1 burner - XC9-PL
97308000: Grid Plate 1 burner - XC9-PR                         1
97309000:
97310000:
97311000:
97312000:
97313000:
97314000: Cleaning Drawer - XFTC
97315000:
```

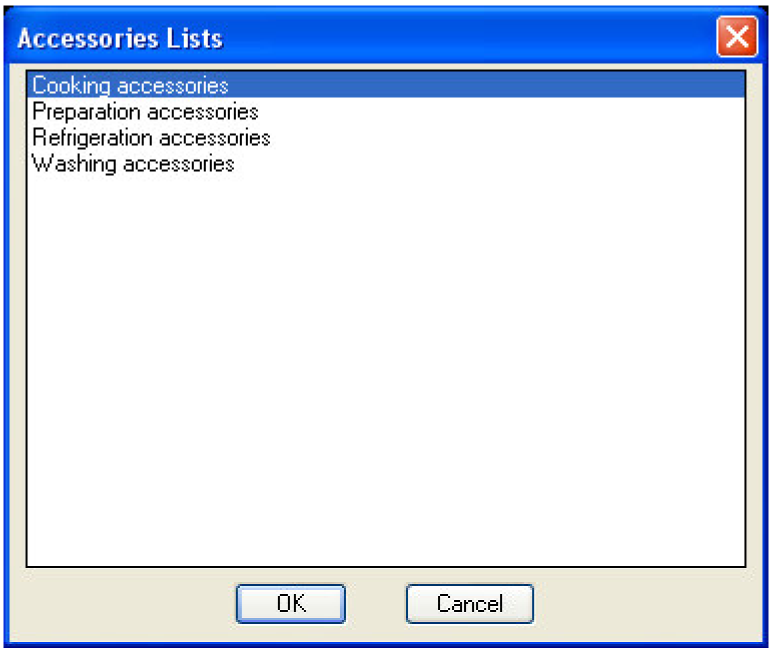
Measure Unit: Increment: 1 ☐ Embedded

Selecting the **Add** button or by double-clicking on an accessory item listed, or by pressing ENTER when is selected the accessory you want, it is added to the object, using the unit of measure and the amount of increase shown in their text boxes.

Selecting the **Remove** button, the selected accessory is removed from the object according to the quantity specified in the **increment** edit box. Repeating the same operations more times, the number that shows the quantity of the accessory is increased or decreased.

To add or remove more than one accessory at a time, simply indicate the quantity desired in the **Increment** edit box: the accessory increase and decrease operations will be made each time with the number indicated.

Selecting the button **Lists**, it is displayed a dialog box with a list with all the available accessories lists, allowing you to change the current accessories list, and to add more accessories from different accessories lists to the same object.



Cooking accessories
Preparation accessories
Refrigeration accessories
Washing accessories

For each accessory, you can specify the **measure unit** for its quantity, thus making possible to specify, for example, square metres or kilograms or other measure units. For this reason, the quantity of each accessor do not need to be an integer like the object quantity do, but can be instead a floating point number,.

The option button **Embedded** allows you to classify an accessory so that its price and its description are not specified separately after the object to which they belong in the offer, as it usually done, but so that the description is embedded with the items of the object description and the accessor price is directly added to the original object price, without appearing as a separate item with its own price. Actually, this is simply a flag, and the implementation of this functionality must be provided by the custom VBA program in Word or Excel; by the way, is useful to set this option at design time, specially if the selected accessory should be treated differently from all other accessories.

After all these choices, the block that has to be inserted is dynamically displayed and the yuser is prompted for the insert point and the inclination angle. The inserted block is the same for all the objects that share the same image, but it contains all the other information that can distinguish it from other objects, including any accessories.

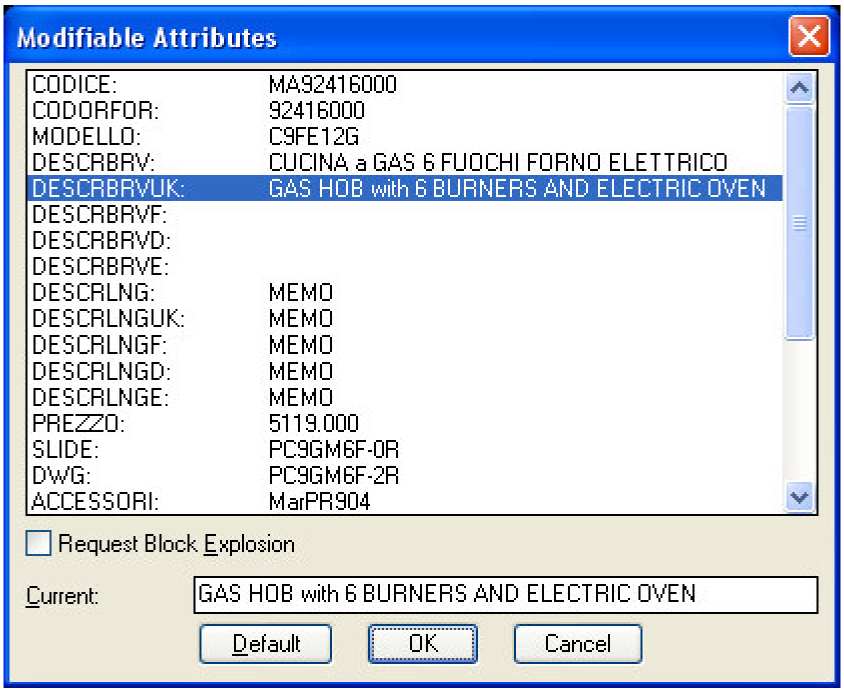
ACCESSORIES COMMAND

If an object provides accessories, their assignment can be modified also after its drawing insertion using this command, which prompts you to select the object to which you want to change the accessories, and displays the accessories dialog box, showing the current setting for each accessor, which can be modified as you want, as described for the accessory setting at the insertion time. Before displaying the dialog box, on AutoCAD text screen are printed the names of the lists from where some accessory has been choosen, allowing the user to control all the used accessory lists, and select them in order to check the accessories selected in each one. If the original object provides no accessory list, the message ”No accessory list set”, is printed on the screen, and the dialog box of the accessories lists choice is displayed.

MODIFICATIONS COMMAND

This command allows you to set some characteristics as different from the standard for special objects. Usually, all the characteristics of an object are described in the list database to which the object belongs, but sometimes it is possible that a particular object should be modified in a exceptional way.

This command prompts for an object selection, and display all the data associated with it in the database, and allows you to change the field values where the changes is possible. Eventually you can even explode the object, to allow some graphic changes. All the changed characteristics are stored within the block reference that represents the object on the drawing, and are valid only for that single reference: they never change the list database, nor the data of other similar objects eventually present in the same drawing. If you want to change the data in the database, do not use this command, but follow the instructions in “SYSTEM MAINTENANCE”.



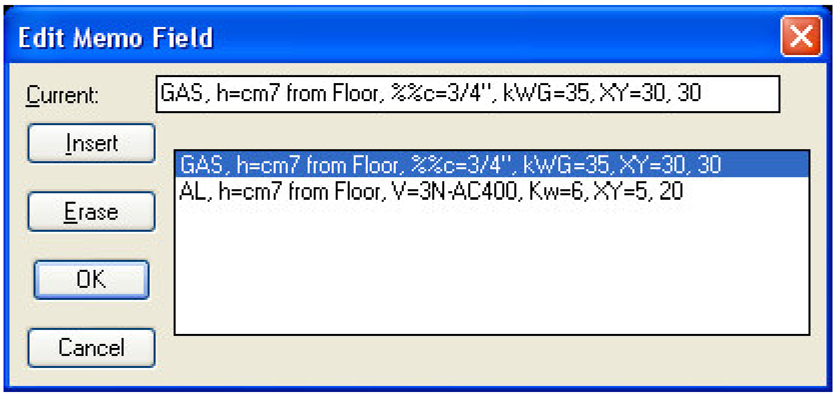
CODICE:	MA92416000
CODORFOR:	92416000
MODELLO:	C9FE12G
DESCRBRV:	CUCINA a GAS 6 FUOCHI FORNO ELETTRICO
DESCRBRVUK:	GAS HOB with 6 BURNERS AND ELECTRIC OVEN
DESCRBRVF:	
DESCRBRVD:	
DESCRBRVE:	
DESCRLNG:	MEMO
DESCRLNGUK:	MEMO
DESCRLNGF:	MEMO
DESCRLNGD:	MEMO
DESCRLNGE:	MEMO
PREZZO:	5119.000
SLIDE:	PC9GM6F-0R
DWG:	PC9GM6F-2R
ACCESSORI:	MarPR904

Request Block Explosion

Current: GAS HOB with 6 BURNERS AND ELECTRIC OVEN

To edit a characteristic, you must select it in the list; if the characteristic can be changed, it will be copied in the editation field at the bottom of the dialog box where it can be changed. To make the changes accepted, you must to finish the editing by pressing ENTER again: the selected field in the list will be updated with the new content.

If the field type is MEMO (that has a content that requires multiple lines to be displayed), instead of using the editing field at the bottom of the dialog box, it's opened a special dialog box that displays the multiple lines of the field one below the other. In a similar way to that for the modification of a single field, you must select the line you want to change, press ENTER so that it will be copied in the top editing field, edit the line and finish pressing ENTER again, so the changes will be accepted. You can also insert or delete a line with the **Insert** and **Erase** buttons, while to add a new line at the last position (or to add the first line if the memo field is empty), just move to the last line (that is alwais blank) and press ENTER, without using the **Insert** button, which is used only to insert lines between those already existing. To accept all the changes to the memo field, return to the previous dialog box selecting **OK**, otherwise use the **Cancel** button (in this case, the memo field will not be changed).



Current: GAS, h=cm7 from Floor, %%c=3/4", kWG=35, XY=30, 30

GAS, h=cm7 from Floor, %%c=3/4", kWG=35, XY=30, 30
AL, h=cm7 from Floor, V=3N-AC400, Kw=6, XY=5, 20

The modified fields are indicated by an asterisk placed after the entry in the list. If you want to remove a modification to restore the value stored in the database, simply press the **Default** button after selecting the desired item.

If the block explosion has been requested, the original block reference is exploded and is inserted in the same position the symbolic block named NUCLEO1S, placed on the NUCLEI layer (that can be hidden for plotting). This is necessary because all the information of each object is stored with the block that represents it, which also contains the name of the list database to which it belongs, to recover additional information: if the block is exploded this information is lost. For this reason, if it's necessary only a graphic change of an object block reference, you must never use directly the AutoCAD EXPLODE command, but you must always use this one, which can be used without setting other changes than the block explosion flag.

To accept all the changes set, exit the dialog box selecting **OK**, otherwise use the **Cancel** button, to not make any change.

MARK COMMAND

Once you place all the objects in the drawing, they must to be grouped and numbered to allow the automatic production of summary tables and the data files that allows the automatic compilation of offers and other documents. The summary table is considered to consist of a number of section tables that group the objects of the same environment, which are supposed to be listed together and marked with consecutive numbers. This command requests to select the objects you want to grouped into a single section table of the summary table, then displays a dialog box that allows you to specify the name of the section table (e.g. "VEGETABLES PREPARATION" or "COOKING DEPARTMENT"), and the first number from which to start the automatic numbering.

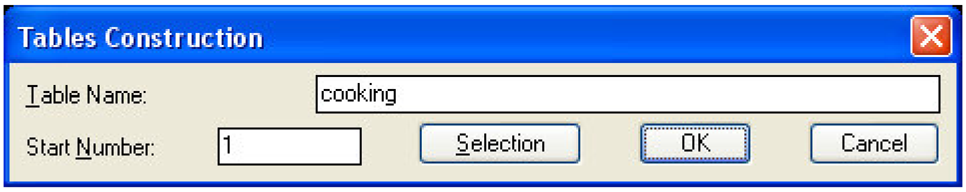

Tables Construction

Table Name: cooking

Start Number: 1

Next to each selected object, following the selection order (or the insertion order if more objects have been selected together using a selection box), it is inserted a numbered baloon (namely, the block specified in the configuration set with the KCAD command described above). The block can then be moved with the standard AutoCAD commands, but must not be exploded, otherwise would be lost the data necessary to identify the linked object. The block scale is influenced by both the overall scale of all the symbols, and by the markings particular scale, if set in configuration with the KCAD command.

If you want to add some objects to an existing section table, you can select an object that already belongs to the desired section table, after pressing the

"**Selection**" button: in this case, the name of the section table will be copied in the corresponding edit field of the dialog box.

The proposed **start number** is simply the next following the last one used before by the same function: the command does not check about number uniqueness, to let you free to manage them easily. You specify a decimal number rather than an integer number. In this case, as the function gradually increases the starting number while it creates the balloons of the selected objects, the increse will not be of a one unit but adjusted to the number of decimals in the given number.

NUMBERS COMMAND

After creating the section tables with the above command, you can change the numbers automatically assigned using this command, that requests to select an object or its number, shows the name of the table to which it belongs and ask for the new number. Again, no checks are made on the uniqueness of the number in the drawing, nor on the numbering sequence through the different section tables.

If you have a set of identical objects, you can assign them the same number. Since this function works only on already numbered objects, you must first assign a number to the objects with the command MARKS, and then change it with this command.

The displayed dialog box is the same as the MARKS command but the specified number in the "Starting number" edit box is used for all the selected objects. If the objects are different, nothing is notified by this conmand, but when you will generate the report these differences will be detected and it will be displayed an error message showing the mark number.

By specifying zero as new number, you get the object unlink from the section table: the number disappears and the object can be assigned to a different section table, using again the MARKS command described above. By the way, you can also erase the number directly with AutoCAD commands and select again the object to mark it with a different number in a different section table.

PLUGS COMMAND

Many objects need feed and discharge plugs which must be considered when generating the plugs map. The command PLUGS examines all the present objects, determines the plugs that each object provides and inserts the corresponding symbol blocks in the drawing, linked to the object in the same way used for the markings, to allow the user to move the blocks with the normal AutoCAD commands without losing reference to the object.

The plug symbol insertion point may be defined with an attribute of block that represents the object, or specified with other data about the plug in the list database; in absence of any specification, the placement is assigned automatically, inserting the different plug symbols from left to right, starting near the insertion point of the object. For more details on how to define the insert point, see “SYSTEM MAINTENANCE”.

During the design, this control can be invoked multiple times, and each time it will add the plugs not present yet in the drawing, acting only on the objects placed since the last call, and those whose plugs have beeen deleted. Since is not automatically deleted any plug removed from PLUGS database field through the MODIFICATION command, eventually you should delete the existing object plugs of the object using the normal AutoCAD commands, and then regenerate it again using this command.

Usually, the blocks used to represent plugs are symbols, that means that their dimensions are not related to the object physical dimensions to which they belong, but rather to those of the texts and other symbolic elements in the drawing. In this case, their block name must end with the two characters “1S”.

However, are considered plugs as well also the inputs and outputs of air hoods, and in this case their dimensions depend on the objects general scale, rather than symbols scale; the name of the blocks representing these plugs must have “1R” as the last two characters. This type of plugs is not affected by the symbols scale nor by the plugs supplementary scale; however, the X and Y dimensions can be set in an absolute way adding the variables DX=*<length in metres>* and DY=*<width in metres>* in the plugs field of object database (see SYSTEM MAINTENANCE for details).

LABELS COMMAND

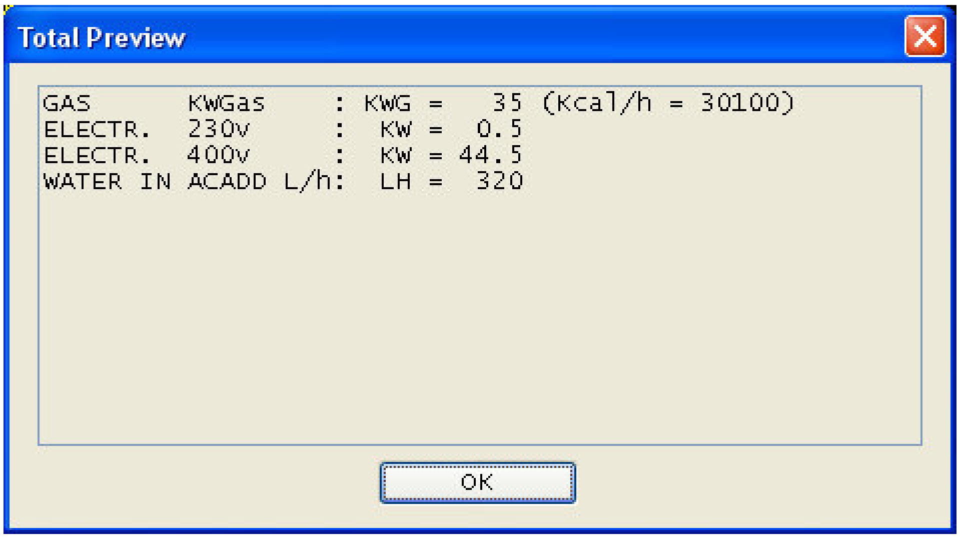
Every plug must be labeled accurately to report all the necessary data; using this command, simply by selecting an object plug symbol, the related plug label is automatically generated, consulting the data in the database. The format of plugs data stored in database field is of type “XXX=YYY”, where XXX is the symbol of a unit of measure and YYY is the value associated with it (for more information about plugs data, see SYSTEM MAINTENANCE).

After having selected the plug you want to label, the command prompts for a sequence of points to form a broken line that begin at the plug symbol and ends to its label; to stop the sequence press ENTER, and the label will be generated on the left or on the right of the last given point, according to the inclination of the last drawn line segment.

If the block that represents the plug is made of a circle with the origin in the insert point, the tip of the arrow will be drawn from the circle edge; if it has a rectangular outer perimeter, the arrow will start from the nearest linear edge; otherwise, it will start from the block insert point (this can happen also when the circle has not its centre exactly coinciding with the block insert point).

CONSUMPTIONPREVIEW COMMAND

It is important to determine at design time the overall consumption of all the objects inserted in the drawing. With this command you can select one or more objects, or even the whole drawing, and check in the displayed dialog box the total of all types of consumption, which are summed taking care of any differences within each type, such as possible differences in voltage of electrical consumption. The results are printed in a text box where can be copied with the normal copy and paste Windows commands, to be reported on any other type of document.

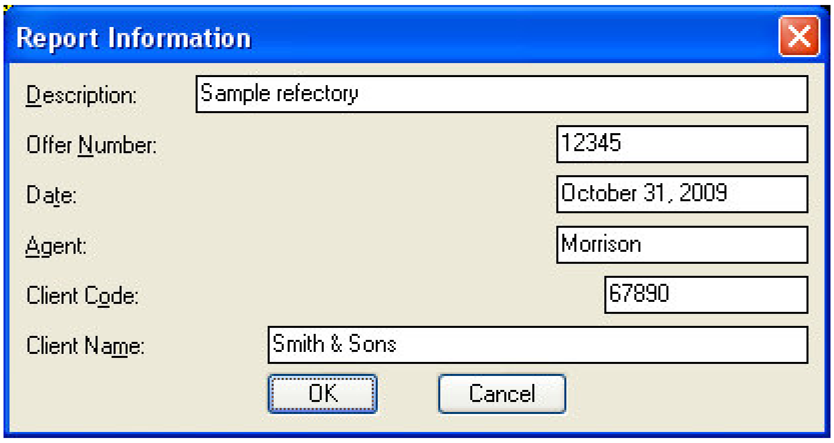


```
GAS       KwGas      : KWG =   35 (Kcal/h = 30100)
ELECTR.   230v       :  KW =  0.5
ELECTR.   400v       :  KW = 44.5
WATER IN ACADD L/h:   LH =  320
```

TABLES COMMAND

Once finished the markings of all the objects, you can automatically generate the summary table simply invoking this command, that examines all the drawing, checks that each object has been marked, and verifies that each section table does not contains two different objects with the same numbers. If the command finds an object not marked, it display a warning in AutoCAD text window, specifying its name and its insertion point. The command allows two or more object within the same section table to share the same number only if the objects are equal and eventually have the same accessories and modifications.

First of all, the command displays a window where you can specify some data concerning the work, such as the client name, the agent, the date, which is initalized with the current date; these informations makes up the report header and may not be specified in this stage, since the same window is accessed from KCADWIN application in Windows environment, Where can be changed all the information already set..


Report Information

Description:	Sample refectory
Offer Number:	12345
Date:	October 31, 2009
Agent:	Morrison
Client Code:	67890
Client Name:	Smith & Sons

The insertion point is prompted for every section table in the summary table. At the first prompt, the table header is inserted before the section table content. After having inserted the first section table, the next ones can be appended to it by pressing ENTER at the prompt for the next insertion point. If you select a new insertion point, the header of the summary table is drawn again and the section table is appended to it. In this way, it is possible to divide a very long table in different parts of the drawing.

At the time of production of the summary table, the program checks if some plug is present. If so, first is internally invoked the PLUGS command, to make sure that all the plugs are present, then it considers the consumptions of all the plugs to determine the format that the summary table should take. For each consumption type provided, a new column is added to the summary table, so that only the columns actually used are added. After having inserted the last table, a new table row is added with the totals of all the consumption types. See in SYSTEM MAINTENANCE which are the consumptions provided.

If you want a table without consumptions calculation even if the plugs symbols are present in the drawing, is possible to set properly the “Consumption calculation prompt” among the configuration variables displayed with the “KCAD” command. In this case, if some plugs are detected in the drawing, you will be prompted if you want the consumption calculation on the table or not.

Together with the summary table physically inserted in the drawing, the command generate also a data file with KCR extension, called “KCAD report”, that can be edited with KCADWIN Windows application and can be read from all the application that generate the offers and other types of document. This file does not change whether the plugs have been considered or not.

REPORT COMMAND

This command is used to generate the KCAD report file of data to manage with KCADWIN and for the automated generation of documents, without having to generate the summary table to be inserted in the drawing. This command does the same checks on the numbering of the TABLES command, but not those related to the presence of plugs (which do not affect the data file format). As this command is internally invoked by the TABLES command, it is useful only if you want to generate again the KCR file from an existing drawing file without modifying it in any way. The generated file is called “KCAD report file” or “KCR file” for its extension.

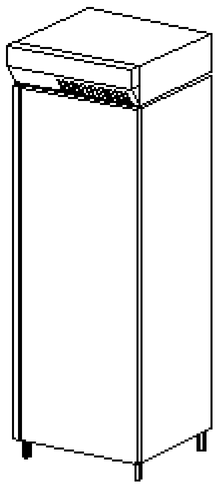
KCAD3D COMMAND

This command allows you to generate three-dimensional drawing from two-dimensional. This transformation is reversible using the KCAD2D command, however we recommended to perform it on a copy of the drawing, not on the original one. Indeed, the very substance of the drawing changes because from design plant it becomes a three-dimensional view.

The command must be run only once: inserting new bidimensional objects after the transformation, is not always possible to change them again in three-dimensional ones. Any changes must be made on the design plant, then the design must be copied and the copy can finally be transformed into three-dimensional view.

The blocks exploded with the MODIFICATIONS command cannot be transformed (that is because it is assumed that their appearance has changed). However, once produced, the three-dimensional view can be changed like any other AutoCAD drawing (except for the insertion of new objects that, as stated above, may not always ge transformed in three-dimensional).

Some objects, such as the shelves, after the transformation in three-dimensional objects must be placed at a certain height from the ground. To achieve this, you can add a custom line of data in the plugs field of the object, that does not refer to any plug, but specifies the height from ground using the syntax "HZ = XXX". See "SYSTEM MAINTENANCE" for further details.



KCAD2D COMMAND

This command allows you to transform back the three-dimensional drawing from the two-dimensional one. This transformation is provided in order to remedy a conversion from 2d to 3d made incorrectly or in a drowing not copied. As already said in the explanations of KCAD3D command, do not to use the original drawing to produce the 3d view, but perform the transformation on a copy of the drawing.

If after running the KCAD3D command, was mistakenly inserted a 2d object corresponding to another already transformed in 3d, you cannot use this command to reproduce the original 2d drawing because the 3d blocks corresponding to the present 2d object cannot be transformed again in 2d. In this case, you can solve the situation in this way:

1) If you tried to run the KCAD2D command, run again the KCAD3D command to bring back the drawing in 3d state. Only the 2d objects mistakenly inserted in the 3d drawing will remain in 2d.
2) Delete all the 2d objects remained in the 3d drawing (they should just be the only objects previously mistakenly inserted in the 3d drawing).
3) Delete all the 2d block definitions of all deleted 2d objects using the PURGE AutoCAD command.
4) Now the command KCAD2D should succeeded; otherwise, probably not all the mistakenly inserted 2d objects were removed, so repeat all the steps starting from the first point.

GLANCE COMMAND

While considering a design in progress, may be useful to have a quick view of the objects appearance. With this command, which prompts you to select an object, is shown on the screen the same slide that was used to select the object whe was inserted in the drawing. The command require to press ENTER to redraw the screen and return to the normal AutoCAD prompt. In AutoCAD text window are given all the information needed to identify the object (list, division, slide and model).

SOUND COMMAND

This command allows you to have a sound feedback of operations you perform, producing a high tone for each operation completed successfully, and a low tone for each operation not completed because of some unexpected event. Running the command the first time, the sounds are enabled (and the high tone is produced); running it a second time, the sounds are disabled (silently).

BULKHEAD COMMAND

This command allows you to calculate the bulkhead square metres that can be placed in a customized way inside a refrigerator or cold room. The command prompts you to select a refrigerator; According to the size of the selected refrigerator, the height of the bulkhead is proposed. Then the command prompts you to select the lines to be considered bulkhead (the selected entities have to be normal AutoCAD lines). Finally you get the total square metres resulting. This number may be used to specify the quantity of the bulkheads chosen among the accessories of the refrigerators (in square metres units).

DISCOUNT COMMAND

During the design, it is not considered the problems relating any discounts applied to each object inserted in the drawing; this kind of data is introduced in a second stage, using the Windows application “KCADWIN”. However, you may require to change a drawing for which discounts have already been set. Since the discounts are stored in report KCR file along with the details of objects and attributes, regenerating the report from the drawing will cause the deletion of any discount set. With this command, given a report of the drawing previously generated with the discounts already set, the corresponding objects in the drawing are updated so that the respective discounts are stored among their other data, so when generating a new report, their discount will be set as specified in the previous report.

To achive this purpose, this command prompts you to select the report file where you want to import the discounts through a dialog box that allows you to choose the disk, the folder and the report file (with KCR extension) that is presumed to be tha last originally generated from the same drawing and then edited with KCADWIN to introduce the discounts to each individual objects. Then, this command searches in the drawing all the objects present in the report, considering the name in the respective section tables and their progressive number, pointing out the differences found not only in the objects identity but also in their possible modifications and accessories; thus allows you to use the command also to check the correspondence between a previuos report (that can be changed by KCADWIN) and the current status of the drawing.

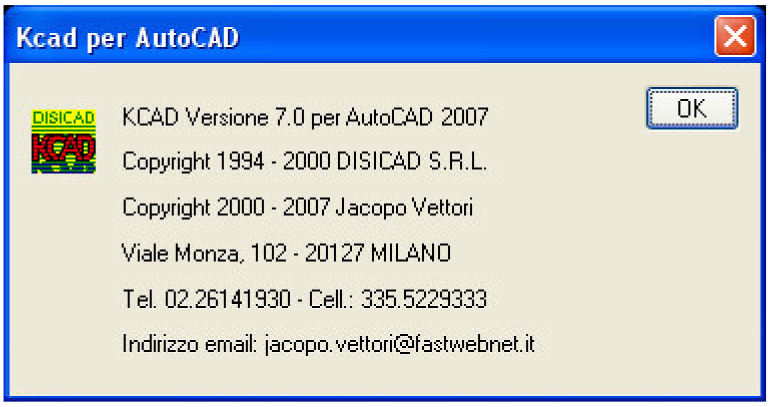
The command does not report any problem with objects inserted in the drawing, but not present in report file, nor those inserted in the drawing after the check: the discounts are set only for the objects that have a match at the actual time of command invocation.

If the differences concern only accessories or modifications, the command allows you to view the different accessories and modifications, and to update the object in the drawing with the same accessories and the same modifications of the corresponding object in the report file. In this case, as when you set the modifications to an object with the **Modifications** command, you must pay attention to possible differences in PLUGS field which could influence the consumption and the plugs symbolism on the drawing. In this case, the program prints a warning message.

All the objects that are in the report file, but are absent or different in the drawing, are shown with appropriate warnings in AutoCAD text windows: this allows you to use this command in order to verify the correspondence between drawing objects and those in the report file. You must however be careful with objects missing from the report file but present in the drawing; for them no warning is displayed, and the only information to detect their presence is the total number of objects in the drawing, printed by this command at the beginning of all the notices related to individual comparisons between objects.

KCADINFO COMMAND

This command displays a dialog box with indications about the product version number and information about contacting the software manufacturing company.



KCAD Versione 7.0 per AutoCAD 2007

Copyright 1994 - 2000 DISICAD S.R.L.

Copyright 2000 - 2007 Jacopo Vettori

Viale Monza, 102 - 20127 MILANO

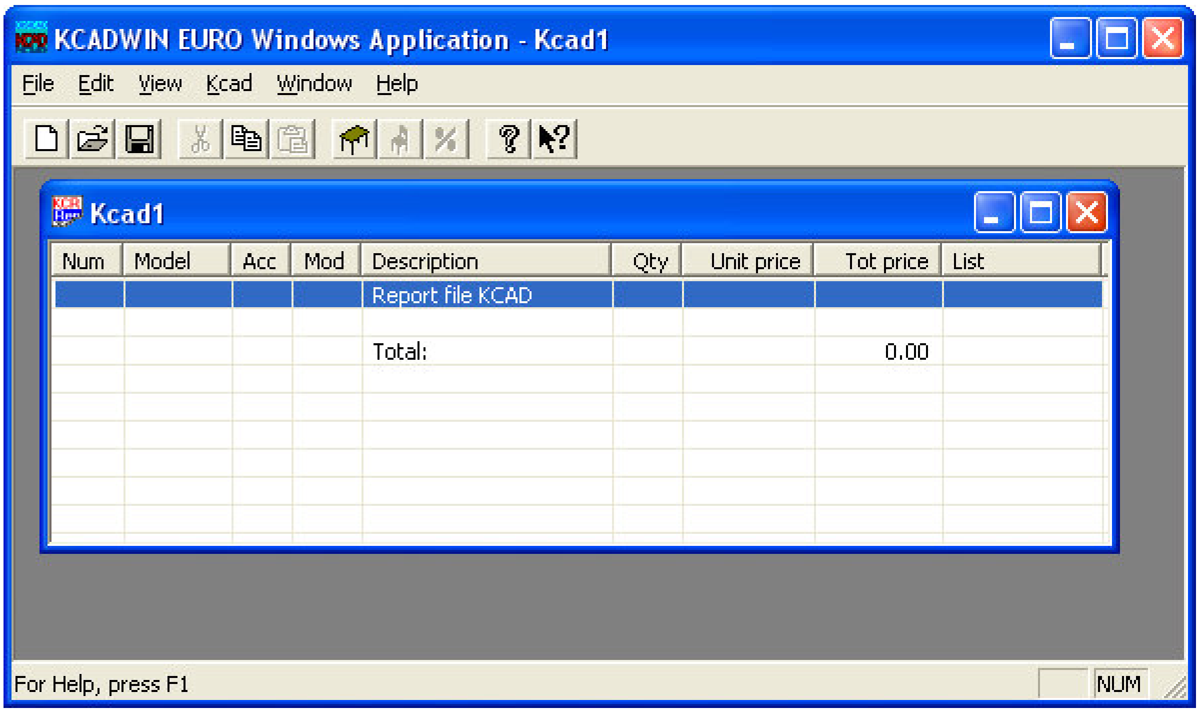
Tel. 02.26141930 - Cell.: 335.5229333

Indirizzo email: jacopo.vettori@fastwebnet.it

KCADWIN APPLICATION COMMANDS DESCRIPTION

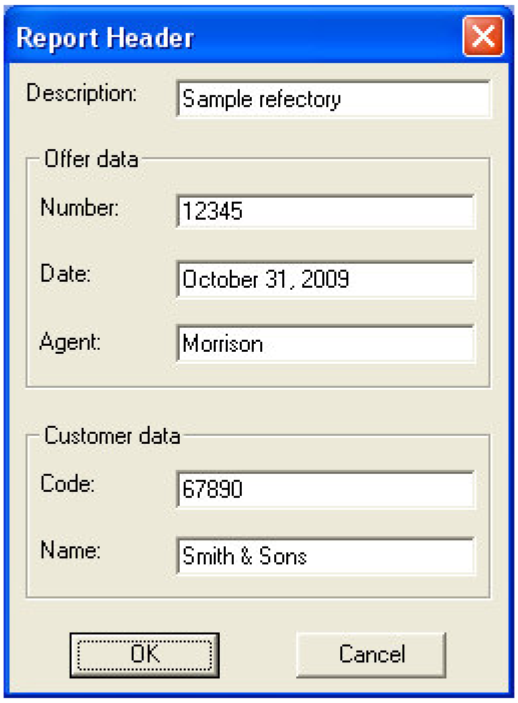
FILE MENU

New: Creation of a new report. Selecting this item, it immediately opens a new window with a new blank report.


KCADWIN EURO Windows Application - Kcad1
File Edit View Kcad Window Help
Kcad1
For Help, press F1
NUM

Num	Model	Acc	Mod	Description	Qty	Unit price	Tot price	List
				Report file KCAD				
				Total:			0.00	

The reports are represented in a similar way of the summary tables in the drawing: each row of the window corresponds to an object or to the header of a section table. The first row, with the name of the report, is the header of the report, with the data about the client, the agent and other general information. The last row displays the total price of the offer. When a new blank report is generated with this command, only the first row (the report header) is created; double clicking on this row (or pressing ENTER when the window is active and the row selected), a dialog box is open, similar to the one provided by generating the report from inside AutoCAD, displaying all the general data of the offer. The date is set by default with the current date.



Description: Sample refectory

Offer data

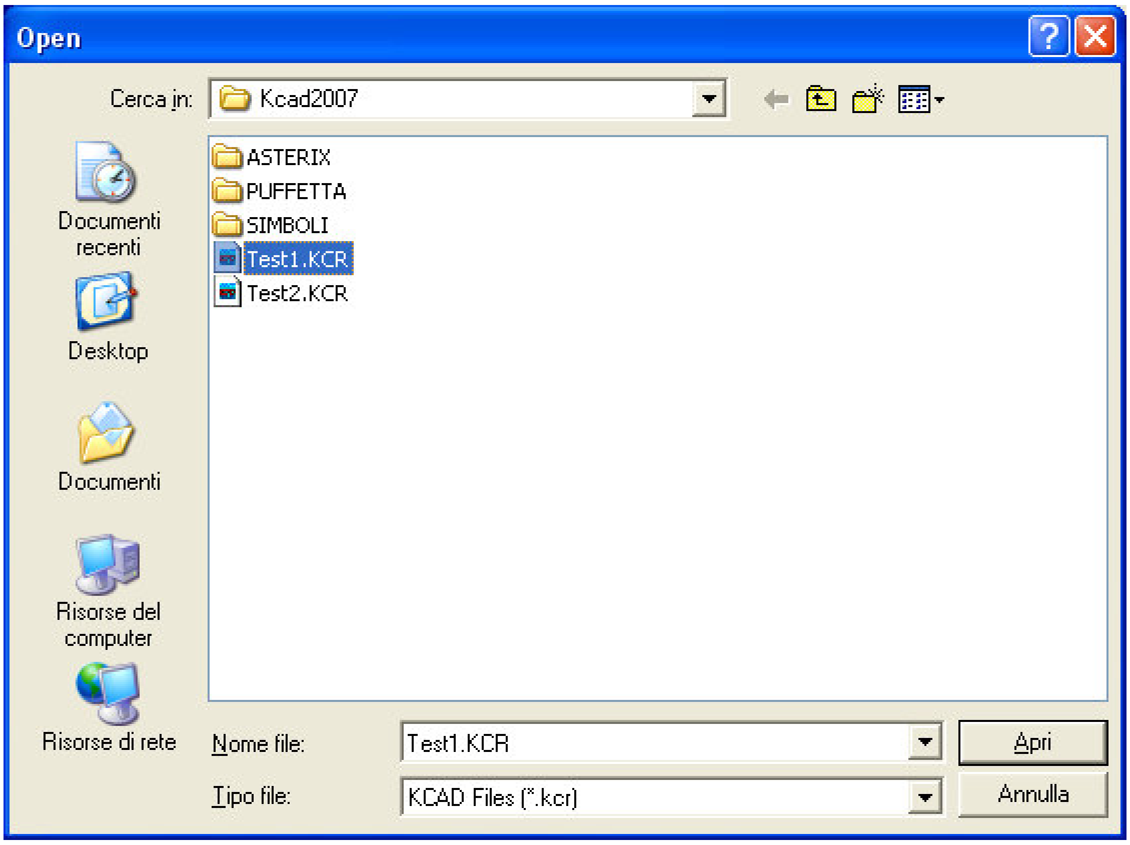
Number:	12345
Date:	October 31, 2009
Agent:	Morrison

Customer data

Code:	67890
Name:	Smith & Sons

To add new section tables and new objects, see below the KCAD menu items INSERT TABLE and INSERT OBJECT. This command **NEW** may be given also selecting the first button of the toolbar buttons below the menu:

Open…: Opens an existing report file (with KCR extension), through the standard Windows dialog box, that allows you to select the disk, the folder and the KCR file you want to load.



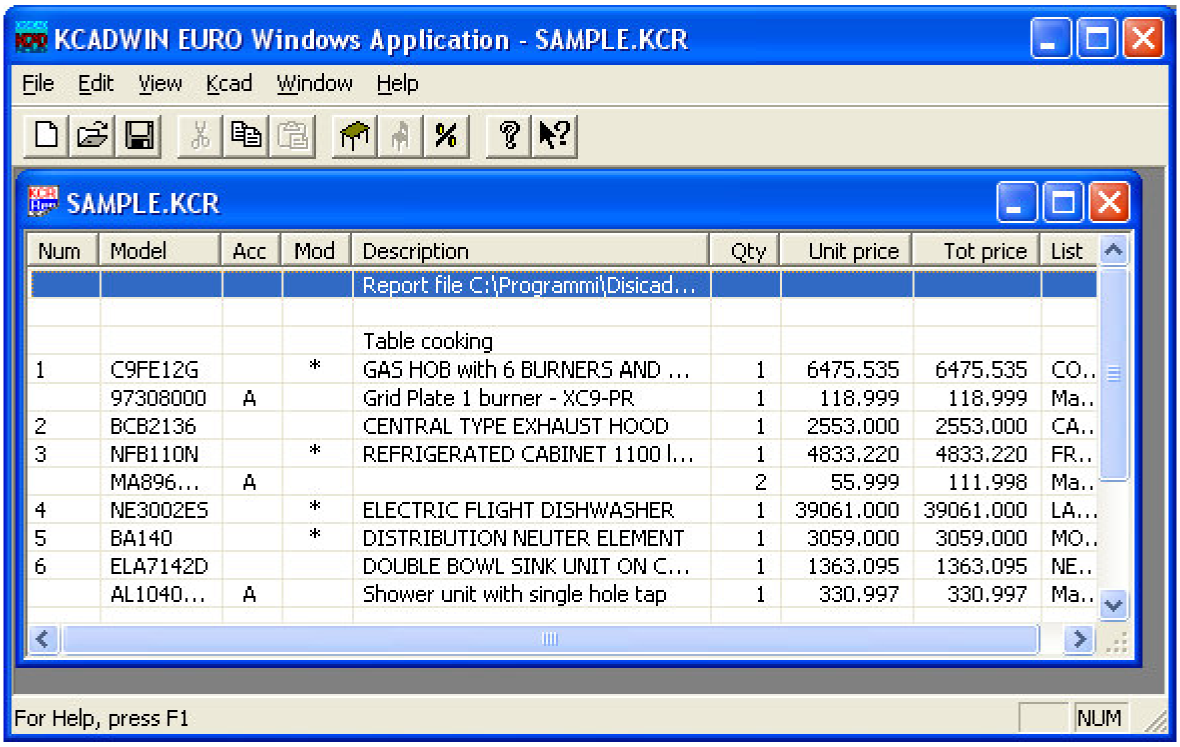
Cerca in: Kcad2007

ASTERIX
PUFFETTA
SIMBOLI
Test1.KCR
Test2.KCR

Nome file: Test1.KCR

Tipo file: KCAD Files (*.kcr)

Once you click the **OK** button of the dialog box, a new window is opened, structured in rows similar to the summary table generated in AutoCAD drawing, where each line corresponds to an object or to the name of a section table, except the first line that corresponds to the report header, and the last one that reports the total price of the offer. Using the arrow up and down keys, you can select any report row.

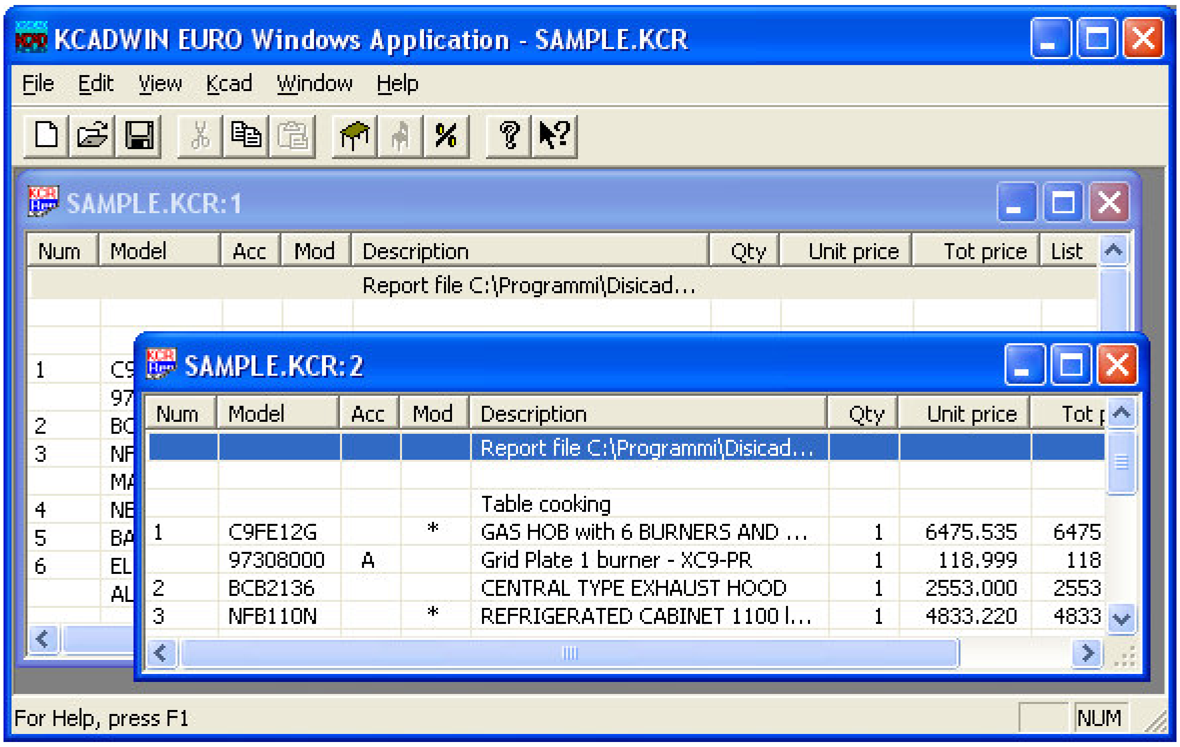


Num	Model	Acc	Mod	Description	Qty	Unit price	Tot price	List
				Report file C:\Programmi\Disicad...				
				Table cooking				
1	C9FE12G		*	GAS HOB with 6 BURNERS AND ...	1	6475.535	6475.535	CO..
	97308000	A		Grid Plate 1 burner - XC9-PR	1	118.999	118.999	Ma..
2	BCB2136			CENTRAL TYPE EXHAUST HOOD	1	2553.000	2553.000	CA..
3	NFB110N		*	REFRIGERATED CABINET 1100 l...	1	4833.220	4833.220	FR..
	MA896...	A			2	55.999	111.998	Ma..
4	NE3002ES		*	ELECTRIC FLIGHT DISHWASHER	1	39061.000	39061.000	LA...
5	BA140		*	DISTRIBUTION NEUTER ELEMENT	1	3059.000	3059.000	MO..
6	ELA7142D			DOUBLE BOWL SINK UNIT ON C...	1	1363.095	1363.095	NE..
	AL1040...	A		Shower unit with single hole tap	1	330.997	330.997	Ma..

Double-clicking with the mouse or pressing ENTER when a row is selected, a dialog box is displayed that let you view and edit all the information associated to the row: if it is the first row, the report header information; if the row represent a section table name, its name, which can be changed; and if the row represent an object, all the information that concerns it, with all its accessories and modifications, and the price for the customer; which can be discounted through a given percentage or entering directly the new price. For more information, see below the description of the individual dialog boxes. The **OPEN…** command may be given also selecting the second button of the toolbar buttons below the menu:

NOTE: The editing of a KCR file with this application makes it miss its matching with the drawing from which it has been extracted (except that the design has not been made and the KCR was created directly from this program, with the **NEW** command described above), so it will be necessary afterwards to restore this correspondence, with the help of the DISCOUNT command of KCAD add-in application in AutoCAD environment, which indicates all the differences between a drawing and a KCR file.

Several KCR files can be opened simultaneously, in order to compare the contents and copy the header, the objects and even entire section tables from one file to another (see the **EDIT** menu below).


KCADWIN EURO Windows Application - SAMPLE.KCR
File Edit View Kcad Window Help
SAMPLE.KCR:1
SAMPLE.KCR:2
For Help, press F1
NUM

SAMPLE.KCR:1

Num	Model	Acc	Mod	Description	Qty	Unit price	Tot price	List
				Report file C:\Programmi\Disicad…				
1	C9							
	97							
2	BC							
3	NF							
	MA							
4	NE							
5	BA							
6	EL							
	AL							

SAMPLE.KCR:2

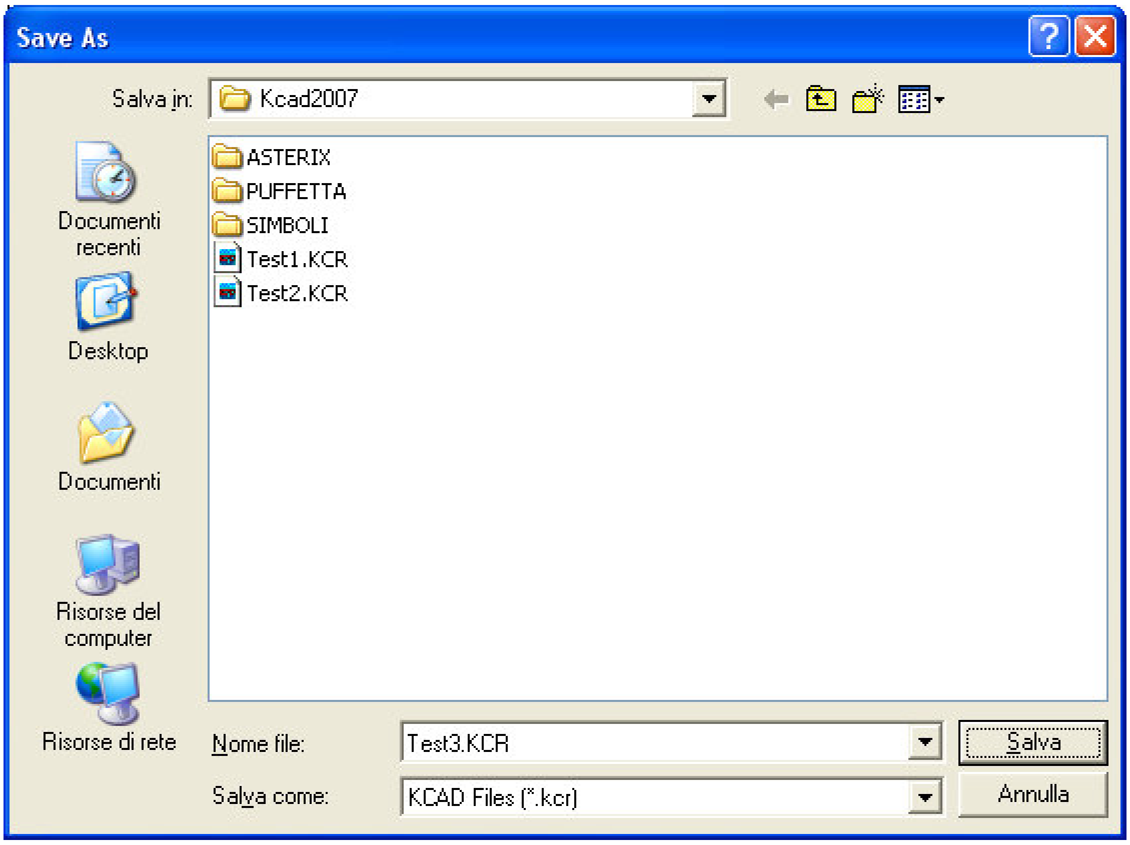
Num	Model	Acc	Mod	Description	Qty	Unit price	Tot p
				Report file C:\Programmi\Disicad…			
				Table cooking			
1	C9FE12G		*	GAS HOB with 6 BURNERS AND …	1	6475.535	6475
	97308000	A		Grid Plate 1 burner - XC9-PR	1	118.999	118
2	BCB2136			CENTRAL TYPE EXHAUST HOOD	1	2553.000	2553
3	NFB110N		*	REFRIGERATED CABINET 1100 l…	1	4833.220	4833

Moreover, you can also open the KCR files directly from Windows Explorer, selecting the file name and pressing ENTER (or double-clicking on the file name):

Windows provides to load Kcadwin and automatically open the selected file. Having simultaneously opened on the screen both Windows Explorer and Kcadwin, you can also to open a KCR file using the “drag and drop” Windows feature. Selecting the name of a KCR file in Windows Explorer, without stopping to press the left mouse button, move the mouse till you bring the cursor inside Kcadwin window, and finally release the left button: Kcadwin will immediately open a new window with the selected KCR file.

Close: This item appears only if at least one KCR file is currently open, and it is used to close the active KCR file; if there are any modifications that have not yet been saved, you are prompted to save changes before closing the file.

Save: This item appears only if at least one KCR file is currently open, and allows you to save any changes done, and then going on editing or closing the file. If the file does not have a name, the standard Windows dialog box is displayed, that allows you to choose the disk , the folder and the name of the new KCR file. You can also specify the file extension, but if it is other than “KCR”, the file will not be automatically recognized by KCAD applications and by Wiondows Explorer. If no extension is specified, the KCR extension is added by default.


Save As

Salva in: Kcad2007

ASTERIX
PUFFETTA
SIMBOLI
Test1.KCR
Test2.KCR

Nome file: Test3.KCR

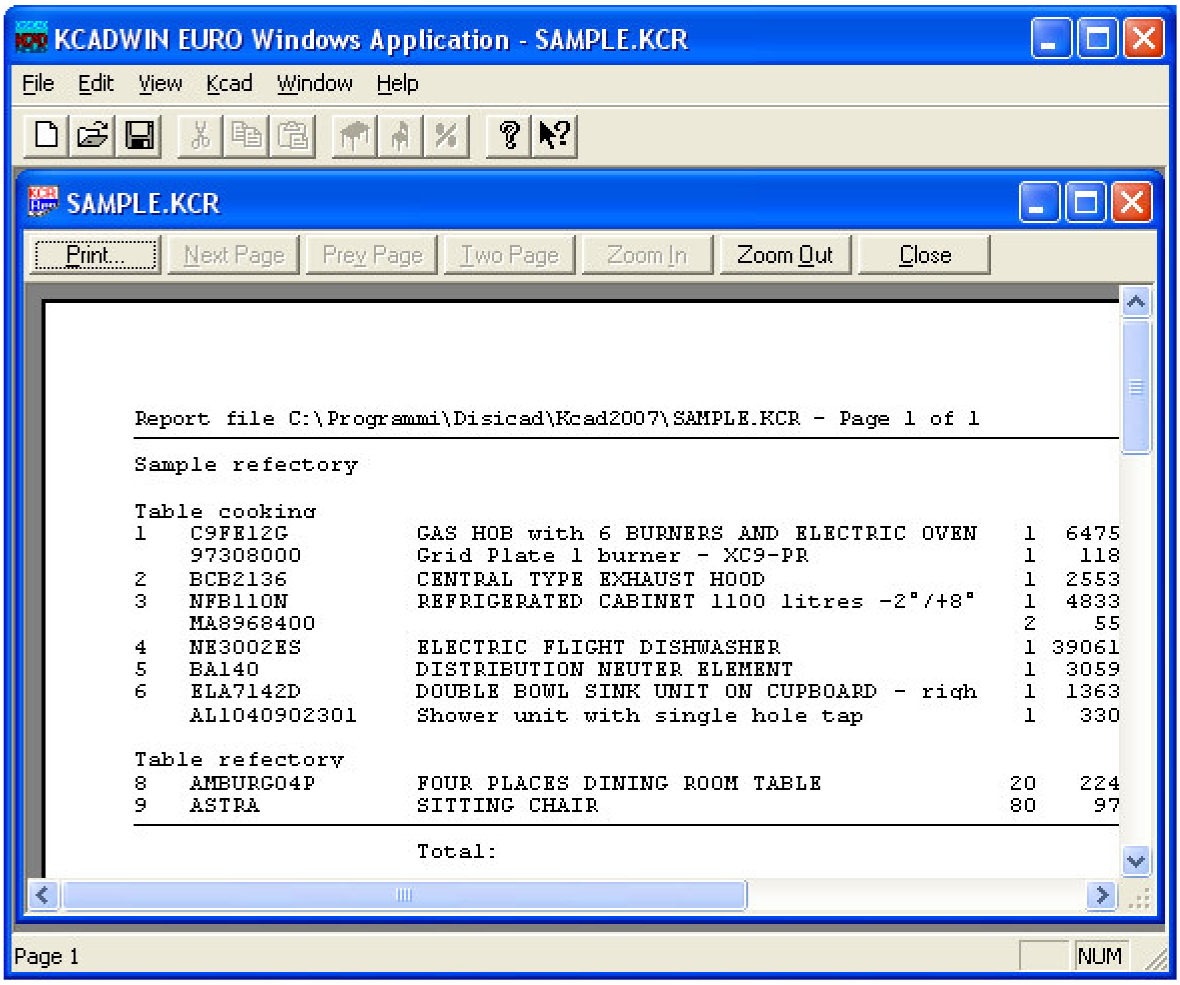
Salva come: KCAD Files (*.kcr)

If the command is enabled, it can be imparted by selecting the third button of the toolbar buttons below the menu:

Save as…: This item appears only if at least one KCR file is currently open and allows you to save any changes done, and then going on editing or closing the file, but choosing a new name for the KCR file, using the same dialog box displayed the first time you save a new file, so you can keep the original copy without modifications and going on changing the new copy with the new name.

Print: This item appears only if at least one KCR file is currently open, and allows us to produce a simple print of the selected KCR file, to quickly produce an estimate for internal use or an example for the customer.

Print Preview: This item appears only if at least one file KCR is currently open, and allows you to preview the printing of the selected KCR file.


KCADWIN EURO Windows Application - SAMPLE.KCR
File Edit View Kcad Window Help
SAMPLE.KCR
Print... Next Page Prev Page Two Page Zoom In Zoom Out Close
Page 1
NUM

```
Report file C:\Programmi\Disicad\Kcad2007\SAMPLE.KCR - Page 1 of 1

Sample refectory

Table cooking
1    C9FE12G          GAS HOB with 6 BURNERS AND ELECTRIC OVEN    1  6475
     97308000         Grid Plate 1 burner - XC9-PR                1   118
2    BCB2136          CENTRAL TYPE EXHAUST HOOD                   1  2553
3    NFB110N          REFRIGERATED CABINET 1100 litres -2°/+8°    1  4833
     MA8968400                                                    2    55
4    NE3002ES         ELECTRIC FLIGHT DISHWASHER                  1 39061
5    BA140            DISTRIBUTION NEUTER ELEMENT                 1  3059
6    ELA7142D         DOUBLE BOWL SINK UNIT ON CUPBOARD - righ    1  1363
     AL1040902301     Shower unit with single hole tap            1   330

Table refectory
8    AMBURG04P        FOUR PLACES DINING ROOM TABLE              20   224
9    ASTRA            SITTING CHAIR                              80    97

                      Total:
```

The names of the last 4 opened KCR files are listed in this menu before the **EXIT** item, so that you can reopen one of these files by simply selecting them and pressing ENTER (or double click with mouse left button).

Exit: This command will exit the Kcadwin application; before you leave, you are asked if you want to save each modified KCR file not closed or saved yet.

MENU EDIT

This menu is displayed only if some KCR file is currently open.

Del: Deletes an object or a whole section table, based on the type of the current row. Before you delete the data, a confirmation is required (unlike **Cut**, **Del** does not keep in memory the section table or the selected object). The report header cannot be deleted, so this menu item is enabled only if the current row is not the first one.

Cut: Deletes an object or a whole section table, keeping a copy in memory, to allow the insertion elsewhere with the **Paste** command. Only one object or one section table at a time can be kept in memory: a second object takes the place of the first in memory, making impossible to recover it with the command **Paste**. Also in this case, a confirmation is required before the deletion of the object or the section table. The report header cannot be deleted, so this menu item is enabled only if the current row is not the first. The **Cut** command can be invoked also selecting the fourth button of the toolbar buttons below the menu:

Copy: With this command you can copy in memory an object, or a section table, or the report header, based on the currently selected row in the active KCR. The objects are copied with all their accessories and modifications, the section tables with all their objects, the header with the general information that it includes. Only one object or one section table or one header at a time can be kept in memory: a second one take the place of the previous. To retrieve the copy, use the **Paste** command, described below. The **Paste** command can be called more times, because the copied item remains in memory untill it is replaceded with something else. This command may be given by selecting the fifth button of the toolbar buttons below the menu:

Paste: If something has been copied in memory, this command is enabled, which allows the insertion in the same KCR file or in different one. Since the copy is stored in the clipboard memory of Windows, it is even possible to close the Kcadwin application and then come back without losing the copy in memory, provided that you don’t logout from Windows and you have not used the functions CUT or COPY of other Windows applications. This command may be given by selecting the sixth button of the toolbar buttons below the menu:

MENU VIEW

Toolbar: This menu item, normally activated (with the check mark), allows the display of the toolbar buttons below the menu.

Status bar: This menu item, normally activated (with the check mark), allows the display of the status bar at the bottom of the window. The status bar displays short messages about the selected commands and the state of the keyboard caps lock and num lock.

MENU KCAD

This menu is displayed only if some KCR files are currently open.

Sort report: If the report has multiple section tables, this command will sort the section tables based on the progressive number of child objects contained in each. If more section tables begin with the same progressive number, are ordered by their names. It's possible to add a number at the beginning of the section table name, to force the desired order. Within each section table, the objects are sorted by their progressive number; if two objects share the same number, are sorted by their description.

Insert table…: Creation of a new section table. Selecting this item, a dialog box is opened, that allows you to specify the section table name. This section table is created in the first available position after the currently selected row: if the current row is the first (corresponding with the report header), the new section table will be the first section table of the report (others section tables will eventually shift down); if the current row is another section table or an object belonging to another section table, the new section table will be added after the last object of that section table: it is not possible to "cut" into two tables a section table adding a new table name between its object items: in this case, you may add the new table below the one you want to divide, and then move the objects from the first table to the next one using the **Cut** and **Paste** commands of **EDIT** menu. This command may be given by selecting the seventh button of the toolbar buttons below the menu:

Insert object…: Creation of a new object. This item is enabled only if in the report window is selected a section table name or an object belonging to that table, but not when is selected the report header, because an object must always belong to a section table. The dialog box displayed let you select an object in a similar way as is selected in AutoCAD add-in application; for the detailed explanation, see below the description of the individual dialog boxes. When an object is added to a table, its position inside the table is determined by its progressive number. There are no controls on the numbering in different tables; the objects are kept in order according to their progressive number only inside each table, and two objects with the same number are not allowed: in this case, the second object replaces the first one. This command may be given by selecting the eighth button of the toolbar buttons below the menu:

Edit table…: Modification of an existing section table name. Selecting this item, a dialog box is opened that allows you to change the name of the selected table. This command may be given also pressing ENTER when the name of the table is selected, or double-clicking on it with the mouse left button. For more information see “DIALOG BOXES DESCRIPTION”.

Edit object…: Modification of an existing object. Selecting this item, a dialog box is opened that allows you to change the selected object. This command may be given also pressing ENTER when the object is selected, or double-clicking on it with the mouse left button. For more information see “DIALOG BOXES DESCRIPTION”.

Discount…: Modification of the discounts of objects and accessories in all the report or in the current table only. Selecting this item, a dialog box is opened that allows you to specify a percentage discount and the different criteria you want to use to apply the discount. Before executing the operation, a further dialog box allows you to check the given criteria. For more information see “DIALOG BOXES DESCRIPTION”. This command may be given by selecting the ninth button of the toolbar buttons below the menu:

MENU WINDOW

This menu is displayed only if some KCR file is currently open.

New window: Selecting this item, a second window is opened, containing the same KCR file of the active child window. In this way, you can have two views on the same KCR, displaying, for example, the beginning and the end simultaneously. In this way, you DO NOT create a second copy of the KCR file, you simple have two viewes on the same KCR file (or even more than two, repeating the same command). Any change made in one of these windows, it automatically propagates in the other windows.

Cascade: Selecting this command, all the opened windows are arranged in “cascade mode”, so that each overlaps the previous one, leaving protrude above the top and the left edges.

Tile: Selecting this command, all the opened windows are arranged in “tile mode”, so that each is placed side by side, leaving all of them visible.

Arrange icons: The views of the KCR windows can be minimized or reduced to icon, selecting the provided button placed in the right part of the top window edge: these icons remain inside the main application window, and can be moved with the mouse and covered by other windows. With this command, they are arranged along the bottom of the main window.

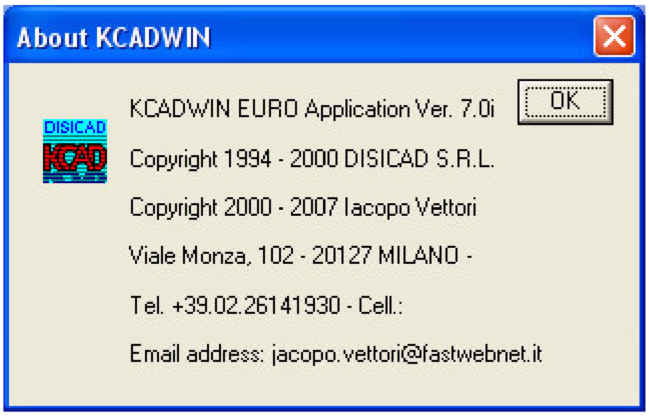
In this menu, under all the items, appears the list of all the windows that are opened or minimized, so that they can be activated simply by selecting their name. The currently active window has a check mark before its name in this menu.

MENU HELP

Index: Select this item to open the Windows Help program and read the Kcadwin commands guide.

Using help: Select this item obtain information about the use of Windows Help.

About Kcadwin…: This command displays a window containing information about the current version of Kcadwin application and the information to contact the software manufacturing company.



KCADWIN EURO Application Ver. 7.0i

Copyright 1994 - 2000 DISICAD S.R.L.

Copyright 2000 - 2007 Iacopo Vettori

Viale Monza, 102 - 20127 MILANO -

Tel. +39.02.26141930 - Cell.:

Email address: jacopo.vettori@fastwebnet.it

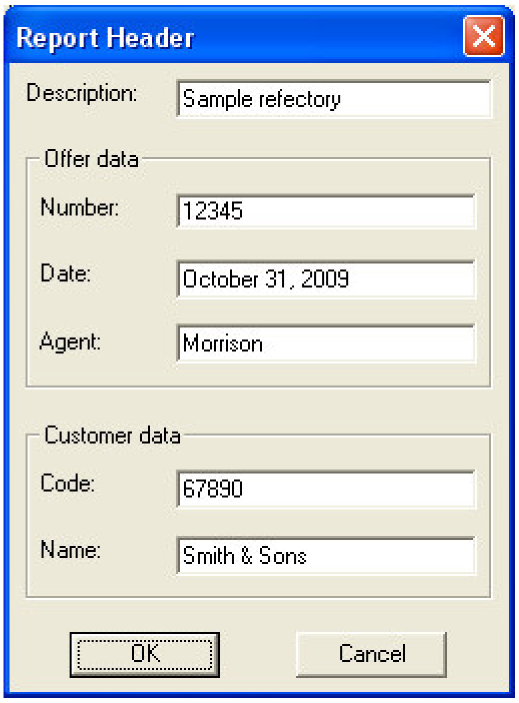
The same window appears clicking the tenth button in the toolbar buttons below the menu:

HELP CONTEXT BUTTON

Selecting this button, the cursor changes its shape in a arrow with a question mark, similar to the one shown on the button. Selecting any menu item, or a button, or a window, the Kcadwin Help is called with all the information about what was selected.

DIALOG BOXES DESCRIPTION

Report Header dialog box



Description: Sample refectory

Offer data

Number: 12345

Date: October 31, 2009

Agent: Morrison

Customer data

Code: 67890

Name: Smith & Sons

Description: The aim of this description is to distinguish the information coming from different KCR files when it is necessary to generate an offer composed from more KCR files; it should provide a practical description of the context in which the report relates. In a customized application it could be used, for example, to view a list of all the offers generated in a month, indicating the number and the description.

Offer data – Number: Specify the offer number. It may also be an alphanumeric identifier.

Offer data – Date: If the date is missing, the current date is automatically set. If there is previously set another date and you want to set the current date, you can delete it, then close the window with **OK** and then open it again, because in this case the current date is set by default. Close again the window with **OK** to store the date in the report header.

Offer data – Agent: Name of the agent that made the offer.

Customer data – Code: Customer internal identifier. It may be alphanumeric.

Customer data – Name: Customer or company name, as it will appear in the offer.

Table window

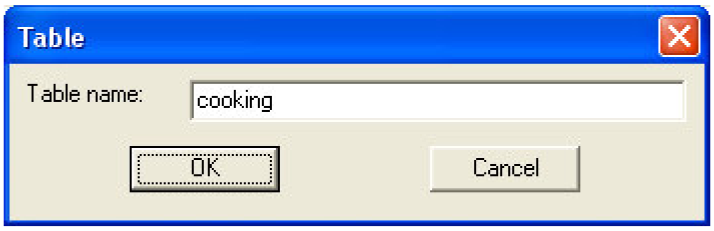
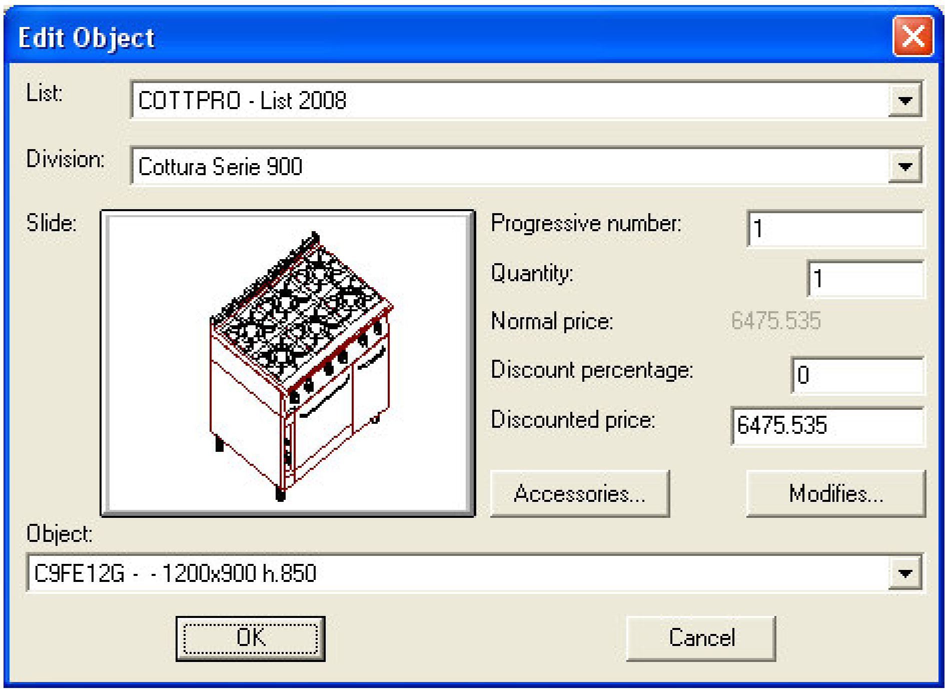


Table name: cooking

Table name: Using this window it is possible to set and change just the section table name and not the objects that it contains. The section table name is used in the drawing and in the offer document to separate the sections related to different environments.

Objects Editing window



List: COTTPRO - List 2008

Division: Cottura Serie 900

Slide:

Progressive number: 1

Quantity: 1

Normal price: 6475.535

Discount percentage: 0

Discounted price: 6475.535

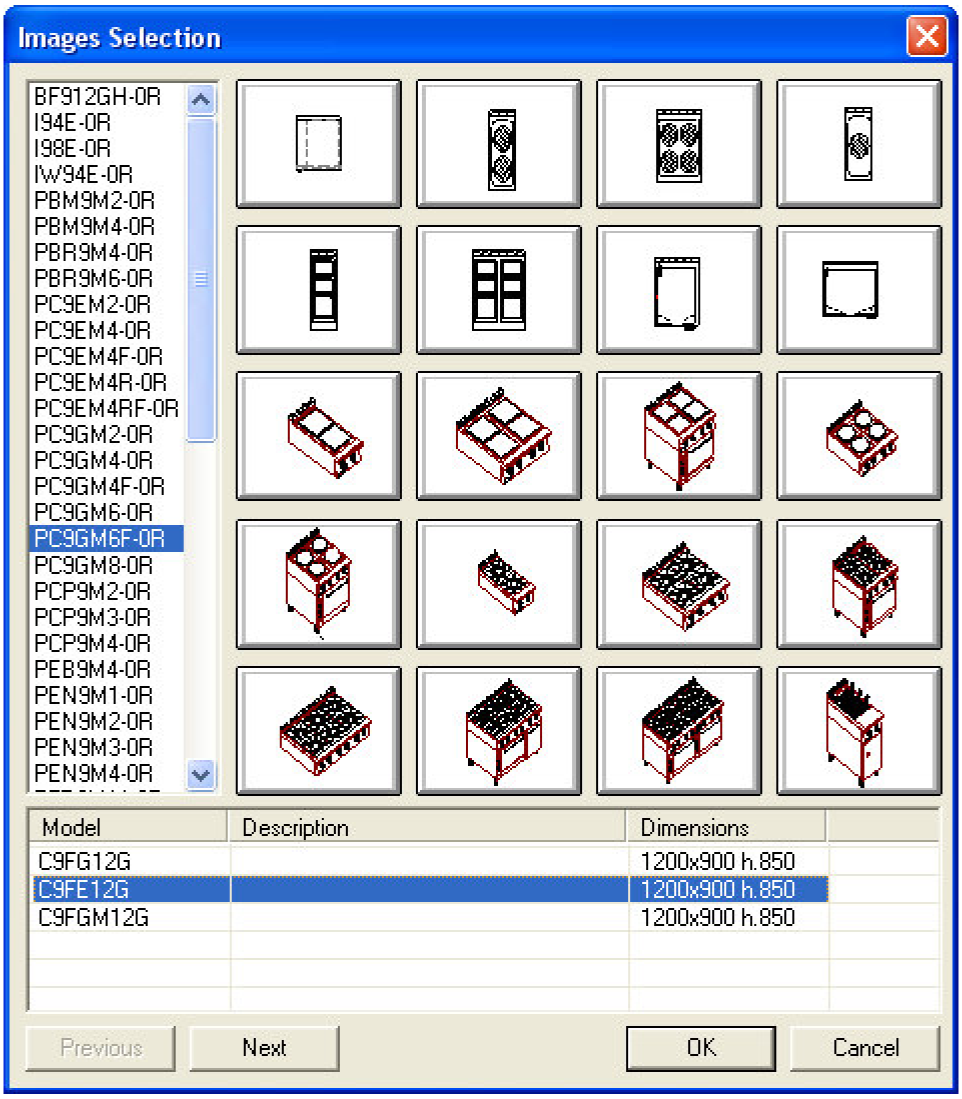
Object: C9FE12G - - 1200x900 h.850

When this window is opened to edit an existing object, all the settings reflect those of the object, but if it is opened to add a new object, the furniture list set by default is the one used for the last edited object (or the first in the list, if the object edited is the first created); the progressive number set by default is the next of the last used progressive number. The division, the slide and the code are set according the first object of the furniture list.

List: Dropdown box for the furniture list selection. The first time you create an object, the default setting is the first furniture list; every next times, the default will be the last selected furniture list. When this selection changes, the division is set to the first one in the list of divisions, and the slide and the code are set according the first object of this division.

Division: dropdown box for the division selection. When the window is opened to create an object, the selected division is alwais the first. When this selection changes, the slide and the code are set according the first object of the new division.

Slide: Inside this button is displayed the slide corresponding to the selected object. If you want to choose a different object within the same division, click on this button and a new dialog box will open; here you can select the new object according to the slide of the entire division, shown in subsequent pages where they are displayed twenty at a time, similar to what appens in KCAD AutoCAD add-in application. The application searches for all the slides whose name appears in the column SLIDE of the database tables; if any slide is not found, instead of the image, inside the button is displayed the words "SLIDE NOT FOUND".



Model	Description	Dimensions
C9FG12G		1200x900 h.850
C9FE12G		1200x900 h.850
C9FGM12G		1200x900 h.850

The listbox at the bottom of the window, lists of all the objects that share the selected slide. Among these you can choose the one you want by its code, the short description and its dimensions.

In the creation of a batabase table, the slide name is essential to divide the objects in groups; for this reason it is better to specify the slide name even if this one has not been produced yet. If the file specified as slide has a invalid or incompatible format, an error messages will be displayed (the application is based on the AutoCAD slide format).

Object: Once selected the slide from the dialog box “Slide selection”, this dropdown box shows the list of all the objects with different code that share the same image. Along with the code, the short description and the dimensions allow to better identify the the differences among the listed objects. If you want to replace the object with another that shares the same slide, you can select it from this list without having to reopen the dialog box with the slides.

Progressive number: When you create the first object in a new report, this is set to one; then, while creating other objects, by default it is set to the successor of the last used number. Since are allowed numbers with decimal point, the successor will vary according to the number of decimal places used: thus, the successor of 3 is 4, and the successor of 3.6 is 3.7.

Quantity:The quantity of the objects is always an integer number because is assumed that it specifies the physical number of objects equal to each other, while the quantity of accessories may also be a decimal and may also have its own unit of measure, to allow, for example, to add to the refrigerators a decimal number of bulkhead square metres. The quantity of the objects can also be 0, if you want, for example, to generate an offer that contains some accessories without any object. In this case, the accessories (which, however, inherit the progressive number of the object at which they are associated) but obviously cannot be embedded (see the description of the **Accessories** window for more details).

Normal price: Here is displayed the public price of the object, calculated by subtracting from its list price (stored in the database) the standard discount of the list (or that of its division, if it has been specified) and adding the standard recharge of the company on that furniture list (or the particular one of its division if it has been specified). This price is displayed but not editable, you may use it to guide you in setting the discounted price. If you need to change the original price, you can set the change through a modification (see the description of the **Modifications** window for more information), or you must update directly the price, the discount and the recharge in the database tables, if the changes do not concern only one object.

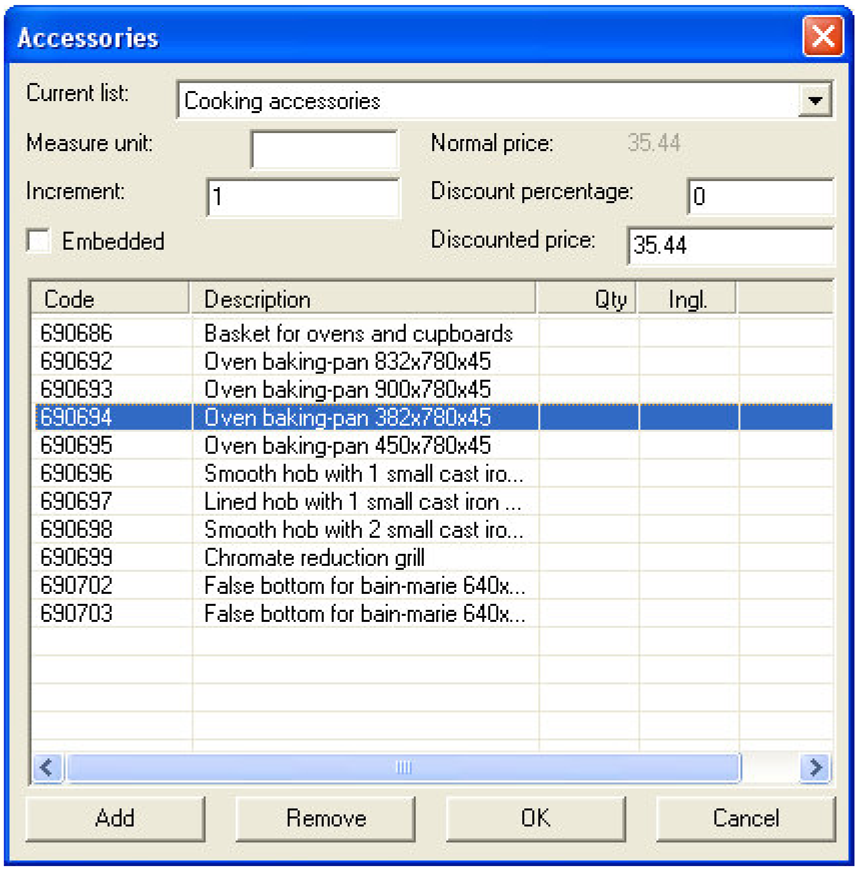
Discount percentage: In this edit box you can specify the discount percentage you want to apply to the normal price to get the discounted price. You can specify a percentage with two digit after the decimal point. Pressing ENTER (or moving to another edit box) the discounted price is updated, and because it do not allow rounded values, it may happen that the percentage discount given undergo a small change. You can use discount command described below to set a percentage discount to whole groups of objects and accessories.

Discounted price: Specify in this edit box the actual discounted price that will be propose to the customer. Pressing ENTER (or moving to another edit box) the discount percentage indicated in the upper edit box is automatically updated. You can use discount command described below to set a percentage discount to whole groups of objects and accessories.

Accessories…: Pressing this button, you access to the **Accessories** dialog box described below.

Modifications…: Pressing this button, you access to the **Modifications** dialog box described below.

Accessories Window



Current list: Cooking accessories
Measure unit:
Increment: 1
Normal price: 35.44
Discount percentage: 0
Discounted price: 35.44

Code	Description	Qty	Ingl.
690686	Basket for ovens and cupboards		
690692	Oven baking-pan 832x780x45		
690693	Oven baking-pan 900x780x45		
690694	Oven baking-pan 382x780x45		
690695	Oven baking-pan 450x780x45		
690696	Smooth hob with 1 small cast iro...		
690697	Lined hob with 1 small cast iron ...		
690698	Smooth hob with 2 small cast iro...		
690699	Chromate reduction grill		
690702	False bottom for bain-marie 640x...		
690703	False bottom for bain-marie 640x...		

Current list: In this dropdown box are listed all the available accessories lists. If the object has its own default accessories list, it is set as current, otherwise the is set the first of the list. In the main listbox window, are listed all the accessories that belong to the selected accessories list. On the right of each accessories list name, an asterisk appears if some accessory belonging to that list has been assigned to the object, so you can figure out which accessories have been assigned to the object, even if they come from different accessories lists.

Normal price: In this edit box, is displayed the accessory price as calculated from the accessories list, minus the supplier discount, plus the company recharge (stored as percentage in the table of the accessories lists). This number is not directly editable, eventually you have to change the database tables.

Discounted price: In this edit box you can set the final price you want to propose to the customer. By default, it is set equal to the normal price.

Unit of measure: Since the accessories can be supplied in not integer quantities, is necessary to be able to specify the unit of measure, with an abbreviation at most 4 characters long.

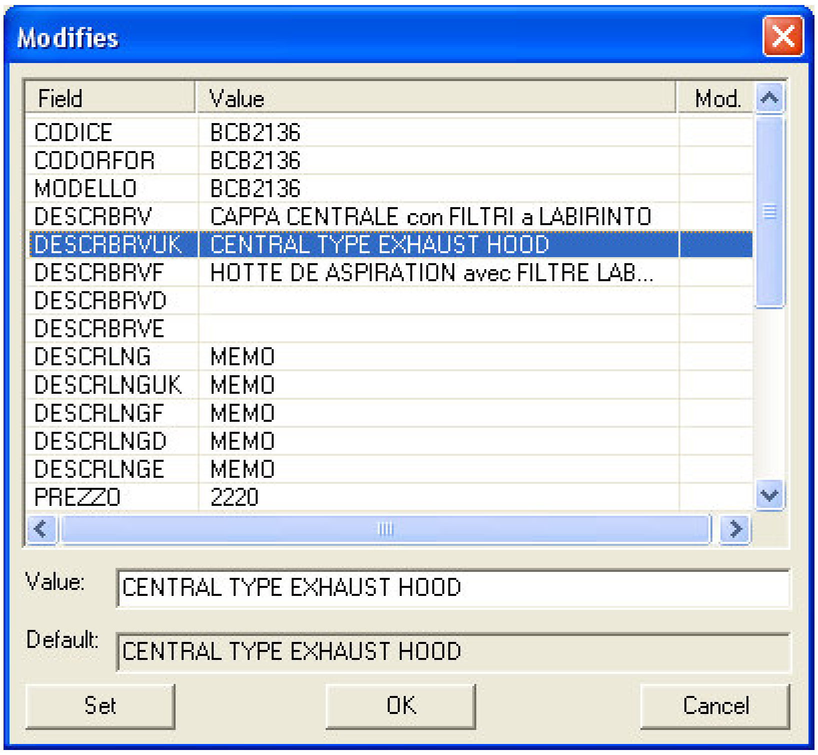
Increment: To set the quantity of accessories, you can use this edit box, usually set at 1, eventually using a decimal number (use the point to separate the units from the decimals). Then, selecting the **Add** and **Remove** buttons, the specified accessories quantity will be added or subtracted to the object. As an alternative to the **Add** button, you can double click on the accessory row in the listbox, or select it and press ENTER.

Embedded: This check box allows you to specify whether an accessory must be considered “embedded” or not. The difference comes out in filling the offer: while an accessory is listed after the object to which it relates, using the same progressive number but showing itself as a separate item, with its own price, an embedded accessory is not a separate item but is appended at the description of the object to which it belongs, so it does not appear as something sold separately. Even the price, in this case, is directly added to that of the object, without prejudice, however, the various discounts and recharges of the object and the embedded accessory.

Actually, this is simply a flag, and the implementation of this functionality must be provided by the custom VBA program in Word or Excel; by the way, is useful to set this option at design time, specially if the selected accessory should be treated differently from all other accessories.

In the main listbox there is the list of all the accessories which belong to the currently selected list. The added accessories display, at the right side of their description, the quality (with the possible unit of measure), the price (eventually discounted) and, if they are embedded, an asterisk.

Modifies Window



Field	Value	Mod.
CODICE	BCB2136	
CODORFOR	BCB2136	
MODELLO	BCB2136	
DESCRBRV	CAPPA CENTRALE con FILTRI a LABIRINTO	
DESCRBRVUK	CENTRAL TYPE EXHAUST HOOD	
DESCRBRVF	HOTTE DE ASPIRATION avec FILTRE LAB...	
DESCRBRVD		
DESCRBRVE		
DESCRLNG	MEMO	
DESCRLNGUK	MEMO	
DESCRLNGF	MEMO	
DESCRLNGD	MEMO	
DESCRLNGE	MEMO	
PREZZO	2220	

Value: CENTRAL TYPE EXHAUST HOOD

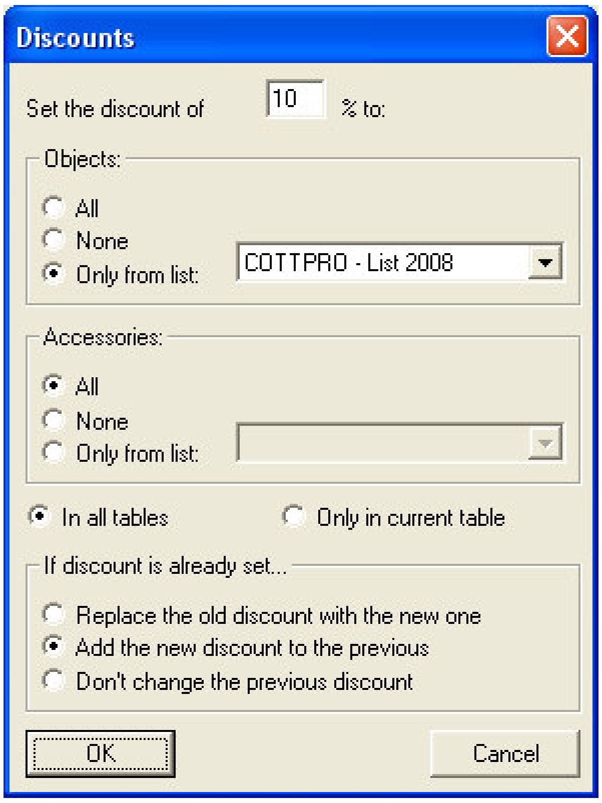
Default: CENTRAL TYPE EXHAUST HOOD

In this window you can view and eventually change all the data stored in the database concerning the current object. The main listview window shows all the fields present in the database table with their actual values, that can be selected one at a time. The fields that have been changed are indicated by an asterisk placed in the column at the end of the row.

Value: In this box is copied the current value of the selected field in the listview, so that it can be changed. The changes becomes effective only after pressing the button **Set**, so that the value in the listview is updated. If the field is a MEMO field (a text with multiple lines), the edit box expands as appropriate.

Default: In this box is copied the original value stored in the database (that cannot be changed from this window). If the field a MEMO field (a text with multiple lines), the edit box expands as appropriate. To set the default value in a previously changed field, you can copy the text of the **Default** edit box, selecting it and pressing the CONTROL+C buttons, and then paste it in the **Value** edit box pressing the CONTROL+V buttons. To apply the change, it is necessary to press the **Set** button. In this way, the object modification is removed and the original value of the database is restored, as shown by the disappearance of the asterisk at the end of the selected row in the listview window.

Discount Window


Discounts

Set the discount of 10 % to:

Objects:
All None [✓ Only from list:] COTTPRO - List 2008

Accessories:
[✓ All] None Only from list:

[✓ In all tables] Only in current table

If discount is already set...
Replace the old discount with the new one
[✓ Add the new discount to the previous]
Don't change the previous discount

OK Cancel

Through this window, you can set and change discounts to objects and accessories throughout the report or to those belonging to current section table, according to the selected criteria. The current section table is the one that contains the selected line in the report window; if the selected line is the header, the first report table is taken as the current.

Set the discount of. In this edit box must be specified the discount percentage that will be applied. The number may be higher or lower than 0 (in this case the discount will be quite a charge).

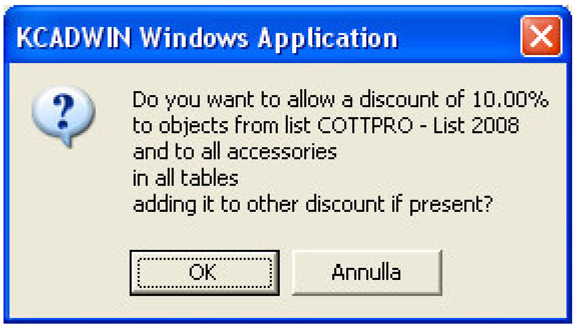
Objects: Select one of the buttons **All**, **None**, **Only from list** if you want to apply the discount respectively to the objects of all report (or of the current section table), to no object at all (to set the discount only to accessories), or only to the objects belonging to the furniture list specified in the dropdown box aside, which is enabled only if you select this option (ever in the whole report or in the single current table).

Accessories: Select one of the buttons **All, None, Only from list** if you want to apply the discount respectively to the accessories of all report (or of the current section table), to no accessory at all (to set the discount only to the objects), or only to the accessories belonging to the furniture list specified in the dropdown box aside, which is enabled only if you select this option (ever in the whole report or in the single current table).

Option buttons **In all tables** and **Only in current table**: Select the first button if you want that function will examine the objects and the accessories of the whole report to verify if they meet the requirements set and apply the discount, or otherwise select the second button if you want the discounts are applied (always according to the criteria selected) only to objects and accessories that belong to the current table (the one that contains the selected row in the report window).

If discount is already set... Select one of the available option buttons to determine the program behaviour if they are found existing discounts for objects and accessories to which you should apply the new discount. If you select **Replace the old discount with the new one**, the existing discount will be eliminated and the new one will take its place. If you select **Add the new discount to the previous**, both the old and new discount will be combined, resulting in higher discount; in this case, it should be noted that applying a 10% discount to an object that has already had a 20% discount, you don't get a 30% discount but a 28% discount, because the new 10% discount is applied only to the 80% of the original price (because the previuos 20% discount). If you select **Don't change the previous discount**, the new discount will be applied only to the objects and accessories that result without any discount, but not to those that have an existing discount.

Pressing OK after making your selections, a dialog box appears that summarize the set conditions, and prompts for further confirmation. If you press OK again, the discount is applied with the given criteria; if you press Cancel, you come back to the discounts dialog box, where you can change the settings. For example, pressing OK after selecting the options las the dialog box shown above, you will get a confirmation dialog box like the following:


KCADWIN Windows Application

Do you want to allow a discount of 10.00%
to objects from list COTTPRO - List 2008
and to all accessories
in all tables
adding it to other discount if present?

PROGRAMS TO GENERATE LINKED DOCUMENTS

GENERAL DESCRIPTION

Once created one or many Kcad report files with the AutoCAD add-in application or with Kcadwin Windows application, It’s possible to use a Word or Excel VBA macro to produce the relative offers and the other related documents and print them.

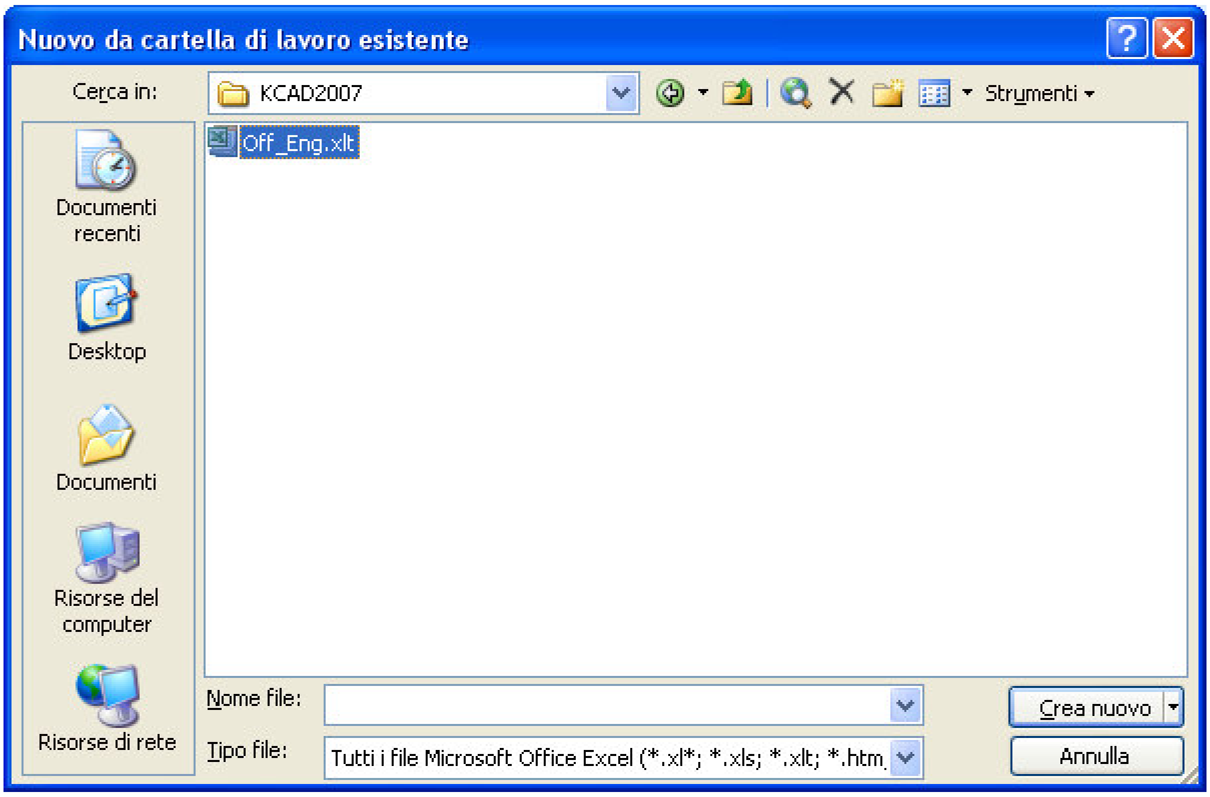
The operation is based on Word or Excel template files that contain customized code written in Visual Basic Application. All the specific Kcad functionalities are provided through the KCAD32.DLL library file installed in the Windows system folder, that allows to access all the data recorded in Kcad report files and in all the databases.

To illustrate the operation of this type of custom programs, we examine the functioning of an application to make an offer based on an Excel spreadsheet, which is provided with the Kcad application suite as an example, but that could be adapted by the end user with a few simple changes.

To use other applications to compose documents that have an internal programming language capable of using the routine of a Windows library, refer to the information provided in the technical documentation in the appendix of this manual.

EXAMPLE PROGRAM TO GENERATE THE OFFER

To produce an offer using the Excel tempate provided as an example with the program, you must start Excel and choose menu item File New..., and then the option of selecting a file from an existing folder. In the dialog that opens, you must navigate to the Kcad installation folder and select the template Off_eng.xlt, and then press the OK button. This creates a new document based on the selected template.



Note: Copingy the template file in the Office default template folder, it will become available through the option of selecting the template on the local computer.

Alternatively, you can open the "Windows Explorer" application, navigate to the Kcad installation folder, and then double-click the file name "Off_eng.xlt". In this way, Excel will load the file "Off_eng1.xls" based on the selected template.

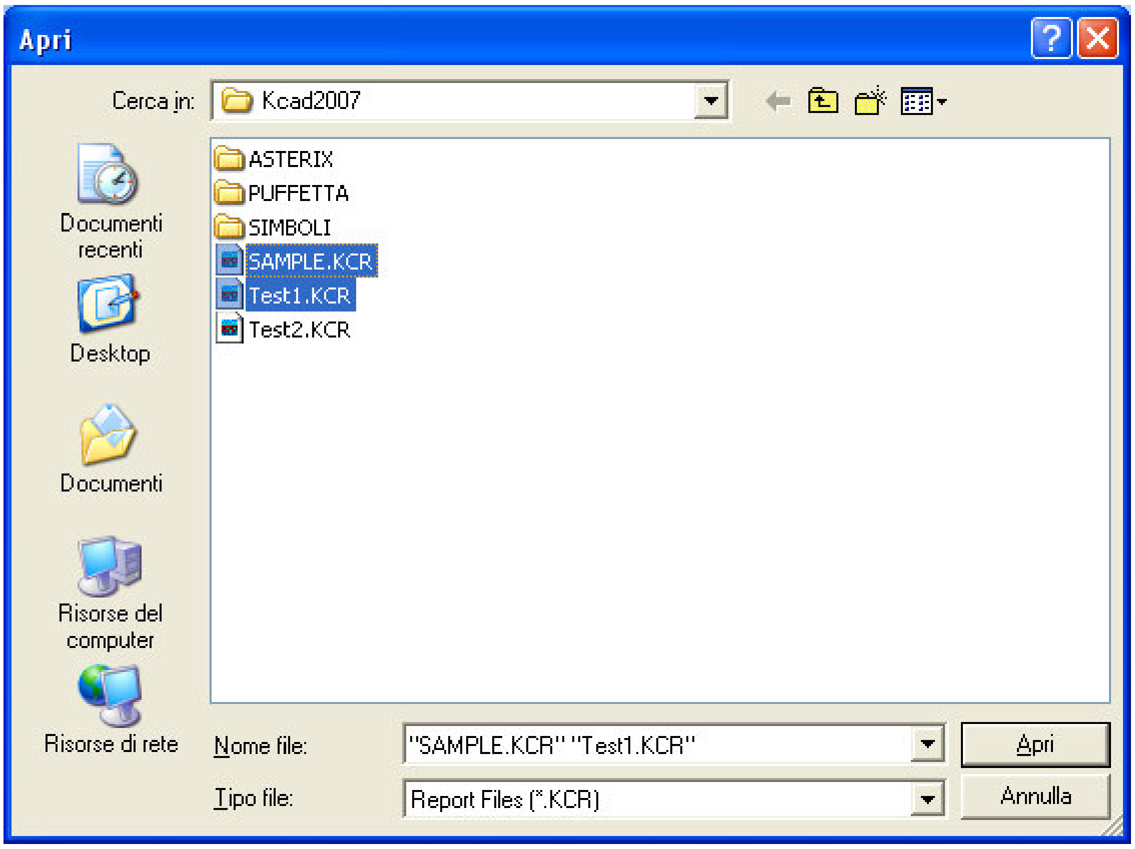
The opened file is not the selected template, but just a copy, and therefore can be edited and saved without affecting the template itself.

The sample file consists of two pages: "Header" where are the customer data and conditions of supply, and "Offer" that must be compiled with the list of equipment in the project.

In the header page, there is the "IMPORT" button. When pressed, the VBA macro "Kcad" is called and executed. The source code is viewable through the menu "Tools->Macro->Visual Basic Editor" to use it as an example for writing a macro to compile a document based on different templates.

Once you click the button to activate the program, a window appears that lets you select one or more KCAD report files with KCR extension. To select multiple files, simply hold down the Control key while you click it with the mouse, or if the file names to select are adjacent to each other, select the first and then hold down the Shift key while you select the last with the mouse.

You are advised to prepare all the files you wish to select together in the same folder, although you can select multiple files from different folders, but in this case you must manually specify the names of the full file path, because if you try to change folder after selecting one or more files, the selection is reset.



Cerca in: Kcad2007

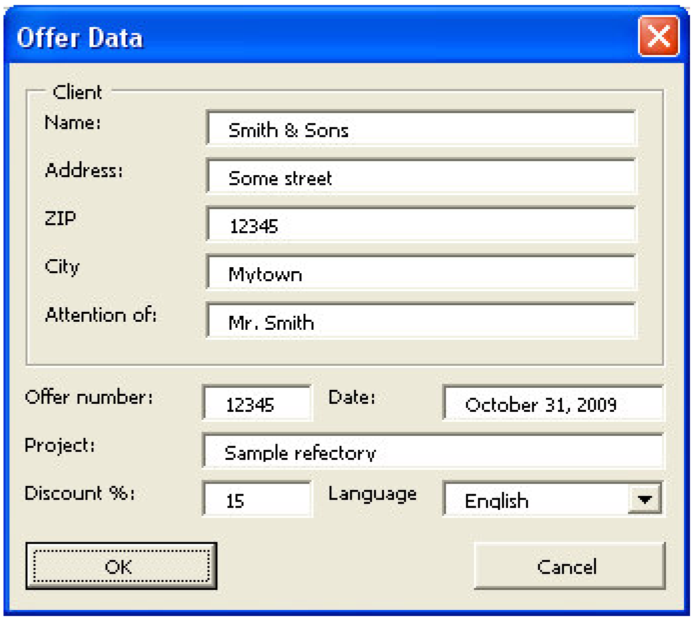
ASTERIX
PUFFETTA
SIMBOLI
SAMPLE.KCR
Test1.KCR
Test2.KCR

Nome file: "SAMPLE.KCR" "Test1.KCR"

Tipo file: Report Files (*.KCR)

This window for the multiple selection of Kcad report files is called through a specific function of KCAD32.dll library, which returns the list of names of the selected files. With other specific library functions, you can access in read-only mode to all the data contained in each report. The details for the use of KCAD32.dll functions are described in the Appendix, while the sample code can be viewed with the VBA editor of Excel.

After selecting the report files, another window is opened, that allows you to specify some information and options. This window is designed in VBA editor, because it must be adaptable to the needs of each document. For example, in this case, it restores some of the data read in the report, and allows you to specify the ZIP code and the city, an additional discount percentage, and the language used to describe the listed items.


Offer Data

Client

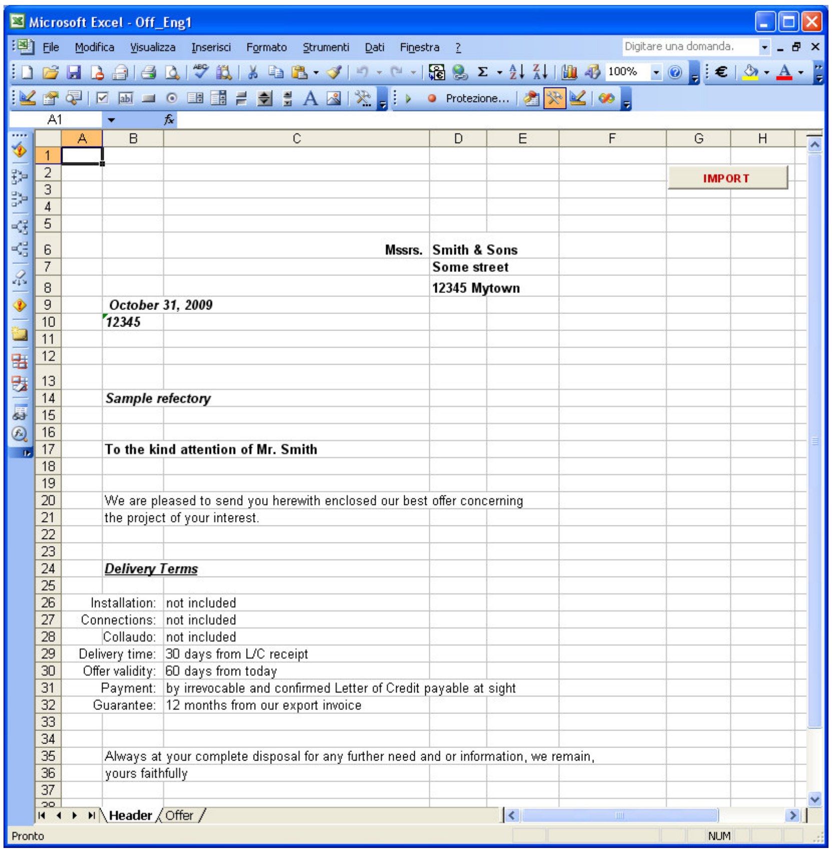
Name:	Smith & Sons
Address:	Some street
ZIP	12345
City	Mytown
Attention of:	Mr. Smith

Offer number:	12345	Date:	October 31, 2009
Project:	Sample refectory		
Discount %:	15	Language	English

Discount box allows you to set a further discount to be applied to the offer total price, which can be used as an alternative to other discounts that can be set with the features of other applications. This discount is applied using the Excel formulas, so that it can be changed even after the initial document compilation.

Closing the window with OK, the information contained in the header of all Kcad report files selected is updated with new data, using a special feature of the library KCAD32.dll which is the only one that allows you to change the data in a file KCR. This allowed for uniform differencies between file reports that are used together in the same offer.

In the pictures below is shown the result of the compilation of the offer. Using VBA macros, you can integrate the information available through the Kcad library functions with any other specific requirement, as the recording of data in another database for administration.


IMPORT

Mssrs. Smith & Sons
Some street
12345 Mytown

October 31, 2009
12345

Sample refectory

To the kind attention of Mr. Smith

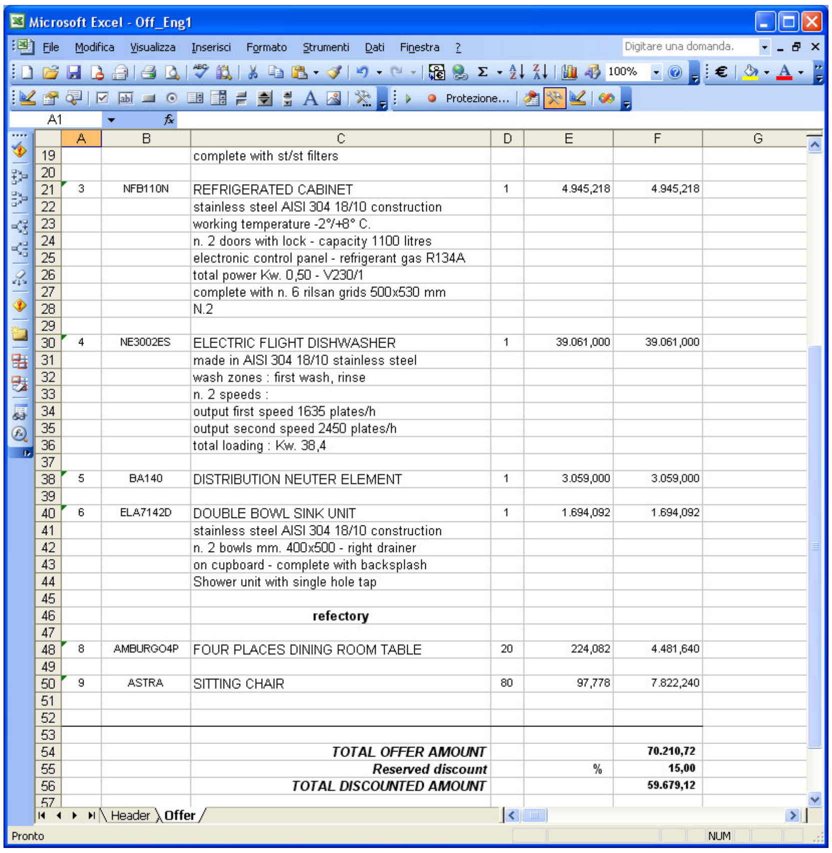
We are pleased to send you herewith enclosed our best offer concerning
the project of your interest.

Delivery Terms

Installation:	not included
Connections:	not included
Collaudo:	not included
Delivery time:	30 days from L/C receipt
Offer validity:	60 days from today
Payment:	by irrevocable and confirmed Letter of Credit payable at sight
Guarantee:	12 months from our export invoice

Always at your complete disposal for any further need and or information, we remain,
yours faithfully

The header page contains the same information specified in the dialog above, plus other data already in place that are assume as independent from the offer contents.



	A	B	C	D	E	F
19			complete with st/st filters			
20						
21	3	NFB110N	REFRIGERATED CABINET	1	4.945,218	4.945,218
22			stainless steel AISI 304 18/10 construction			
23			working temperature -2°/+8° C.			
24			n. 2 doors with lock - capacity 1100 litres			
25			electronic control panel - refrigerant gas R134A			
26			total power Kw. 0,50 - V230/1			
27			complete with n. 6 rilsan grids 500x530 mm			
28			N.2			
29						
30	4	NE3002ES	ELECTRIC FLIGHT DISHWASHER	1	39.061,000	39.061,000
31			made in AISI 304 18/10 stainless steel			
32			wash zones : first wash, rinse			
33			n. 2 speeds :			
34			output first speed 1635 plates/h			
35			output second speed 2450 plates/h			
36			total loading : Kw. 38,4			
37						
38	5	BA140	DISTRIBUTION NEUTER ELEMENT	1	3.059,000	3.059,000
39						
40	6	ELA7142D	DOUBLE BOWL SINK UNIT	1	1.694,092	1.694,092
41			stainless steel AISI 304 18/10 construction			
42			n. 2 bowls mm. 400x500 - right drainer			
43			on cupboard - complete with backsplash			
44			Shower unit with single hole tap			
45						
46			**refectory**			
47						
48	8	AMBURGO4P	FOUR PLACES DINING ROOM TABLE	20	224,082	4.481,640
49						
50	9	ASTRA	SITTING CHAIR	80	97,778	7.822,240
51						
52						
53						
54			***TOTAL OFFER AMOUNT***			**70.210,72**
55			***Reserved discount***		%	**15,00**
56			***TOTAL DISCOUNTED AMOUNT***			**59.679,12**

The offer page contains all of the objects of all tables in all the reports selected. For each object, is displayed the long description that spans multiple consecutive rows. In the case of embedded accessories, additional lines are added to the description, while the price is added to that of the object. Note: because the quantity of accessories is multiplied by the quantity of objects, in this offer tamplate embedded objects must be used only in objects of unitary quantity.

This offer template is fairly generic to be used as a base for your custom templates, but if the document compilation must follow some special rules, you must modify the existing VBA code or write a new VBA macro using the functions in the KCAD32.dll library documented in the appendix of this manual.

SYSTEM MAINTENANCE

The KCAD data are stored in Microsoft Access database, which are files with ".MDB" extension, and in "DWG" and "SLD" file for graphic representation in AutoCAD.
Note: the table and field names are often in italian to maintain database compatibility beetween italian and international versions of the software.

DATA FILE LIST

File KCAD.MDB with table “LISTINI”, “ACCESSOR” and “VALUTE”
File .MDB for each list, with a table of the same name containing the divisions list, the divisions tables and eventually one or more accessories lists.
File .DWG for the 2d and 3d blocks of each list
File .SLD for the AutoCAD slide for each block
File .DWG for mark balloons symbols
File .DWG for plugs symbols
File .DWG for riepilogative table column header

KCAD AND DATABASE DIRECTORIES

To operate correctly, the KCAD applications must access a large amount of data, with all the necessary information for each list used. To create new lists and modify existing lists, you must use Microsoft Access.

The data of sample lists are installed in the same folder as the program file, but if exists a folder named "C:\KCADDATA", lists placed in that are used. To use a different folder, for example to share the database on a local network, you must use the environment variable "KCADDATA", set to indicate the folder where KCAD need to look for the lists. For example:

```
KCADDATA = F:\LISTS
```

The environment variables can be set in the window accessible through the Control Panel, selecting the icon "System", then the “Advanced” tab and the "Environment Variables" button. In the window that appears, you can create "user variables" and "system variables". Since the system variables are accessible to all users, you must have administrator rights to add "KCADDATA" in this group of variables, but in this case the setting will be valid for all computer users.

“KCAD.MDB” MAIN DATABASE STRUCTURE

The main database that KCAD uses is KCAD.MDB, which contain the three tables "LISTINI", "ACCESSOR" and "VALUTE".

The table "VALUTE" is used to convert the list prices that are not already expressed in the system currency which is the Euro. In addition to the currency symbol, is indicated the currency description, the value corresponding to 1 Euro, and the digits of precision to be specified. By default, the Euro is specified with three digits of precision, which correspond to the thousandths of a Euro, but if you prefer to see prices with only two digits of precision, which correspond to the cents, just set 2 in the field of precision.

This table can be updated with changes in stock, but please note that the changes affect all the prices of all lists that use the currency changed. To "freeze" a list, it may be convenient to use a SQL Access query to get a copy that uses only the system currency, using the desired conversion rate during the conversion.

The table “LISTINI” contains general information about all the available furniture lists, and specifies what subfolders and database search for relevant information. In practice it works as index to all the available furniture lists.

The columns of the table "LISTINI" are as follows:
The field "TABELLA", table name and database in which are present all the information of the list. Must match the name of a subfolder and an MDB database file contained within it, which must contain a table with the same name.
The field "LISTINO", the italian description of the lists, which must be at most 63 characters long, used by the application to display it in the list of available furniture lists.
The field "LIST", the english description of the list, long at most 63 characters, used by the international version of Kcad. It was planned to allow you to use the same database format used by the Italian version.
The "VALUTA" with the abbreviation of the currency used for the list prices, which must be listed in the table "VALUTE" described above.
The "SCONTO", which indicates the percentage discount on the list price applied by the manufacturer that allows you to calculate the actual cost of the items.
Field "RICARICO", which indicates the percentage of recharge to be applied to get the sale price. The rate of charge is applied to the cost obtained by applying the discount to the previous list price of the supplier, so if an object has a list price of 100 Euro, applying a 20% discount and then a 20% recharge it’s not achieved again 100 Euro but 96 Euro (equivalent to 100 * 0.8 * 1.2).

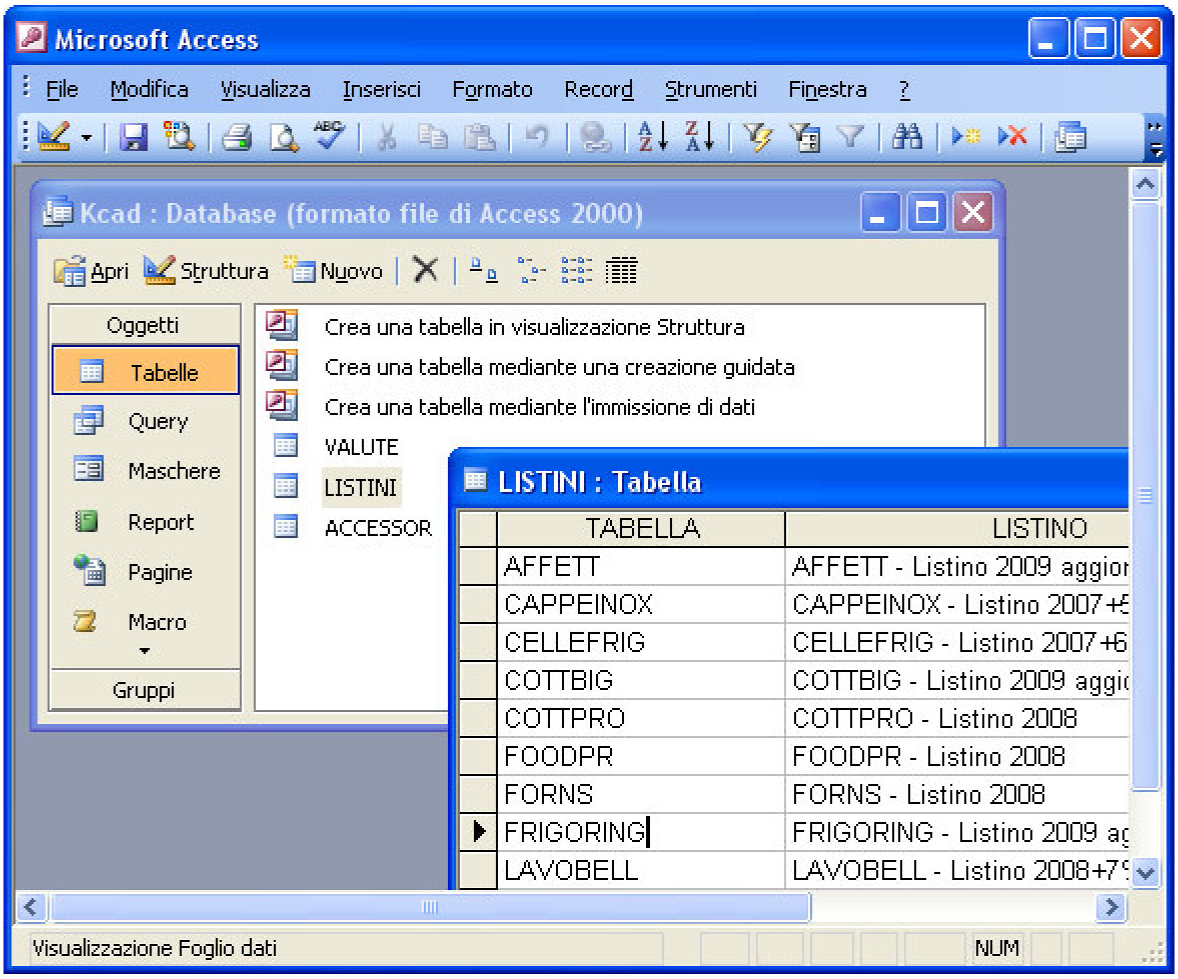

Microsoft Access
File Modifica Visualizza Inserisci Formato Record Strumenti Finestra ?
Kcad : Database (formato file di Access 2000)
Apri Struttura Nuovo
Oggetti
Tabelle
Query
Maschere
Report
Pagine
Macro
Gruppi
Crea una tabella in visualizzazione Struttura
Crea una tabella mediante una creazione guidata
Crea una tabella mediante l'immissione di dati
VALUTE
LISTINI
ACCESSOR
LISTINI : Tabella
Visualizzazione Foglio dati
NUM

TABELLA	LISTINO
AFFETT	AFFETT - Listino 2009 aggior
CAPPEINOX	CAPPEINOX - Listino 2007+5
CELLEFRIG	CELLEFRIG - Listino 2007+6
COTTBIG	COTTBIG - Listino 2009 aggio
COTTPRO	COTTPRO - Listino 2008
FOODPR	FOODPR - Listino 2008
FORNS	FORNS - Listino 2008
FRIGORING	FRIGORING - Listino 2009 ag
LAVOBELL	LAVOBELL - Listino 2008+7

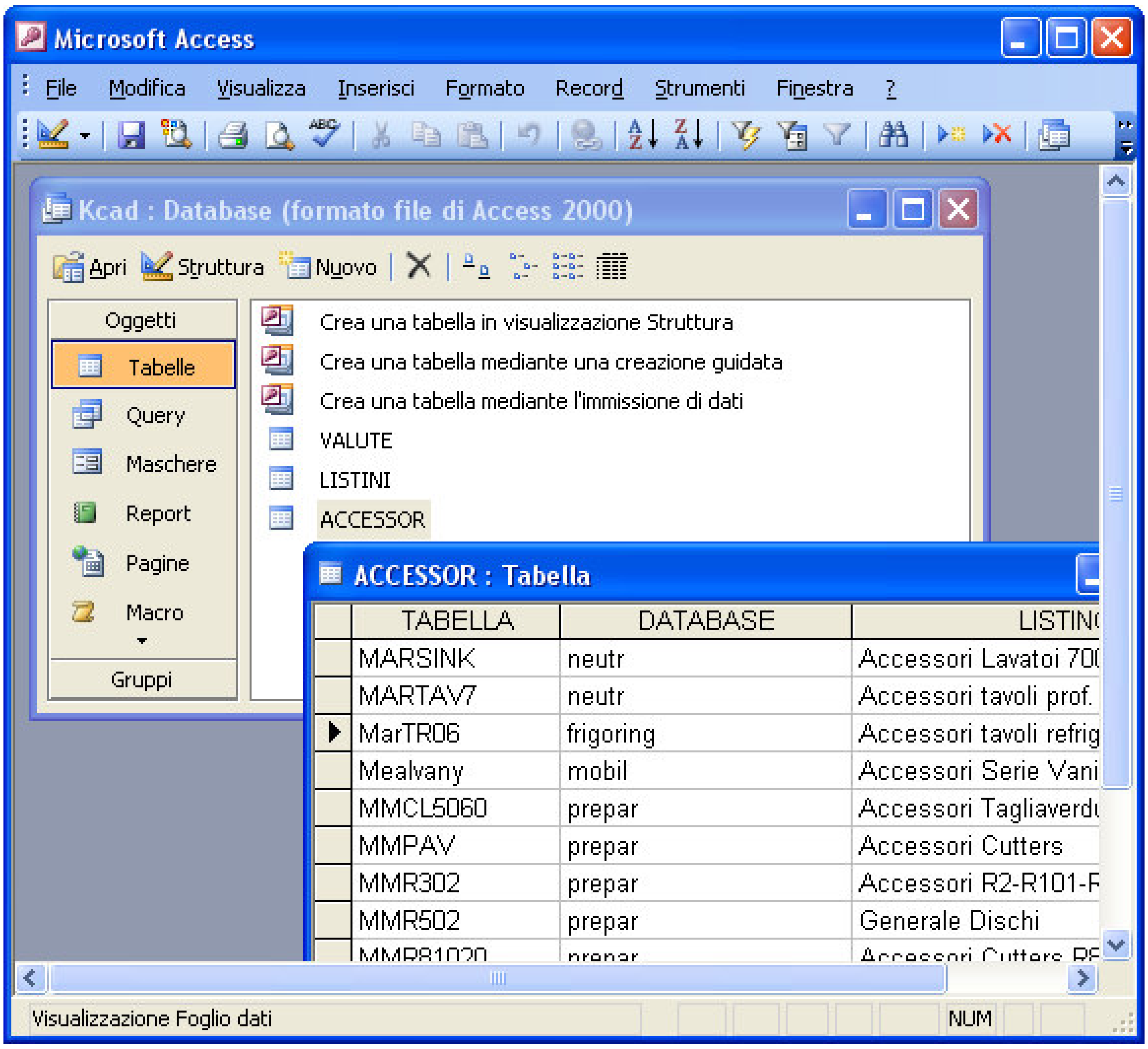
Discounts and charges set in this table are valid for the whole list, but as we shall see, in each furniture list database is present a table of the list divisions in which you can specify a different discount and a different charge for each division in the list (see below the description of general furniture list database).

If you need to change a data, for example the discount percentual of "COTTPRO" furniture list, you must start ACCESS, load th KCAD.MDB database, open the “LISTINI” table, move to the row and column of the data to edit, and insert the new data.

To insert a new list, you must open the “LISTINI” table, move to the last line (which is always empty to allow the insertion of new data), enter the table name, a description of the list, its currency, its discount and recharge percentual. Also, you must create the corresponding folder that contains the new furniture list database file, and all drawings and slides of AutoCAD required, as explained below.

In addition to these fields, the table “LISTINI” may contain other information that may be useful for completing the offer and related documents, such as the producer company name or an identifier that can serve as a foreign key to a table of anagraphical data in another database. Such information does not affect the working the Kcad application suite.

The "ACCESSOR" table contains the list of accessories lists. The accessories are different from objects in that they are not represented graphically in the AutCAD drawing, and therefore do not require to have any block or slide, but they must be specified together with some object, even if it is possible that the accessory belongs to a different producer other than that of the object.



Crea una tabella in visualizzazione Struttura
Crea una tabella mediante una creazione guidata
Crea una tabella mediante l'immissione di dati
VALUTE
LISTINI
ACCESSOR

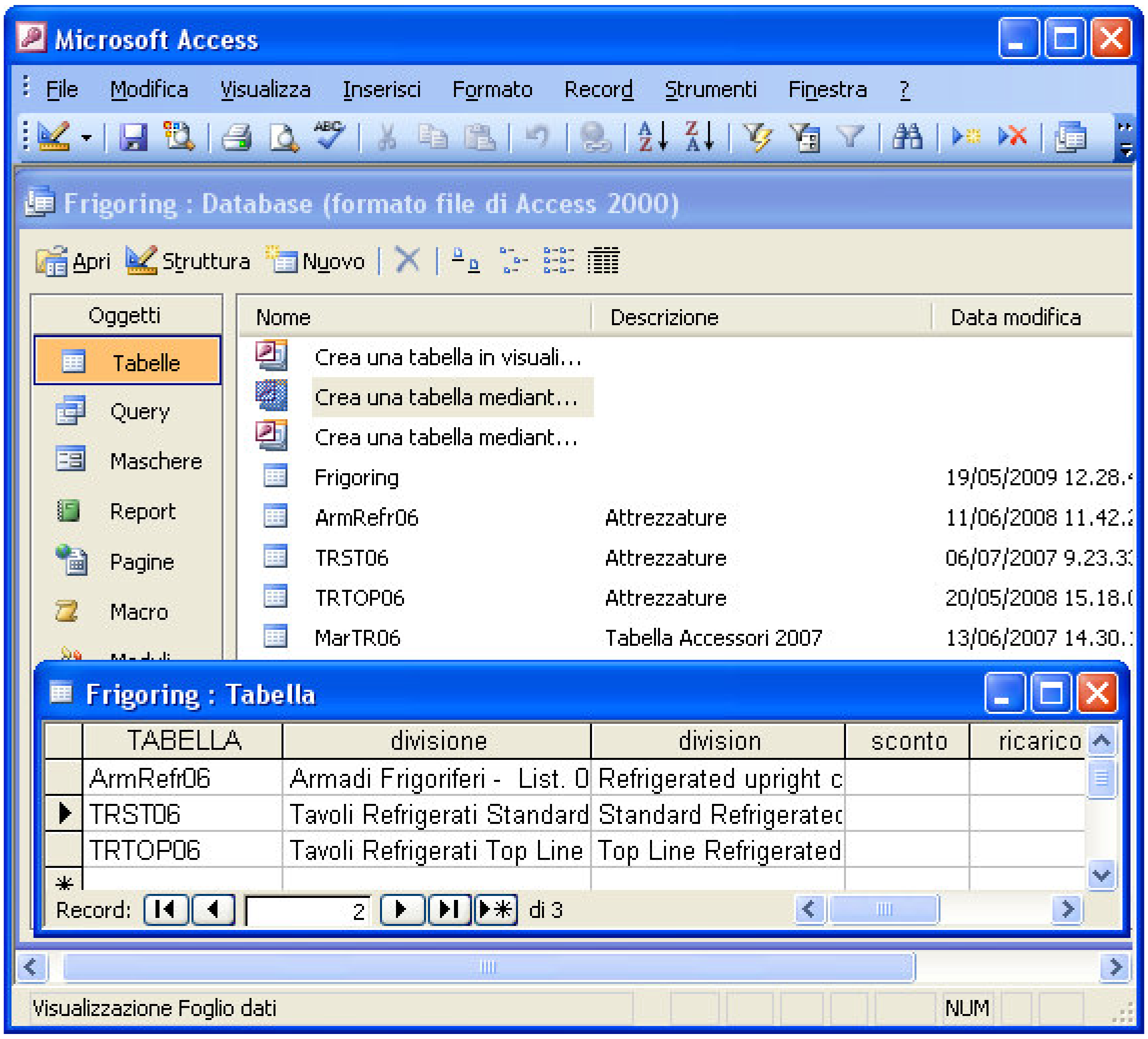
TABELLA	DATABASE	LISTIN
MARSINK	neutr	Accessori Lavatoi 70
MARTAV7	neutr	Accessori tavoli prof.
MarTR06	frigoring	Accessori tavoli refrig
Mealvany	mobil	Accessori Serie Vani
MMCL5060	prepar	Accessori Tagliaverd
MMPAV	prepar	Accessori Cutters
MMR302	prepar	Accessori R2-R101-F
MMR502	prepar	Generale Dischi
MMR81020	prepar	Accessori Cutters RF

The "ACCESSOR" table contains essentially the same fields seen for the furniture lists, with two main differences:
The field "DATABASE" is the name of the subfolder and of the database file that contains the list of accessories. In general, this database is a furniture list of object already listed in the table "LISTINI", that also contains some tables of accessories.
The field "TABELLA" is the name of a table of accessories within the specified database.
The fields "LISTINO", "LIST", "VALUTA", "SCONTO" and "RICARICO" have the same function as those found in the table "LISTINI".

GENERAL FURNITURE LIST DATABASE STRUCTURE

Each furniture list is composed by an MDB database with the same name shown in field "TABELLA" in the table “LISTINI” seen above, that resides in a subfolder with the same name. That contains more subfolders, one for each division of the list, which contain the 2d and 3d blocks with DWG extension, the slides, with extension SLD, for display in AutoCAD, and any other optional file for objects documentation.

Consider the example of “Frigoring” furniture list. As can be seen from the table “LISTINI” seen above, "Frigoring - 2009 updated List" is the english description of the list in the "LIST" field, but the referenced name is "FRIGORING," in the field "TABELLA". This means that if the KCAD data are in "C:\KCADDATA" folder, the main database will be "C:\KCADDATA\KCAD.MDB", and the list of "Frigoring" should stay in the "C:\KCADDATA\FRIGORING" folder. Inside this folder and its subfolders will reside all the files that make up the list. The main file of the list is the database "C:\KCADDATA\FRIGORING\FRIGORING.MDB" which must contain a table called "FRIGORING" with the list of all the divisions of the furniture list. For each division, there is another database table and a another subfolder with all the corresponding AutoCAD files. In our example, the division table is named "FRIGORING", and is made as follows:



Nome	Descrizione	Data modifica
Crea una tabella in visuali…		
Crea una tabella mediant…		
Crea una tabella mediant…		
Frigoring		19/05/2009 12.28.
ArmRefr06	Attrezzature	11/06/2008 11.42.
TRST06	Attrezzature	06/07/2007 9.23.3
TRTOP06	Attrezzature	20/05/2008 15.18.
MarTR06	Tabella Accessori 2007	13/06/2007 14.30.

TABELLA	divisione	division	sconto	ricarico
ArmRefr06	Armadi Frigoriferi - List. 0	Refrigerated upright c		
TRST06	Tavoli Refrigerati Standard	Standard Refrigeratec		
TRTOP06	Tavoli Refrigerati Top Line	Top Line Refrigerated		

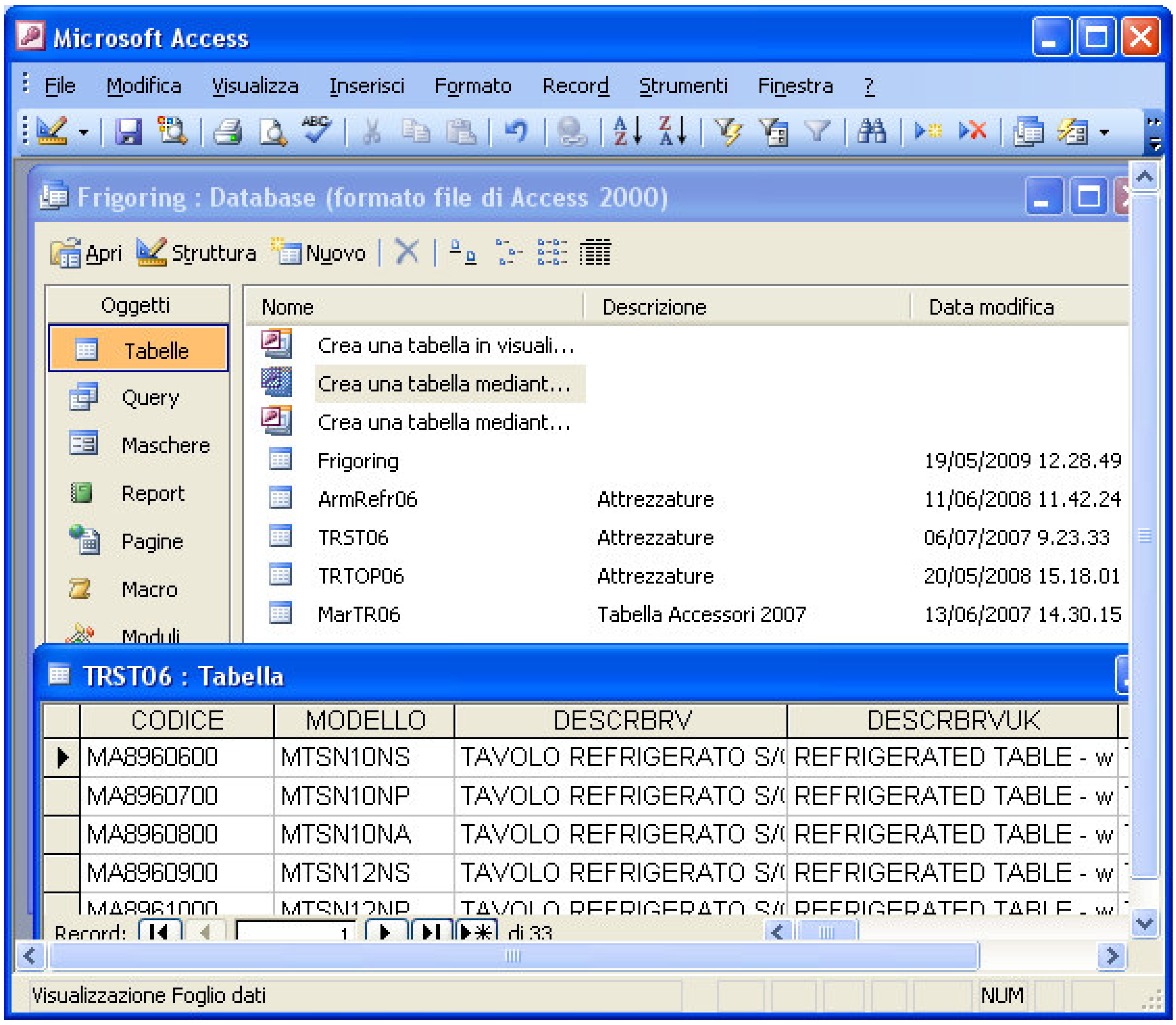
In the column "TABELLA" must be specified the short name of the division, which is used for the corresponding table in the same database, and the subfolder containing the graphic files for AutoCAD. In the "DIVISIONE" and "DIVISION" shall specify the extended descriptions for division in Italian and English, as they appear in the list of divisions which is shown by the italian and international version of Kcad.

In the "SCONTO" and "RICARICO" may be indicatet the discount and recharge percentage, if for this division must not be applied the same settings made in the table "LISTINI" in "KCAD.MDB database, as previously seen. For example, if the list

"Frigoring" had stated as 40% discount and 60% as recharging the “LISTINI” table, but in this table was given a different discount or charge for the division “ArmRefr06”, for it would be applied the new percentages shown here and for all other divisions of the list will remain valid percentages given in the table "LISTINI" of "KCAD.MDB.

For each division of the list, should exist a subfolder of the same name mentioned in the column "TABELLA", which must contain the required AutoCAD file for that division and any other ausiliary files. In our example, the folder "C:\KCADDATA\FRIGORING" must contain all the subfolders of the divisions: "C:\KCADDATA\FRIGORING\ArmRefr06", "C:\KCADDATA\FRIGORING\TRST06" and "C:\KCADDATA\FRIGORING\TRTOP06 ", each containing blocks, slides and other files related to objects belonging to it.

For each division, there is a table of the same name, which contains the information for each single object. They all have the same structure, but the fact that they are maintained in independent tables facilitates the management and any updates.


Microsoft Access
File Modifica Visualizza Inserisci Formato Record Strumenti Finestra ?
Frigoring : Database (formato file di Access 2000)
Apri Struttura Nuovo
Oggetti
Tabelle
Query
Maschere
Report
Pagine
Macro
Moduli
TRST06 : Tabella
Record: 1 di 33
Visualizzazione Foglio dati
NUM

Nome	Descrizione	Data modifica
Crea una tabella in visuali…		
Crea una tabella mediant…		
Crea una tabella mediant…		
Frigoring		19/05/2009 12.28.49
ArmRefr06	Attrezzature	11/06/2008 11.42.24
TRST06	Attrezzature	06/07/2007 9.23.33
TRTOP06	Attrezzature	20/05/2008 15.18.01
MarTR06	Tabella Accessori 2007	13/06/2007 14.30.15

CODICE	MODELLO	DESCRBRV	DESCRBRVUK
MA8960600	MTSN10NS	TAVOLO REFRIGERATO S/(	REFRIGERATED TABLE - w
MA8960700	MTSN10NP	TAVOLO REFRIGERATO S/(	REFRIGERATED TABLE - w
MA8960800	MTSN10NA	TAVOLO REFRIGERATO S/(	REFRIGERATED TABLE - w
MA8960900	MTSN12NS	TAVOLO REFRIGERATO S/(	REFRIGERATED TABLE - w
MA8961000	MTSN12NP	TAVOLO REFRIGERATO S/(	REFRIGERATED TABLE - w

The first field “CODICE”, is the code of the object. The code can consist of numbers and letters, its maximum length is 15 characters and must be unique in the whole list, not only within its own division. This simplifies the identification of the object when it is listed in the offer or in other documents. Are not considered differences between upper and lower case letters. This code must always be specified, as it allows to identify objects.

The second field “MODELLO”, is the model of the object. Like the code, may not exceed 15 characters and does not distinguish capital letters from lowercase. Unlike the Code, most objects can have the same model. This field has been given because there are encodings of the producer companies that differ from those used by KCAD users. The code and the model may be the same, and whether an object has the model field empty, the functions of the program automatically uses the code in its place.

The field DESCRBRV and is the italian short description at most 75 characters long. This description allows to choose the right object when several objects have the same graphical representation, and it is displayed also in the summary table of the drawing. It can also be used by custom programs for compilation of the offer or other documents, as an alternative to long description described below. In international version of Kcad, the same role is covered by DESCBRVUK.

Fields DESCRBRVUK, DESCRBRVF, DESCRBRVD, and DESCRBRVE are versions respectively in English, French, German and Spanish DESCRBRV; they are used alternatively to Italian description when you need to produce documentation in other languages. The document language setting does not depend on the version of KCAD, which may be only in Italian or English (international version) Both versions can produce the documentation and the other offerings in each of the five languages.

Fields DESCRLNG, DESCRLNGUK, DESCRLNGF, DESCRLNGD and DESCRLNGE contain a detailed description of the object on multiple lines, in Italian, English, French, German and Spanish language. These long descriptions can be used in the production of the offer, and list the characteristics of the object. The first line is used as a header, and for clarity should be similar to the short description. This field is a MEMO, which means it has no limits in length and allows you to enter multiple lines. To produce a new line in Access without leaving the field, type CONTROL+ENTER instead of just ENTER. However, to display issues and offer formatting, KCAD change the display of lines longer than 64 characters. You should not exceed this length for each line at the time of creation.

The field “PREZZO” contains the price of object as specified by productor list, in the currency specified in the table "LISTINI" of KCAD.MDB.

The field “SLIDE” contains the name of the AutoCAD slide, which is to be displayed in the menu icon that allows the object selection, which must not exceed 64 characters. This name is important because the name of the 2d and 3d blocks corresponding to the object are derived automatically from this by replacing the last letters "0R" with "2R" and "3R".

The “DWG” field was added to allow you to share the same slide for blocks that have in fact a graphical representation only slightly different, or may be different only in three-dimensional version. This allows you to specify the same slide to insert different blocks. For more details on the fields "SLIDE" and "DWG", see "ADDITION OF A NEW OBJECT IN A LIST" later in this chapter.

The field “ACCESSORI” indicates the name of the default accessory list for the object. If it is empty, the object does not include accessories, but again, you can add accessories to any list, and you can specify accessories belonging to lists different from the default, or more accessories from different lists. The accessory list specified here only determines which accessory list is offered automatically when placing the object in the drawing.

The field “DIMENSIONI” shows the size of the object in millimeters, with the numbers separated from one another with an "x" in lowercase. In the windows for the object selection, this size is shown following the description, as a further distinction between similar objects. Moreover, it is shown in a special column in the summary table generated in the drawing. The "BULKHEAD" function consider the third number (after the second "x") as the height of the object.

The fields “ATTACCHI” e “PLUGS” are MEMO fields, in Italian and English language, where in several lines are described the features and the consumption of different technological plugs that the object has. For each plug there is a line of description, which must follow the syntactic conventions explained in "PLUGS CODIFICATION" later in this chapter.

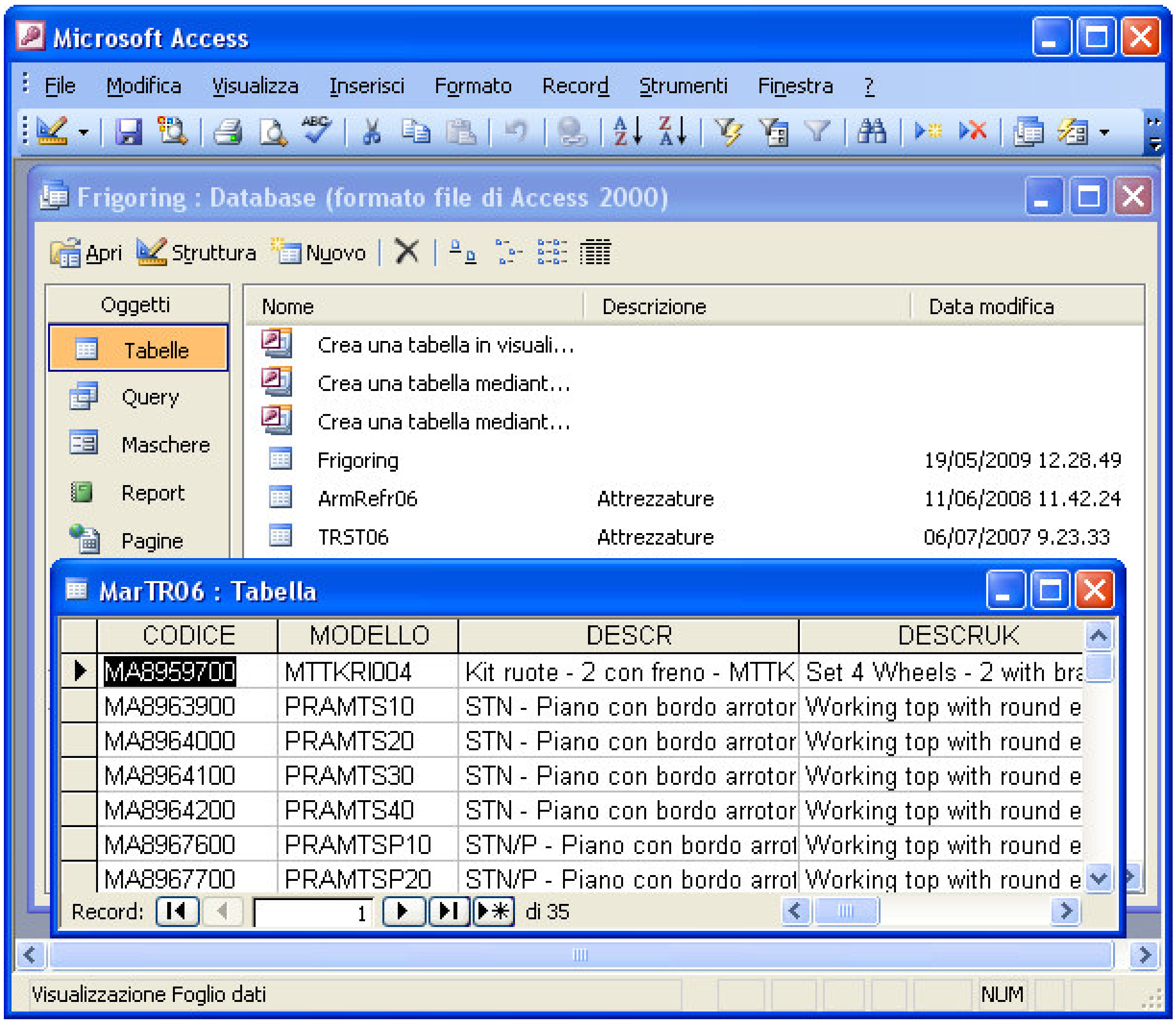
Other fields can be added by users, such as weight or the anme of a technical paper that could be printed with the offer. KCAD not use the information added, but they are displayed in the windows to modify the object and it’s possible to change their values. These data can be used during automatic compilation of offers and related documentation, using the generic functions described in the technical documentation reported in Appendix.

ACCESSORIES TABLES

The lists of accessories are listed in the table "ACCESSOR" of KCAD.MDB, and each list resides in the database specified in the "DATABASE" field. In general, the database specified is the same one that contains a list of objects, but you can also use databases containing only one or more lists of accessories. The name of the table containing the data of the accessories is specified in the "TABELLA" field. In the sample data, the list of accessories "MarTR06" is contained in the database "Frigoring", and therefore in the file "C:\KCADDATA\FRIGORING\FRIGORING.MDB".

The list of accessories are not split into divisions, but may be more than one within the same database. If it is considered useful, you can combine them into a single database containing lists of accessories. The table that lists the names and the percentage discount and recharge is always the "ACCESSOR" table in KCAD.MDB database, as described above.

Tables of accessories do not need references to AutoCAD graphics files or data for plugs and their consumption, so that their format is simpler than that of the tables of objects.


Microsoft Access
File Modifica Visualizza Inserisci Formato Record Strumenti Finestra ?
Frigoring : Database (formato file di Access 2000)
Apri Struttura Nuovo
Oggetti
Tabelle
Query
Maschere
Report
Pagine
MarTR06 : Tabella
Record: 1 di 35
Visualizzazione Foglio dati
NUM

Nome	Descrizione	Data modifica
Crea una tabella in visuali…		
Crea una tabella mediant…		
Crea una tabella mediant…		
Frigoring		19/05/2009 12.28.49
ArmRefr06	Attrezzature	11/06/2008 11.42.24
TRST06	Attrezzature	06/07/2007 9.23.33

CODICE	MODELLO	DESCR	DESCRUK
MA8959700	MTTKRI004	Kit ruote - 2 con freno - MTTK	Set 4 Wheels - 2 with bra
MA8963900	PRAMTS10	STN - Piano con bordo arrotor	Working top with round e
MA8964000	PRAMTS20	STN - Piano con bordo arrotor	Working top with round e
MA8964100	PRAMTS30	STN - Piano con bordo arrotor	Working top with round e
MA8964200	PRAMTS40	STN - Piano con bordo arrotor	Working top with round e
MA8967600	PRAMTSP10	STN/P - Piano con bordo arrot	Working top with round e
MA8967700	PRAMTSP20	STN/P - Piano con bordo arrot	Working top with round e

In the field “CODICE” must be specified for each accessory a code at most 15 characters long that must be unique in all the accessories table.
In the field “MODELLO”, it is possible to indicate the model code specified by the manufacturer, which may be displayed in the offer, but is not used directly by Kcad applications.
In fields “DESCR”, “DESCRUK”, “DESCRF”, “DESCRD” and “DESCRE” can be found a description in Italian, English, French, German and Spanish languages, which must be long at most 75 characters.
In the field “PREZZO”, the list price of the producer, in the currency specified in the “VALUTA” field of the table "ACCESSOR" in KCAD.MDB database.
Other fields can be added and used by custom applications for the automatic compilation of offers and related documents, but they are not necessary for Kcad applications.

ADDITION OF A NEW OBJECT IN A LIST

If you need to create a whole new list, you must create a new Access database, giving it the short name of the list (used in the “TABELLA” field in “LISTINI” table of KCAD.MDB database), and create inside it the table of the divisions, which must have the same short name of the list, and include the fields described in "GENERAL FURNITURE LIST DATABASE STRUCTURE”. You must add at least one line corresponding to a division, and create the table of objects contained in the division, and add at least one line with the data of an object, as explained below. Also must be added the line for the new list in “LISTINI” table of KCAD.MDB database, specifying the short name and the description of the list, the currency used for prices and any discount and recharge percentages.

To create a new object should be create an AutoCAD 2d block, a slide and optionally a 3d block which must be copied in the subfolder of the division to which the object belongs. These files can be shared by more than one object, if they look the same but have different technical characteristics. Finally, you must add a line with all object data in the table of the division to which the object belongs.

We must follow a nomenclature that allows the program to distinguish between 2d blocks, 3d blocks and symbols: The convention taken is that the last two characters of the names of the blocks have a special meaning. The second last character "1" indicates that the lock should not be converted from 2d to 3d and vice versa, the "2" indicates a block 2d which has a corresponding block 3d, like most of the objects KCAD, and character " 3 "indicates a block 3d which has a corresponding block 2d. The last character must be "R" (to indicate a real object) if the block is a physical object, such as ordinary objects of KCAD, or "S" (to indicate a symbol), if the block represents an indication in the design, as the symbols of the plugs, the marking symbols and the column headers of the summary table. By convention, the names of the slide must be equal to those of the corresponding block 2d and 3d, at least the second last character, which must be "0", and the extension “.SLD”.

In blocks of real objects, a unit of AutoCAD corresponds to a meter in the scale of the drawing. In symbolic blocks, a unit of AutoCAD corresponds to a centimeter in the actual printed design.

Although it is not essential, the convention used in the design of 2d block is that the insertion point is at the angle at the bottom left (to the point 0,0,0), and the hand touch the wall is horizontal at bottom side (along the x axis). The design must all lie on the floor level (z=0).

3d blocks must have the insertion point corresponding to that of 2d block, always at floor level (z=0). Must match both the scale and orientation of the two blocks. The direction of the height (from floor to top) should be along the positive semiaxis of z. For the design of 3D blocks, there are no particular limitations, but in order not to make the design too heavy to load, can be used entities “LINE” and “SOLID” with THICKNESS set equal to wanted height, or “POLYFACE” and “3DFACE” entities.

The blocks like the consoles, which have variable elevation from the ground, must be drawn at z=0 level, and the elevation above ground should be specified with a particular coding explained in the "SPECIAL PLUGS", later in this chapter. In this way, AutoCAD may place the 3d to block the request elevation, and this may be specified from time to time, using the data MODIFICATIONS command. If a block has a fixed elevation from the ground, you can not use this encoding and drawing it directly to correct elevation from the earth, holding the insertion point at z=0 level.

We recommend that you get the slides for viewing in the selection of objects from a 3d block image in isometric view, after the removal of hidden lines, but without fill areas with colors (set the AutoCAD system variable FILLMODE = 0). In this way the displaying of the slides in the selection window will be quickly and the images clearer. The overall impression must be close to what usually appears in the paper lists to represent the object, in order to find it easily. We recommend that you use the commands "_VPOINT 1,1,1" to choose the point of view, "_PAN" to center the picture in the monitor and finally the "_HIDE" to generate the image to be saved with "_MSLIDE".

To add the object to a list, you must start ACCESS, load the database of the furniture list and open the table corresponding to the division to which the object belongs. The last row of the table is always empty to allow the insertion of new rows. Specify its code in "CODICE" field, making sure that it is unique in the whole list (ACCESS can check that it is unique in the division, but can't check that it is unique in the whole list), the model in the "MODEL" field (if the productor uses a different model coding convention model than the code specified in "CODICE"), the short description and long description in all languages used.

The price must be specified as the original supplier list price, from which will be subtracted the percentage discount specified to obtain the effective purchasing cost, to which is added the rate of recherge to get the official selling price. These percentuals are those given in the table "LISTINI" of KCAD.MDB or specified for each single division in the table of the divisions within the database list.

Provide the name of the slide to view in the selection of objects, and optionally also the name of the DWG, where the same slide may reflect the different blocks to insert. The slide name must end with "0R", without extension ".SLD, and the drawing name must end with "2R", without extension ".DWG". If the name of the DWG field is not specified, is derived from the name of the slide, which must always be specified.

If the object provides accessories, in the "ACCESSORI" field must be specified the name of the list of accessories to be used by default, which can be positioned in a different database, but must always be listed in the table "ACCESSOR" of KCAD.MDB. In this way, when the positioning of the object in the design, is presented the window for the selection of accessories from the list specified.

The dimensions must be specified using the format "<dimension x>x<dimension y>x<dimension z>" or "<dimension x>x<dimension y> h.<dimension z>", specifying the measures in millimeters. For example, for a table 120x90 cm and 85 cm high, you can specify "1200x900x850" or "1200x900 h.850".

The fields "ATTACCHI" and "PLUGS" contain the same information on plugs and consumption, expressed in notation suitable for the Italian and English language, and which are described in detail in "PLUGS CODIFICATION", later in this chapter .

Other fields, such as "WEIGHT" or "TECHNICALPAPER" or "OVERALLDIM" may be added as needed. These additional fields do not affect the operation of KCAD applications, but can be used by custom macros for compiling the offer and other documents.

If you need to create a new table for accessories, you must create a new table in an Access database that already exists, or create a blank database in a folder by the same name as the database. The table name is the short name of the list of accessories, and the format should be the one described above in "ACCESSORIES TABLES". After adding at least one object, specifying the values of the fields needed, you must update the "ACCESSOR" table in KCAD.MDB, specifying the name of the list of accessories in the "TABELLA" field and the name of the database that contains it in the "DATABASE" field. You must also specify the currency in which the list prices are expressed, and any percentage of discount and recharge.

PLUGS CODIFICATION

The plugs have a complex encoding because it must be very flexible, and indicate not only the presence and location of the various types of plugs, but also the possible use of gas or electricity. Moreover, the field must be replicated as "ATTACCHI" for the Italian language, and "PLUGS" for English, due to some text notes that are displayed in the drawing.

The plugs field of an object is made up of multiple lines, one for each plug, whose general syntax is as follows:

<Plug>[number],<Symbol>=<Value>[,<Symbol>=<Value>…]

Consider this example:
"AF, h=cm7 from Floor, %%c3/4",XY=335,92"

AF is the abbreviation the plug, which in this case is a plug of cold water. The serial number is optional and is not used here, but serves to distinguish possibly two plugs of the same type. If it is absent or equal to 1, is the first plug of that type, otherwise it must be given as 2, 3 and so on.

From the plug name is derived the name of block to be inserted as a symbol. To build it, the application remove any serial number, then enter the character underscore "_" to reach a total of six characters, and finally adds the suffix "1S" or "1R" and the ".DWG" extension to get the block name.

In this example, in the "SIMBOLI" subfolder of KCAD installation folder must be present a block of 2d by name AF____1S.DWG, representing the symbol for the plug of cold water, drawn in the desired size expressed in centimeters referred to the printed drawing. You can also use plug blocks dimensioned in real size, as is explained later in this chapter.

Follows some pairs *<Simbolo>=<value>*, separated by a comma. Some of these pairs have a special meaning, because indicate a consumption of energy and must be considered when calculating consumption totals, others are simple indication to be displayed along with the characteristics of the plug in the project. For example, "h = cm7 from Floor " indicates the plug elevation from the floor, in centimeters, and is shown in the plug label on the drawing, but has no other relevance. So also the diameter indication "%%c=3/4"" uses the code "%%c" to display in AutoCAD the character symbolizing the diameter "ø" and indicates its value in inches, but has no other relevance.

This type of indication is completely free and can be specified other pairs "*<Symbol>=<value>*" separated by commas, without any change to the program. Since this information may contain words such as "h=cm7 from Floor", which will appear in the labels generated in the drawing, you must use the two fields "ATTACCHI" and "PLUGS", to contain the same data, but using Italian words in the first field and English words in the second.

In the sample line is shown also the relative position of the insertion point of the block. The syntax used is "XY = xxx, yyy" where xxx and yyy indicates the plug coordinates with regard to the insertion point of the block not rotated. If you need to specify decimal values for the coordinates, use the point as decimal separator and not a comma. In this case, the presence of the comma between the x and y coordinates is necessary and is an exception to the general rule for which the value corresponding to a symbol must not contain any commas. The program will rotate and scale, if necessary, the insertion point to adjust it to the orientation and scale of the object block.

The most important pairs symbol / value are those that specify the consumption of gas, electricity and compressed air, because they can be recognized and its values can be added to verify the total consumption of all the objects in the project. These consumptions should be all provided in the total consumption summary table, so the encoding of consumption of individual plugs must follow a well-defined conventions.

GAS CONSUMPTION – To show the gas consumption, the plug name must be GAS and must be specified the consumption in kilowatt gas with the syntax “KWG=*nnn*”. When Kcad calculates the total gas consumption, looks for all the plugs of GAS kind and adds all the values associated to the symbol KWG.

230v ELECTRICITY CONSUMPTION – In this case it is not important the plug name but the presence, among the plug characteristics, of an indication “V=AC230”. So KCAD looks in the same line another indication in the format “KW=*nnn*”, to sum it to the total of the used kilowatt of 230v electric energy.

400v ELECTRICITY CONSUMPTION – As the 230v type, the name of the plug is not important but the presence of an indication “V=3N-AC400” and another one in the same line, in the format “KW=*nnn*”.

SOFTENING HOT WATER CONSUMPTION– The plug name must be ACADD and must contain an indication as “LH=*nnn*” of consumed liters per hour .

SOFTENING COLD WATER CONSUMPTION– The plug name must be AFADD and must contain an indication as “LH=*nnn*” of consumed liters per hour .

STEAM CONSUMPTION – The plug name must be VAP and must contain an indication as “KGH=*nnn*“ of consumed kilos per hour.

COMPRESSED-AIR CONSUMPTION – It is not important the name of the plug, it is sufficient that among its characteristics, there is an indication “CPA=*nnn*”, showing the quantity of used compressed-air in cubic meters per hour.

INLET AIR – The name of the plug must be AI, and the name of the corresponding block must be AI____1R.DWG, with final “R” to allow the plug real dimensioning, as explained below. Must be present an indication such as “MCH=*nnn*”, showing the cubic meter of inlet air per hour.

OUTLET AIR – The name of the plug must be AO, and the name of the corresponding block must be AO____1R.DWG, with final "R" to allow the plug real dimensioning, as explained below. Must be present an indication such as "MCH=*nnn*", showing the cubic meter of oulet air per hour.

You can specify the insertion point of the plug as an attribute of the blockof the object. At the time of the creation of the object block, add an attribute with the flag I (invisible) and P (default) set. The tag of attribute should be "PLUG", the prompt does not need to be set, the default value should be the name the plug to which the attribute refers, including the possible sequence number. In the example above, would be "AF". The position in the attribute block defines the location of the plug.

Note: If the object is exploded its attributes are deleted and cannot be used to define the location of the plugs. In this case, their position can be defined as given in the description line or is assigned automatically by the program. There is a check that if an object has an attribute for the insertion point of an plug but not has a line in the description of the object plugs data, at the time of the analysis carried out by the command PLUGS it is generated a warning message indicating the name of the object and the plug, but did not stop the command.

If is not present neither the attribute nor the indication of the position in the row describing the plug, the program shall assign an arbitrary insertion point, trying not to overlap the various plugs. In any case, you can move the symbols of the plugs, everywhere they have been placed, with the usual AutoCAD commands, without affecting the proper functioning of the program.

The TABLES command automatically creates the summary table of the objects in the drawing. If the plugs have been already displayed with the PLUGS command, the summary table will contain also the total of consumptions. Because maybe that after having run the PLUGS command there was the insertion of other objects that results with no displayed plugs, if some plug is found, the command TABLE recall automatically the PLUGS command, to ensure that all the plugs are present. Then it examines again all the drawing to create the tables with all the needed consumption columns.

This ensures that no plugs is forgotten, but implies that it is not possible to delete an plug symbol to make it ignored by the command. In this case, you may change the details of the object plugs using the MODIFICATION command, to remove the plug only for the selected object. If the deleted plug has an insertion point specified with the attribute method, the command will generate a warning indicating that it does not find the plug corresponding to the attribute but in this case this advice may be ignored

PLUGS WITH REAL SIZE

In the most common case, the plug blocks are just symbols, whose dimension shall follow that of the texts and other symbols, so that the printed size are constant. In the case of special plugs, such as those of outlet hoods, it is desirable that the blocks have variable size, related to the scale of other objects, and not to those of symbols, and shall be changed easily.

In this case, the block names corresponding to the plugs must have "R" as the last character instead of "S" as the symbol blocks. In this way, KCAD recognize the block as a plug with real size, and scale it consistently with the other drawing objects.

In addition to this feature, you can change the dimensions X and Y of the plug, when inserted in the drawing, specifying between its plug data row the two pairs symbol / value "DX = xxx" and "DY = yyy" to indicate the dimensions in meters, remembering to use the point as decimal separator. In this way, you can only use one block for all the rectangular plugs of any size. When the plug block is inserted, the X scale and Y scale are set so that the sides are having the desired size. If one measure is specified, the other is left as it was drawn in the original plug block. This feature is based on an analysis of the block that expects to find lines or polylines along the outer perimeter of the block. If you use different entities for the perimeter of the results could be incorrect.

This type of plug is rotated with the object to which it is connected, so that they are correctly oriented, while symbolic plugs are moved in order to meet the insertion point on the object, but their angle is always oriented in the same way.

OBJECTS WITH VARIABLE FLOOR ELEVATION

There are items such as shelves, whose elevation from ground must be specified from time to time. This elevation, even if it is irrelevant in the process of composition of the 2d drawing, becomes important when it needs to be generated a three-dimensional view with the command KCAD3D.

To solve the problem easily, was introduced to the convention to specify the elevation from ground in a line of the PLUGS field, using the syntax HZ = zzz, where zzz indicates the elevation in meters, using the dot as decimal separator. The floor is assumed to be at z = 0. This information must be specified in a single line of the PLUGS field, without any other data from some plug.

In this way, the object is always positioned at level z=0 in the 2D drawing, but it will be automatically moved to the indicated elevation at the time of transformation in 3d. If you later run the reverse command KCAD2D to return to two-dimensional drawing, the insertion point of the block will be moved again at z = 0.

PLUGS LABELLING

The LABEL command allows you to indicate on the drawing all the data concerning the selected plug. The command recognizes the selected plug, the object that is connected, and research data between the lines of PLUGS filed. Once found, it simply produces the label reporting the various indications, in the found format. The only data not reported data are the position data “XY = *xxx*, *yyy*”, and any size data “DX = *xxx*”, “DY=*yyy*”.

The command try to position the tip of the arrow on the bound of the plug. This is possible if the plug outer perimeter consists of a circle with the center coinciding with the insertion point to the plug, or if it consists of a not rotated rectangle made with lines or polylines. Otherwise the arrow tip is placed at the insertion point of the plug block.

BALLOONS

Even the numbered block that forms the balloon is customizable; it is sufficient to show its name in the configuration modifiable through the KCAD command.

The block must be constructed on a “MARK” level, have the flag P (Predefined) set, the label (TAG) set to "NMRK", the prompt and the default set to nothing (answer with ENTER to the relative questions). Consider that a unit of AutoCAD in the block design is a centimeter in the final plotted drawing.

The block design must be kept in the subdirectory SYMBOLS of KCAD.

The default position of the ballons is set with the command KCAD, where you have to specify a X distance and a Y distance from the insertion point of the object (supposing that the object is inserted without being turned).

MARKINGS

Also the numbered mark block is customizable, just indicating its name in the configuration edited using the command KCAD.

The block must be drawn on the layer "MARCHE", and must have an attribute with the flags P (Predefined) set, and the tag set to “NMRK”, while the prompt and the default should be set to an empty string (press ENTER to respond to these requests). Consider that an AutoCAD unit in the drawing of the block corresponds to one centimeter in the printed design. The file of the block must be kept in the subfolder “SIMBOLI” of the installation KCAD folder.

The mark default location is set by using the command KCAD, where it can be specified the x and y distance from the insertion point of the object, imagining that the object is inserted without being rotated.

BLOCKS OF HEADING OF TABLES

For the production of summary tables in the drawing, it’s possible to use the column headers made up of simple text entity, or create blocks with column header more elaborate. Among the configuration parameters set with the KCAD command from within AutoCAD, the option "Use the blocks in the table header” must be set to 1.

In this case, to represent the column header of summary tables are used the blocks searched in the “SIMBOLI” subfolder, which may be customized.

The block with all the fixed column headers – for the fields indicating the position, quantity, model, description and size - is called "HDRSTD". Those for the columns of consumption of gas, electricity, water, steam, drain and air inlet are called respectively " HDRGAS "," HDRELC "," HDRHYD "," HDRSTM "," HDRDRN "," HDRAIR ". These names will be added the suffix "IT" or "FR" if the table should be produced in Italian or French language.

All these blocks can be changed with normal AutoCAD commands to get the column header with the desired appearance.

APPENDIX – TECHNICAL DOCUMENTS

This chapter contains technical information for the programmers who want to have direct access to the information coded by KCAD in the KCR files and who want to write their own macro in Visual Basic Application edition using the functions of the KCAD “KCAD32.DLL” library. Knowing the information below described is not necessary for the correct working and for the system maintenance, but it allows you to write programs able to exploit the KCAD data and functions. These informations are subject to change in future releases of KCAD.

KCAD32.DLL FUNCTION LIBRARY

The filling in of the offer and of the other related documents is made through some macros in Visual Basic Application edition that recall some compiled routine contained in the KCAD32.DLL library.

The DLL has been designed to interface with Visual Basic Application edition, that is it can be used with Excel 95 and inserted later in Word 97. In general, the DLL can be used by all the applications that implement a 32 bit version of Visual Basic Application edition.

There are over 50 functions that allow you to have access to the different KCR file that form an offer, to their objects and all their data. Here below are listed in the order they are usually recalled. Beyond, each function will be described in detail.

getRptSelection	Select a report or a report group via dialog box.
selNextRpt	Set the first or the next report as current, and return the name.
getRptDescription	Return the description of the current report.
setRptDescription	Set the description of the current report.
getRptOffNum	Return the offer number of the current report.
setRptOffNum	Set the offer number of the current report.
getRptOffDate	Return the offer date of the current report.
setRptOffDate	Set the offer date of the current report.
getRptOffAgent	Return the agent’s name of the current report.
setRptOffAgent	Set the agent’s name of the current report.
getRptClientCode	Return the customer’s code of the current report.
setRptClientCode	Set the customer’s code of the current report.
getRptClientName	Return the customer’s name of the current report.
setRptClientName	Set the customer’s name of the current report.
selNextTab	Set the first or the next table as current, and return the name.
selNextObj	Set the first or the next object as current, and return the code.
getObjCde	Return the code of the current object.
getObjLst	Return the first or the next accessory as current and return the code.
getObjDiv	Return the division of the current object.
getObjSld	Return the slide of the current object.
getObjNPr	Return the progressive number of the current object.
getObjQty	Return the quantity of the current object.
getObjMod	Return the model of the current object.
getObjList	Return the list description of the current object.
getObjSDs	Return the short description of the current object.
getObjLDsRow	Return the first or the next line of long description of the current object.
getObjDim	Return the dimension of the current object.
getObjDBPrc	Return the supplier’s original price for the current object.
getObjDBDsc	Return the supplier’s discount percentage to the firm for the current object.
getObjCostPrc	Return the effective price to the firm of the current object.
getObjDBRch	Return the recharge percentage of the firm on the real price of the current object.
getObjSalePrc	Return the official firm price for the current object.

getObjClientDsc	Return the discount price to the customer on the official firm price for the current object.
getObjClientPrc	Return the effective price to the customer of the current object.
getObjDBFld	Return the generic field value for the current object.
getObjDBMemoFld	Select or return the first or the next line of a generic memo field from the database of the current object.
selNextAcc	Set the first or the next accessory of the current object as current, and return the code.
getAccCde	Return the current accessory code.
getaccLsa	Return the accessories list of the current accessory.
getAccUnit	Return the unit of measure of the current accessory.
getAccQty	Return the quantity of the current accessory.
getAccList	Return the list description of the current accessory.
getAccDes	Return the description of the current accessory.
isAccInglobed	Return the embedding state of the current accessory.
getAccDBFld	Return a generic field value for the current accessory.
getAccDBPrc	Return the original supplier price for the current accessory.
getAccDBDsc	Return the supplier discount percentage to the firm for the current accessory.
getAccCostPrc	Return the effective cost to the firm of the current accessory.
getAccDBRch	Return the firm recharge percentage on the effective cost of the current accessory.
getAccSalePrc	Return the firm official price for the current accessory.
getAccClientDsc	Return the discount percentage to the customer on the official firm price for the current accessory.
getAccClientPrc	Return the effective price to the customer of the current accessory.
getAccDBFld	Return a generic field from the database of the current accessory.
selNextLst	Return the first or the next list name used in alphabetical order.
getMDBFld	Return whichever field from whichever database.
aboutKcad	Display a dialog box with information about the current DLL version.

FUNCTIONS DECLARATION

Before using one of these functions, it is necessary to declare the function with the instruction DECLARE. For example:

```
Declare Function getRptSelection Lib "KCAD32.DLL" () As Integer

Declare Function selNextRpt Lib "KCAD32.DLL" (first As Integer,
buffer As String) As Integer

Declare Function getObjLDsRow Lib "KCAD32.DLL" (first As
Integer, buffer As String, Language As Integer) As Integer

Declare Function getAccDBFld Lib "KCAD32.DLL" (field As String,
buffer As String) As Integer
```

The syntax below described with every function is the original one used in “C” and the one to use in Visual Basic Application: to obtain the corresponding declaration in Visual Basic Application, keep in mind that all the functions give back a whole and have as arguments only strings or integrals: their declarations will follow the model of one of the declaration mentioned above.

As the functions return an integral (-1 if ended with success and 0 if ended with failure), they have always to be used as right part of an assignment to a integer variable, for example:

```
'Starting demand of selection of one or more report files
nRep = getRptSelection
If (nRep = 0) Then
  Exit Sub
End If
```

Thanks to the DLL integration with the Visual Basic Application edition, it is not necessary to pay attention to the strings lenght used as buffer to obtain the return value, nor it is necessary to initialize them before the call with null values. In the same way, the strings returned by the functions are not strings in “C” style (that end with the letter ASCII 0), but they are normal strings in Basic format, so it is not necessary to clear them before being used again.

FUNCTIONS REFERENCE AND GENERAL GUIDE LINES

The functions are designed in order to allow the sequential reading of the data contained in every selected report file.

The first necessary operation is the demand of the selection of one or many report files (with KCR extension), from the disk through the function getRptSelection. The function displays a dialog box that allows the multiple selection of many files. For comfort, it is suggested to keep the KCR files to select together in the same directory.

With the function selNextRpt, you obtain the name of the first selected file, if the function is recalled with the argument “first” different from zero, or the name of the next file after the last previously returned, if it is recalled with the argument “first” equal to zero. In this way you can have, in order, all the names of the selected files. Reached the last name, the next call returns zero and an empty string as name of the file.

In addition to the name of the selected KCR file, the function selNextRpt sets it as “current report”.

Once you set the current report in this way, you can recall the functions that return and set the report data linked to the offer, such as the offer number, the customer’s name and other data of this kind, that do not depend on the objects in the design.

As these data can be known only at the moment of the filling of the offer, they are the only ones that can be set; typically, if a group of reports are selected together to compose only one offer, they will have these data one like the others. You can use these setting functions to rewrite the data (if modified by the operator) both in the first report and in all the next ones.

The DLL library places at your disposal no othe function to modify the design content, such as the tables, the objects and the accessories: these datas can be seen only with the available functions.

The function selNextTab, recalled with the argument “first” different from zero, sets the first table contained in the current report file, and returns its name; if the argument “first” is zero, it sets the table next to the last previously returned. The return value is zero if you have reached the end of the report. Setting a table as current allows to obtain the contained objects, using the function selNextObj.

The function selNextObj recalled with the argument “first” different from zero sets the first object contained in the current table as current object, and returns its code; if the argument “first” is zero, it sets as current the object next to the the last previously returned.

When an object is current, you can obtain its data using the different functions “getObj…”; among these, the only one that needs attention is getObjLDsRow, that returns the single lines that form the long object description: also this one has the argument “first” that, if set to a value different from zero, allows the function returns in the buffer the first line of the current object long description; otherwise, the line next to the last one will be returned. The return value of the function is zero if all the lines have been returned.

For more flexibility, it has been added the “low level” function getObjDBFld, that allows to obtain the datas of an object specifying the field name in the database. It is suggested to use the already existing functions for the expected field, and to use this function only for the database fields added in the personalization of the program.

To allow also long descriptions in languages not expected, it has been added even the function getObjDBMemoFld that allows to obtain, one at a time, the lines of an object memo field specifying its name. It is suggested to use the functions already existing for the expected fields and to use this function only for the memo fields added in the personalization of the program.

To read the accessories data of the current object, you have to use the function selNextAcc that, recalled with the argument “first” different from zero, allows to set the first accessory of the current object as current accessory, and recalled afterwards with the argument “first” is zero, it sets as current the accessory next to the last previously set.

Once set the current accessory, you can ask for its data with the functions “getAcc…”. As for the objects, for possible fields added in the personalization it is available the “low level” function getAccDBFld that allows to obtain the accessory datas specifying the name of the field in the database. It is suggested to use the functions already existing for the expected fields and to use this function only for the database fields added in the personalization of the program.

Sometimes, it is necessary to have a view of data for used lists: to this aim, it is available the function selNextList that, in the same way, returns in alphabetical order the first of the list among all the used lists inside the current report, if recalled with the argument “first” different from zero, and the next one in alphabetical order, if recalled with “first” is zero. As side effect, this function sets again the first report table as current table, so you can browse it searching for object of the selected list.

To access to the information of other tables (as the table of the lists or the list divisions), it has been added the function getMDBFld, that demands as parameter the name of a database, the name of a table, the name of a field to use as key, the value of the key field to look for, the name of the field to read and a buffer where there will be written the value found in the demanded field of the first record that has in the key field the given key value.

This function must be used only to read that information that do not refer to the objects. For example, if in the table LISTS in KCAD.MDB a column IDGROUP with an identification of the belonging group has been added, if you want to read the field value for the list FRIGORING you could use the function:

```
tmp = getMDBFld("KCAD.MDB", "LISTS", "TABLE", "FRIGORING",
"IDGROUP", buffer$)
idgroup = Val(buffer$)
```

getRptSelection

Syntax: short FAR PASCAL **getRptSelection** (void)
VBA Declaration: Declare Function **getRptSelection** Lib "KCAD32.DLL" () As Integer

This function displays the file selection dialog box of Windows that allows the multiple selection of KCR files.

The user can select one or many KCR files pressing the CONTROL key. The function returns the number of selected files or zero if you have selected the Cancel button.

The names of the next files can be obtained one at a time, using the function setNextRpt that set them as current reports.

Before ending, this function set the first report as current, however it don't returns neither the value returned by that function, nor the report name; for this reason you can recall again the function setNextRpt with subject first different from 0.

Returned Values:

0, no report file selected.

> 0 number of the selected files.

selNextRpt

Syntax: short FAR PASCAL **selNextRpt** (short *first, BSTR *lpbsBuffer)
VBA Declaration: Declare Function **selNextRpt** Lib "KCAD32.DLL" (first as integer, buffer as string) As Integer

This function has as argument an integer used as flag TRUE or FALSE, and a string passed as buffer to contain the name of the KCR file with the path.

It returns in the string passed as argument the name of the next report file of the multiple selection previously done with getRptSelection, with unit and complete path.

If the argument first is different from zero, the name of the first file of the selection is returned; otherwise, it is returned the name of the file next to the last one previously selected with this function.

The returned file is automatically set as “current report” so that the next operations of data reading will be done on it.

Returned values:

0, if no selection has been done with a call preceding the function getRptSelection, if there is no current report file and the function has been called with first is zero, if all the selected files are over or if the program has not been able to correctly open the report file (in this case its name is returned in the buffer passed as argument).

-1, if the wanted file has been selected (whose name has been copied in the buffer passed as argument).

getRptDescription

Syntax: short FAR PASCAL **getRptDescription** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getRptDescription** Lib "KCAD32.DLL" (buffer as string) As Integer

This function has as argument a string where it copies the description of the current report.

Returned values:

0, if a current report has not been set or if there is no description.

-1, if there is the description and it has been copied in the passed string.

setRptDescription

Syntax: short FAR PASCAL **setRptDescription** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **setRptDescription** Lib "KCAD32.DLL" (buffer as string) As Integer

This function has as argument a string containing the new description of the current report and replaces it to the previous one.

The modification is immediately written on disk.

The passed string can have maximum 63 letters and cannot be a null string otherwise there is no substitution.

Returned values:

0, if a current report is not set or if it cannot be modified.

-1, if there are no errors.

getRptOffNum

Syntax: short FAR PASCAL **getRptOffNum** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getRptOffNum** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the offer number of the current report.

Returned values:

0, if a current report has not been set or if there is no offer number.

-1, if there is the offer number and it has been copied in the passed string.

setRptOffNum

Syntax: short FAR PASCAL **setRptOffNum** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **setRptOffNum** Lib "KCAD32.DLL" (buffer as string) As Integer

This function accepts as argument a string containing the new offer number of the current report and replaces it to the previous one. The number can also have alphanumeric letters and symbols such as "-" or "/".

The modification is immediately written on disk.

The passed string can have maximum 63 letters and cannot be a null string, otherwise there is no substitution.

Returned values:

0, if a current report is not set or if it cannot be modified.

-1, if there are no errors.

getRptOffDate

Syntax: short FAR PASCAL **getRptOffDate** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getRptOffDate** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the offer date of the current report.

There is no check about the format or the validity of the date.

Returned values:

0, if a current report has not been set or if there is no offer date.

-1, if there is the offer date and it has been copied in the passed string.

setRptOffDate

Syntax: short FAR PASCAL **setRptOffDate** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **setRptOffDate** Lib "KCAD32.DLL" (buffer as string) As Integer

This function accepts as argument a string containing the new offer date of the current report and replaces it to the previous one.

There are no checks about the format or the validity of the date.

The modification is immediately written on disk.

The passed string can have maximum 63 letters and cannot be a null string otherwise there is no substitution.

Returned values:

0, if a current report is not set or if it cannot be modified.

-1, if there are no errors.

getRptOffAgent

Syntax: short FAR PASCAL **getRptOffAgent** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getRptOffAgent** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the offer agent's name of the current report.

Returned values:

0, if a current report has not been set or if there is no agent's name.

-1, if there is the agent's name and it has been copied in the passed string.

setRptOffAgent

Syntax: short FAR PASCAL **setRptOffAgent** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **setRptOffAgent** Lib "KCAD32.DLL" (buffer as string) As Integer

This function accepts as argument a string containing the new agent's current report and replaces it with the previous one.

The modification is immediately written on disk.

The passed string can have maximum 63 letters and cannot be a null string otherwise there is no substitution.

Returned values:

0, if a current report is not set or if it cannot be modified.

-1, if there are no errors.

getRptClientCode

Syntax: short FAR PASCAL **getRptClientCode** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getRptClientCode** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the customer’s offer code of the current report.

Returned values:

0, if a current report has not been set or if there is no customer’s code.

-1, if there is the customer’s code and it has been copied in the passed string.

setRptClientCode

Syntax: short FAR PASCAL **setRptClientCode** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **setRptClientCode** Lib "KCAD32.DLL" (buffer as string) As Integer

This function accepts as argument a string containing the customer’s new code of the current report and replaces it with the previous one.

The modification is immediately written on disk.

The passed string can have maximum 63 letters and cannot be a null string otherwise there is no substitution.

Returned values:

0, if the current report is not set or if it cannot be modified.

-1, if there are no errors.

getRptClientName

Syntax: short FAR PASCAL **getRptClientName** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getRptClientName** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the offer customer's name of the current report.

Returned values:

0, if a current report has not been set or if there is no customer's name.

-1, if there is the customer's name and it has been copied in the passed string.

setRptClientName

Syntax: short FAR PASCAL **setRptClientName** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **setRptClientName** Lib "KCAD32.DLL" (buffer as string) As Integer

This function accepts as argument a string containing the offer customer's new name of the current report and replaces it with the previous one.

The modification is immediately written on disk.

The passed string can have maximum 63 letters and cannot be a null string otherwise there is no substitution.

Returned values:

0, if a current report is not set or if it cannot be modified.

-1, if there are no errors.

selNextTab

Syntax: short FAR PASCAL **selNextTab** (SHORT *first, BSTR *lpbsBuffer)
VBA Declaration: Declare Function **selNextTab** Lib "KCAD32.DLL" (first as integer, buffer as string) As Integer

This function wants as arguments an integer used as TRUE or FALSE flag and a string.

It returns the next table of the current report as argument in the passed string and sets it as current table.

It can happen that a table has no name: in this case the returned string will be empty, but the value returned by the function will be in any way –1 (no error).

If the argument first is different from zero, the first report table is returned; otherwise the table next to the last previously selected with this function is returned.

The returned table is automatically set as “current table” so that the next operations of data reading will be done on it.

Returned values:

0, if the current report has not been set or if all the report tables are over.

-1, if there are no errors.

selNextObj

Syntax: short FAR PASCAL **selNextObj** (SHORT *first, BSTR *lpbsBuffer)
VBA Declaration: Declare Function **selNextObj** Lib "KCAD32.DLL" (first as integer, buffer as string) As Integer

This function wants as argument an integer used as TRUE and FALSE flag and a string.

It returns the code of the next object in the current table as argument in the passed string and sets it as current object.

If the argument first is different from zero, the first object of the table is returned; otherwise the next object to the last one previously selected with this function is returned.

The returned object is automatically set as “current object” so that the next operations of data reading will be done in it.

Returned values:

0, if the current table has not been set or if the table objects are over.

-1, if there are no errors.

getObjCde

Syntax: short FAR PASCAL **getObjCde** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjCde** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the code of the current object.

The returned code is the same returned by the function selNextObj.

Returned values:

0, if the current object has not been set.

-1, if there are no errors.

getObjLst

Syntax: short FAR PASCAL **getObjLst** (BSTR *lpbsBuffer)
VBA declaration: Declare Function **getObjLst** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the list name of the current object.

The name of the list of an object corresponds to the MDB file name that contains it and to the name of the directory that contains the MDB file.

Returned values:

0, if the current object has not been set.

-1, if there are no errors.

getObjDiv

Syntax: short FAR PASCAL **getObjDiv** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjDiv** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the division name of the current object.

The name of the division of an object corresponds to the name of the table that contains it in the list MDB file and to the name of the subdirectory that contains the object DWG abd SLD file, contained in the list directory.

Returned values:

0, if the current object has not been set.

-1, if there are no errors.

getObjSld

Syntax: short FAR PASCAL **getObjSld** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjSld** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the name of the slide of the current object.

The name of a slide of an object corresponds to the DWG and SLD files name that contain the block and the slide used by AutoCAD and that can be shared by many objects with different code. The last two letters of the name are 0R for the slide, 2R for the 2d block and 3R for the 3d block.

Returned values:

0, if the current object has not been set.

-1, if there are no errors.

getObjNPr

Syntax: short FAR PASCAL **getObjNPr** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjNPr** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies, as string, the progressive number of the current object.

The progressive number of an object can to be a non-integer number. This function returns it as string but if you want to memorize it in a numerical variable, it is necessary to use a floating point number with singular precision.

Returned values:

0, if the current object has not been set.

-1, if there are no errors.

getObjQty

Syntax: short FAR PASCAL **getObjQty** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjQty** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies, as string, the quantity of the current object.

Returned values:

0, if the current object has not been set.

-1, if there are no errors.

getObjMod

Syntax: short FAR PASCAL **getObjMod** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjMod** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the current object model, reading it from the database of the object list (or from its modifications, if it has ben modified).

Returned values:

0, if the current object is not set or if its database has not been found.

-1, if there are no errors.

getObjList

Syntax: short FAR PASCAL **getObjList** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjList** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the description of the current object list, reading it from the lists database.

Returned values:

0, if the current object is not set or if the lists database has not been found.

-1, if there are no errors.

getObjSDs

Syntax: short FAR PASCAL **getObjSDs** (BSTR *lpbsBuffer, SHORT *language)
VBA Declaration: Declare Function **getObjSDs** Lib "KCAD32.DLL" (buffer as string, language as integer) As Integer

This function wants as argument a string where it copies the short description of the current object, reading it from the database of the object list (or from its modifications, if it has been modified).

The integer number language specifies in which language you want the description. If language = 0, the returned description is in English; if language = 1, the description is in Italian; if language = 2, the description is in French. If other languages are implemented, other numbers can be at your disposal.

Returned values:

0, if the current object is not set or if its database has not been found.

-1, if there are no errors.

getObjLDsRow

Syntax: short FAR PASCAL **getObjLDsRow** (SHORT *first, BSTR *lpbsBuffer, SHORT *language)

VBA Declaration: Declare Function **getObjLDsRow** Lib "KCAD32.DLL" (first as integer, buffer as string, language as integer) As Integer

This function wants as argument a string where it copies a line of the long description of the current object, reading it from the database of the object list (or from its modifications, if it has been modofied).

The integer number language specifies in which language you want the description. If language = 0, the description is in English; if language = 1, the description is in Italian; if language = 2, the description is in French.

If the argument first is different from zero, the first line of the description is returned; otherwise, it is returned the next line to the last previously returned with this function.

The language argument is considered only if first is different from zero, otherwise it is in any way used the one set in the last call with first different from zero.

If the current object is changed from the last call, the first line of the description of the new object is returned anyway.

The description can contain empty lines: in this case, the returned string will be an empty string, but the value returned by the function will be –1.

Returned values:

0, if the current object is not set or if its database has not been found.

-1, if there are no errors.

getObjDim

Syntax: short FAR PASCAL **getObjDim** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjDim** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the current object dimensions, reading them from the object database (or from its modifications, if they have been modified).

Returned values:

0, if the current object is not set or if its database has not been found.

-1, if there are no errors.

getObjDBPrc

Syntax: short FAR PASCAL **getObjDBPrc** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjDBPrc** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the original price of the current object, reading it from the database of the object list (or from its modifications, if it has been modified).

Returned values:

0, if the current object is not set or if its database has not been found.

-1, if there are no errors.

getObjDBDsc

Syntax: short FAR PASCAL **getObjDBDsc** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjDBDsc** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the discount percentage that the supplier grants to the firm for the current object on the original price returned by the previuos function.

The discount percentage is read in the LISTS table of the KCAD.MDB database, but each division can have a different discount percentage specified in the divisions table of the specified list database that, if present, replaces, for the specified division, the discount shown in the KCAD.MDB database.

Returned values:

0, if the current object is not set or if the lists database or the one of its division has not been found.

-1, if there are no errors.

getObjCostPrc

Syntax: short FAR PASCAL **getObjCostPrc** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjCostPrc** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the firm effective cost of the current object, that is the supplier original price minus the discount percentage that the firm has on the object, returned by the previous function.

Returned values:

0, if the current object is not set or if its database, or the lists database, or the one of its division has not been found.

-1, if there are no errors.

getObjDBRch

Syntax: short FAR PASCAL **getObjDBRch** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjDBRch** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the recharge percentage that the firm applies to the current object on the effective price returned by the previous function.

The recharge percentage is read in the LISTS table of the KCAD.MDB database, but each division can have a different discount percentage specified in the divisions table of the specified list database that, if present, replaces, for the specified division, the recharge shown in the KCAD.MDB database.

Returned values:

0, if the current object is not set or if the list database or the one of its divisions has not been found.

-1, if there are no errors.

getObjSalePrc

Syntax: short FAR PASCAL **getObjSalePrc** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjSalePrc** Lib "KCAD32.DLLbuffer as string) As Integer

This function wants as argument a string where it copies the firm sale official price for the current object, that is the effective cost price plus the the recharge percentage returned by the previous function.

Returned values:

0, if the current object is not set or if its database, or the lists database, or the one of its divisions has not been found.

-1, if there are no errors.

getObjClientDsc

Syntax: short FAR PASCAL **getObjClientDsc** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjClientDsc** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the discount percentage that the firm applies to the customer for the current object according to the official sale price returned by the previous function.

The discount percentage is calculated according to the official price and the customer's final price specified for each object with KCADWIN, the modifications program of the report files under Windows.

Returned values:

0, if the current object is not set or if its database, or the lists database, or the one of its division has not been found.

-1, if there are no errors.

getObjClientPrc

Syntax: short FAR PASCAL **getObjClientPrc** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjClientPrc** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the final price to the customer of the current object, that is the price specified for each object with the KCADWIN modification program of the report files under Windows.

If the final price to the customer has not been specified, the firm official price is returned.

Returned values:

0, if the current object is not set or if its database, or the lists database, or the one of its divisions has not been found.

-1, if there are no errors.

getObjDBFld

Syntax: short FAR PASCAL **getObjDBFld** (BSTR *lpbsField, BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjDBFld** Lib "KCAD32.DLL" (field as string, buffer as string) As Integer

This function wants as argument a string containing the name of the field from where you want to obtain the value and a string to contain the value of the demanded field.

If the specified field exist in the object database, its value (or the value of its modification, if it has been modified) is copied in the passed string as buffer; otherwise, an empty string is returned and 0 is returned by the function.

It can happen that a field exists but has no content: in this case the returned string will be empty, but the function will return –1 (no error).

Returned values:

0, if the current object is not set or if its database or the shown field has not been found.

-1, if there are no errors.

getObjDBMemoFld

Syntax: short FAR PASCAL getObjDBMemoFld (const BSTR *fldName, BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getObjDBMemoFld** Lib "KCAD32.DLL" (field as string, buffer as string) As Integer

This function wants as argument a string containing the memo field name from where you want to obtai the value, and a string to contain a line of the demanded memo field.

The first time the function is called for the current object, it is necessary to specify the name of the memo field that you want to read. If the specified memo field exists in the object database, the first line is copied in the string passed as buffer; otherwise, an empty string is returned or 0 is returned by the function.

To read the next lines, it is necessary to recall again the function specifying an empty string as name of the memo field. In this case, the last set field is used, and the next line of the memo field is returned.

If the current object is changed from the last call, and the name of the field is not specified, it returns error.

The description can contain empty lines: in this case, the returned line will be an empty string, but the value returned by the function will be –1.

The function can be used with normal fields, returning the entire content at the first call, and in case 0 at the next calls, as it was a memo field with only one line.

Returned values:

0, if the current object is not set or if its database, or the shown field has not been found if the end of the memo field has been reached.

-1, if there are no errors.

selNextAcc

Syntax: short FAR PASCAL **selNextAcc** (SHORT *first, BSTR *lpbsBuffer)
VBA Declaration: Declare Function **selNextAcc** Lib "KCAD32.DLL" (first as integer, buffer as string) As Integer

This function wants as argument an integer used as TRUE or FALSE flag and a string.

It returns in the argument string the code of the next accessory of the current object and sets it as current accessory.

If the argument first is different from zero the first accessory of the object is returned; otherwise, it is returned the next object to the one previously selected with this function.

The returned accessory is automatically set as “current accessory” so that the next operations of data reading will be done on it.

Returned values:

0, if the current object is not set or if the object accessories are over.

-1, if there are no errors.

getAccCde

Syntax: short FAR PASCAL **getAccCde** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getAccCde** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the code of the current accessory.

The returned code is the same returned by the function selNextAcc.

Returned values:

0, if the current accessory is not set.

-1, if there are no errors.

getAccLsa

Syntax: short FAR PASCAL **getAccLsa** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getAccLsa** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the name of the list of the current accessory.

Returned values:

0, if the current accessory has not been set.

-1, if there are no errors.

getAccUnit

Syntax: short FAR PASCAL **getAccUnit** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getAccUnit** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the measure unit abbreviation of the current accessory.

Returned values:

0, if the current accessory has not been set.

-1, if there are no errors.

getAccQty

Syntax: short FAR PASCAL **getAccQty** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getAccQty** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the quantity of the current accessory as string.

Unlike the objects quantity, that is an integer number, the accessories quantity can be a floating point number. This function returns it as string but if you want to memorize it in a numerical variable, it is necessary to use a floating point number with singular precision.

Returned values:

0, if the current accessory has not been set.

-1, if there are no errors.

getAccList

Syntax: short FAR PASCAL **getAccList** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getAccList** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the list description of the current accessory, reading it from the database of the accessories lists.

Returned values:

0, if the current accessory has not been set or its list database has not been found.

-1, if there are no errors.

getAccDes

Syntax: short FAR PASCAL **getAccDes** (BSTR *lpbsBuffer, SHORT *language)
VBA Declaration: Declare Function **getAccDes** Lib "KCAD32.DLL" (buffer as string, language as integer) As Integer

This function wants as argument a string where it copies the description of the current accessory reading it from its list.

The integer number language specifies in which language you want the description. If language = 0, the returned description is in English; if language = 1, the description is in Italian; if language = 2, the description is in French. If other languages are implemented, other numbers are available.

Returned values:

0, if the current accessory has not been set or its database has not been found.

-1, if there are no errors.

isAccInglobed

Syntax: short FAR PASCAL **isAccInglobed** ()
VBA Declaration: Declare Function **isAccInglobed** Lib "KCAD32.DLL" () As Integer

This function returns –1 if the current accessory is embedded, that is if it has to be considered in the offer as a characteristic of the current object, and not as an accessory with description and price distinguished from those of the object.

Returned values:

0, if the current accessory is not set or not embedded.

-1, if the current accessory is embedded.
.

getAccDBFld

Syntax: short FAR PASCAL **getAccDBFld** (BSTR *lpbsField, BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getAccDBFld** Lib "KCAD32.DLL" (field as string, buffer as string) As Integer

This function wants as argument a string containing the name of the field from where you want to obtain the value, and a string to contain the value of the demanded field.

If the specified field exists in the accessory database, its value is copied in the string passed as buffer; otherwise, an empty string is returned and 0 is returned by the function.

It can happen that a field exists but has no content: in this case, the returned string will be empty, but the function will return –1 (no error).

Returned values:

0, if the current accessory is not set or if its database or the shown field has not been found.

-1, if there are no errors.

getAccDBPrc

Syntax: short FAR PASCAL **getAccDBPrc** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getAccDBPrc** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the original price of the current accessory, reading it from the database of the accessory list.

Returned values:

0, if the current accessory has not been set or its database has not been found.

-1, if there are no errors.

getAccDBDsc

Syntax: short FAR PASCAL **getAccDBDsc** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getAccDBDsc** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the discount percentage that the supplier grants to the firm for the current accessory on the original price returned by the previous function.

The discount percentage is read in the table ACCESSOR of the accessories database ACCESSOR.MDB in the subdirectory ACCESSOR of KCAD data.

Returned values:

0, if the current accessory has not been set or the accessories lists database has not been found.

-1, if there are no errors.

getAccCostPrc

Syntax: short FAR PASCAL **getAccCostPrc** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getAccCostPrc** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it is copied the effective cost to the firm of the current accessory, that is the supplier original price minus the discount percentage that the firm has on the accessory, returned by the previous function.

Returned values:

0, if the current accessory has not been set or if its database or that of the accessories list has not been found.

-1, if there are no errors.

getAccDBRch

Syntax: short FAR PASCAL **getAccDBRch** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getAccDBRch** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the recharge percentage that the firm applies to the current accessory on the effective cost returned by the previous function.

The recharge percentage is read in the table ACCESSOR of the accessories database ACCESSOR.MDB in the subdirectory ACCESSOR of KCAD data.

Returned values:

0, if the current accessory has not been set or if the database of the accessories lists has not been found.

-1, if there are no errors.

getAccSalePrc

Syntax: short FAR PASCAL **getAccSalePrc** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getAccSalePrc** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the firm official price for the current accessory, that is the effective cost plus the recharge percentage returned by the previous function.

Returned values:

0, if the current accessory has not been set or if its database or the one of the accessories lists has not been found.

-1, if there are no errors.

getAccClientDsc

Syntax: short FAR PASCAL **getAccClientDsc** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getAccClientDsc** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the discount percentage that the firm applies to the customer for the current accessory according to the official price returned by the previous function.

The discount percentage is calculated according to the official price and to the final price to the customer specified for each accessory with KCADWIN, the modification program of the report file under Windows.

Returned values:

0, if the current accessory has not been set or if its database or the one of the accessories lists has not been found.

-1, if there are no errors.

getAccClientPrc

Syntax: short FAR PASCAL **getAccClientPrc** (BSTR *lpbsBuffer)
VBA Declaration: Declare Function **getAccClientPrc** Lib "KCAD32.DLL" (buffer as string) As Integer

This function wants as argument a string where it copies the final price to the customer of the current accessory, that is the price specified for every object with KCADWIN, the modification program of the report files under Windows.

If the final price to the customer is not specified, the firm official price is returned.

Returned values:

0, if the current accessory has not been set or if its database or the one of the accessories lists has not been found.

-1, if there are no errors.

selNextList

Syntax: short FAR PASCAL **selNextList** (SHORT *first, BSTR *lpbsBuffer)
VBA Declaration: Declare Function **selNextList** Lib "KCAD32.DLL" (first as integer, buffer as string) As Integer

This function wants as argument an integer number used as TRUE or FALSE flag and a string.

If the argument first is different from zero, the description of the first list in alphabetical order among those used in the current report (of objects or accessories) is returned; otherwise, it is returned the one next, in alphabetical order, to the last list description previously returned with this function.

This function is necessary to form the documents where the objects of every single report are ordered according to the belonging list.

After having used this function, it is necessary to recall again the various functions selNextTab, selNextObj and selNextAcc to have access to the elements of the current report.

Returned values:

0, if the current report is not set or if the alphabetical lists panel is over.

-1, if there are no errors.

getMDBFld

Syntax: short FAR PASCAL getMDBFld (const BSTR *mdb_name, const BSTR *tbl_name, const BSTR *key_name, const BSTR *key_value, const BSTR *fld_name, BSTR *lpbsBuffer)
VBA Description: Declare Function **getMDBFld** Lib "KCAD32.DLL" (mdbname as string, tblname as string, keyname as string, keyvalue as string, fieldname as string, buffer as string) As Integer

This function wants as argument the name of a MDB database to refer, complete of path, the name of a table inside the specified database, the name of a key field present in the specified table, the key value that you need to find in the key field to identify the wanted record, the name of a second field where you want to read the value in the identified record, and a string to contain the returned value.

If the field exists but the content of the field is empty, an empty string is returned but the function returns –1.

Returned values:

0, if one of the input parameters is not valid.

-1, if there are no errors.

aboutKcad

Syntax: short FAR PASCAL **aboutKcad** (void)
VBA Declaration: Declare Function **aboutKcad** Lib "KCAD32.DLL" () As Integer

This function displays a dialog box with information about the DLL version used and with the address, the telephone and fax numbers, the web site and the email address of the society producing the software.

Returned values:

The function always returns –1.

SOMMARIO