



# TpaWorks

2.1.0

## *Working Editor*

---



Tecnologie e Prodotti per l'Automazione

This documentation is property of TPA S.p.A. Any unauthorized duplication is forbidden. The Company reserves the right to modify the content of the document at any time.

---

# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Check of the user licence and operating modes</b>	<b>3</b>
<b>3</b>	<b>Let's start with TpaWorks</b>	<b>5</b>
<b>4</b>	<b>Information about TpaWorks...</b>	<b>7</b>
<b>5</b>	<b>How to choose/select the plant</b>	<b>9</b>
<b>6</b>	<b>How to create and open a working database file of the current plant</b>	<b>11</b>
<b>7</b>	<b>How to create and open a working database file outside current plant</b>	<b>15</b>
<b>8</b>	<b>Printing the database</b>	<b>17</b>
<b>9</b>	<b>Save and close the database</b>	<b>19</b>
<b>10</b>	<b>Commands referred to nodes</b>	<b>21</b>
<b>11</b>	<b>Command referred to the workings</b>	<b>23</b>
<b>11.1</b>	<b>Insert a working</b>	<b>23</b>
	Automatic procedure of setting fixed Cycle	25
	Automatic setting procedure of a global function	25
<b>11.2</b>	<b>Enter a working with copy</b>	<b>26</b>
<b>11.3</b>	<b>Deselect all</b>	<b>26</b>
<b>11.4</b>	<b>Delete a working</b>	<b>26</b>
<b>11.5</b>	<b>Copy and paste a working</b>	<b>26</b>
<b>11.6</b>	<b>Find</b>	<b>27</b>
<b>11.7</b>	<b>Replace</b>	<b>28</b>
<b>12</b>	<b>The current working</b>	<b>29</b>

<b>13</b>	<b>The attributes of the working</b>	<b>31</b>
13.1	Operation code	32
13.2	ASCII name	32
13.3	Description	32
13.4	Status	34
13.5	Typology	34
13.6	Sub-typology	35
13.7	Technological typology	36
13.8	r Variables	36
13.9	Apply in matrix	36
13.10	Insert options	37
	Insertion faces	37
	Exclude from sequences	40
	Apply sequences in expanded list	40
	Exclude the snap in expanded list	40
	Excluded fields	40
	More	41
13.11	Apply in insertion	43
	Field L, Field O,....	43
	More	44
13.12	Limit in transforms	44
	Excluded transforms	45
	Transforms not propagated	46
	More	47
	Insertion modes	48
	More	48
13.13	Representation modes	48
	Parameter groups	49
	Graphic representation	51
<b>14</b>	<b>Commands concerning current working</b>	<b>53</b>
<b>15</b>	<b>Working parameters</b>	<b>55</b>
15.1	ID	55
15.2	ASCII Name	55
15.3	Status	56
15.4	Format	56
15.5	Minimum value and maximum value	57
15.6	Default \$ values	57
15.7	Dimension	57
15.8	Transforms	57
	Geometrical transforms	58
	Application criteria of assignments in case of translation	59
	Application criteria of assignments in case of mirror	60
	Application criteria of assignments in case of rotation	60
	Application criteria of assignments in case of stretch	60

Application criteria of assignment in case of inversion	60
Complete example	60
Parameter technology	61
Depth parameter	62
More	62
<b>15.9 Description</b>	<b>64</b>
<b>15.10 Column in matrix</b>	<b>64</b>
<b>15.11 Auxiliaries</b>	<b>64</b>
Graphic representation	65
Control typology	67
Enabling level	69
Assignments for List control	69
Non-operational functions	70
Enhanced features	71
Function of list messages swap	71
Active status conditions	72
Active status values	74
More	75
Cumulative selection	75
<b>16 Configurations in TpaWorks</b>	<b>77</b>
<b>16.1 Commands referring to working parameters</b>	<b>77</b>
<b>17 The working database</b>	<b>79</b>
<b>17.1 Workings in custom codes intervals</b>	<b>79</b>
Point and Setup codes	79
The application point	79
The technology	81
Oriented tool	81
Additional information in setup	83
Custom parameters	85
Custom logical codes	85
Direct logical conditioning	85
<b>17.2 Profile codes</b>	<b>86</b>
<b>17.3 Logical instructions</b>	<b>86</b>
<b>17.4 Global functions</b>	<b>86</b>
Function identification	87
Assignment of function call arguments	87
Assignment of function return elements	87
<b>17.5 Complex codes</b>	<b>88</b>
Direct logical conditioning	88
How the subroutine and the macro are identified	89
How <r> variables of subroutine and macro are assigned	89
Subroutine placement mode	90
Point hook	91
How a face to be applied is chosen	92
Selection of induced faces	93
Placement mode of induced calls	93
Application of geometric transforms	93
Inversion	93
Rotation	93
Mirror	94
Scale	94

Repetitions in subroutine running	95
Repetitions with free distribution	95
Repetition with matrix distribution	96
Repetitions with distribution for choice	96
Activation of emptying	97
Text generation	100
Generation of text and emptying at the same time	101
Generation of spline curves	101
Transforms application codes with programmed workings	102
Application of ISO curves	103
Why si an ISO file applied?	104
How a macro-program is defined	104
How the ISO file is identified	105
How it is shown to load the ISO file	106
ISO file placement mode	106
How is the ISO file interpreted	107

# 1 Introduction

TpaWorks is the program for the management of TpaCAD workings database. The workings make the elements by means of which a piece-program is made up.

A right knowledge of workings is required to use TpaWorks, that is a knowledge of both base assignments (definition of a working, its functioning, parameters,...) and customized assignments (for example: organization of palettes workings in TpaCAD).

TpaWorks allows to open, modify and record a database file of the workings only if the access level is checked **Constructor**.

For the references to the TpaCAD application program read the handbook of Configuration of TpaCAD.

Together with the setup of TpaCAD a base database of the general purpose workings is provided. The base database base is made starting from a macro-programs selection that are also provided together with the base setup.

The original base database and its macro-programs assure the rightness of the use of the workings, on the basis of what is recorded. That is why it is recommended not to modify the settings which are in these files and to go on with the workings customization following the modes as follows.






## 2 Check of the user licence and operating modes

The functioning of TpaWorks is protected by a usb hardware key in use. The hardware key can be moved from a computer to another, allowing the operation on different installations, but not at the same time. In fact, the presence of the key is detected on each request of execution of specific commands. If the check of the key fails, a message advise the operator and TpaWorks keeps enabled the only commands allowing to close the application program.

### Language change

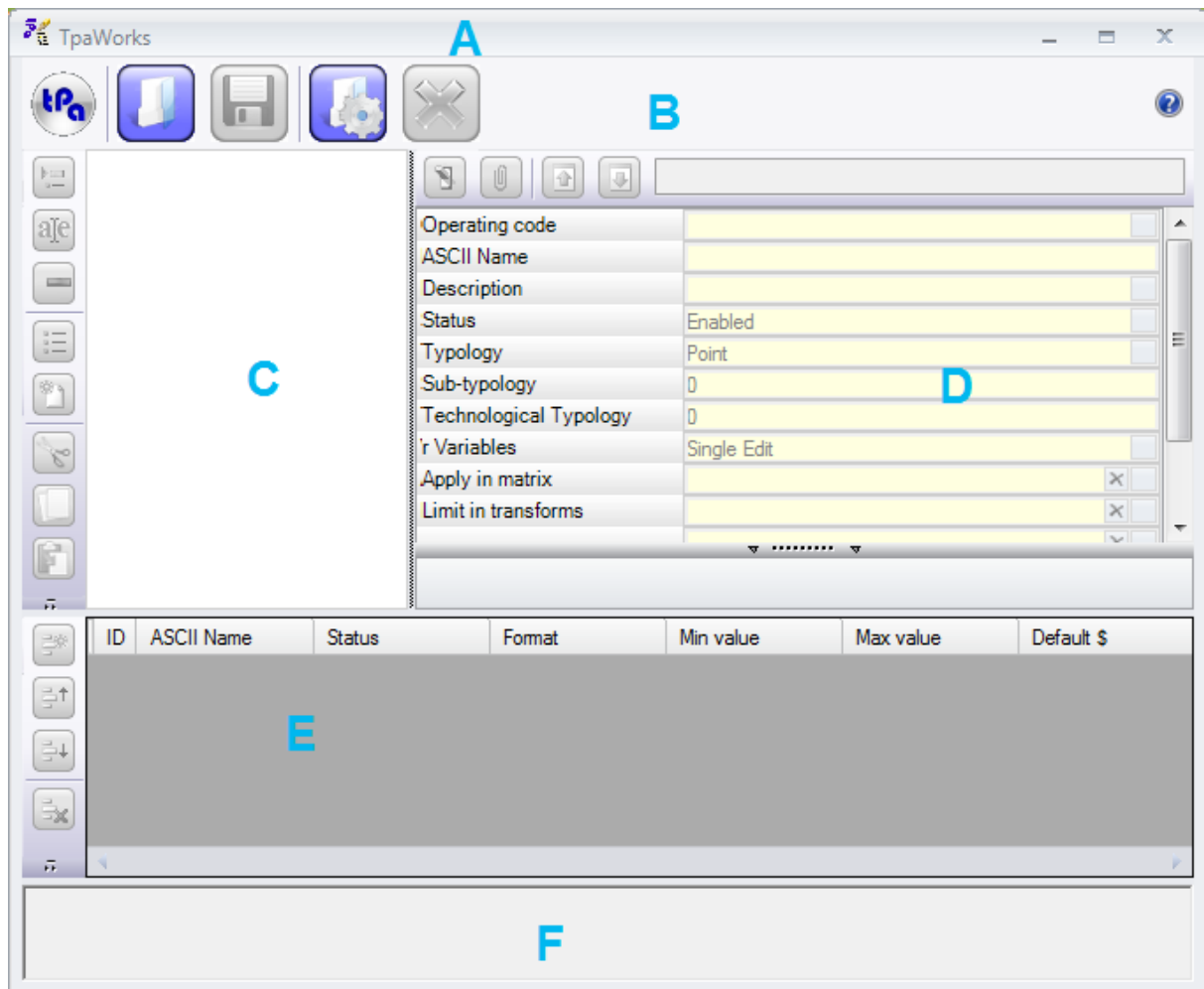
The language may be changed at any logon level. To change the selection of the language you should

use the key combination **[CTRL + /]** or click the icon  from Windows "**application tools**". In the opening window select the requested language and click the button **[OK]**. The language does not change immediately, but at the next startup of TpaCAD.



### 3 Let's start with TpaWorks

At the program startup no workings database files are automatically loaded. This is TpaWorks at its start:




The screen is arranged in areas identified with the following letters:

<b>A</b>	Window title
<b>B</b>	Menu bar
<b>C</b>	Area for workings list: it is arranged in a tree-path. A commands palette for management of workings list is in use, too
<b>D</b>	Area for assignment of attributes of current working
<b>E</b>	Area for assignment of parameters of current working
<b>F</b>	Area for visualisation of alarms/warnings and specific messages



## 4 Information about TpaWorks...


The command **Informations about TpaWorks** is selected from menu   
Display version number, TpaWorks program operating mode, legal information, about copyright and about licences, information about Tpa site.





## 5 How to choose/select the plant

A plant consists of one or more machines. The machines (or modules) consist of an assembly, divided into sub-assemblies groups and devices. Usually, the plant is single, thus no change is provided for. Sometimes, more configurations are to be installed for different plants.

The window of the plant choice can be recalled from menu ->**Plant**. This is an optional and to be configured command. It is active only when there is no database that is open. The window shows the list of the configured plants in alphabetical order, marking the current selection. Select the name of the plant you want to use and confirm by pressing the button **[Ok]**.

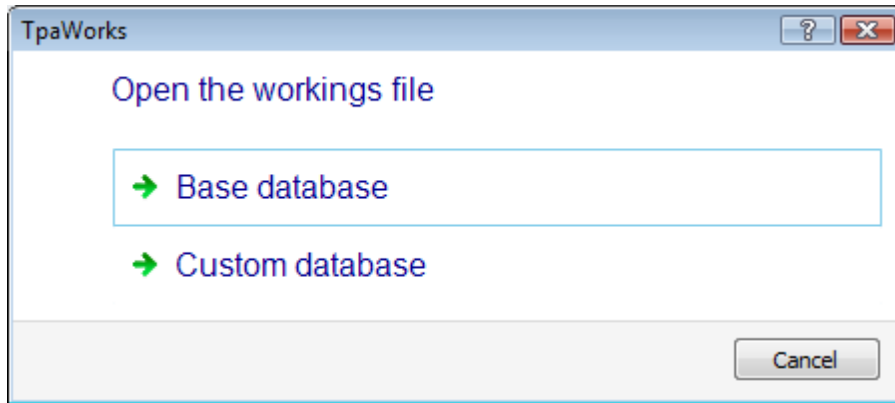




## 6 How to create and open a working database file of the current plant



To open or create a database select the button  and chose in the selection window:



To open the base database of the current plant, confirm the window on the first item/entry. To open the database of custom workings, confirm the window on the second item/entry. The base database is stored in the file DBWORKS.wcad in the folder tpacadcfg. The custom database is stored in the file DBWCUST.wcad in the folder tpacadcfg\custom. If the file to be opened doesn't exist, a new database is created.

The DBWORKS.wcad file is provided together with the TpaCAD base set up and makes available the general purpose workings.

The DBWCUST.wcad file contains additional or modified workings in respect of the base database, for specific needs of a particular application of the item.

The workings are arranged in two database that integrate base workings with customised workings for each custom or also for each application.

### **Normally, it is not possible to modify the base database.**

Adjustment/Change of base database is question for TPA internal development.

If it is necessary to change a base database working adding, for example, custom parameters:

- copy the working from the base database into/in the custom one
- modify the copied working.

Functioning with both loaded databases ensures that in case of doubled workings in the two databases the base working is deleted, so that the main choice is made in favour of customizations.

A new working has to be added in the custom database.

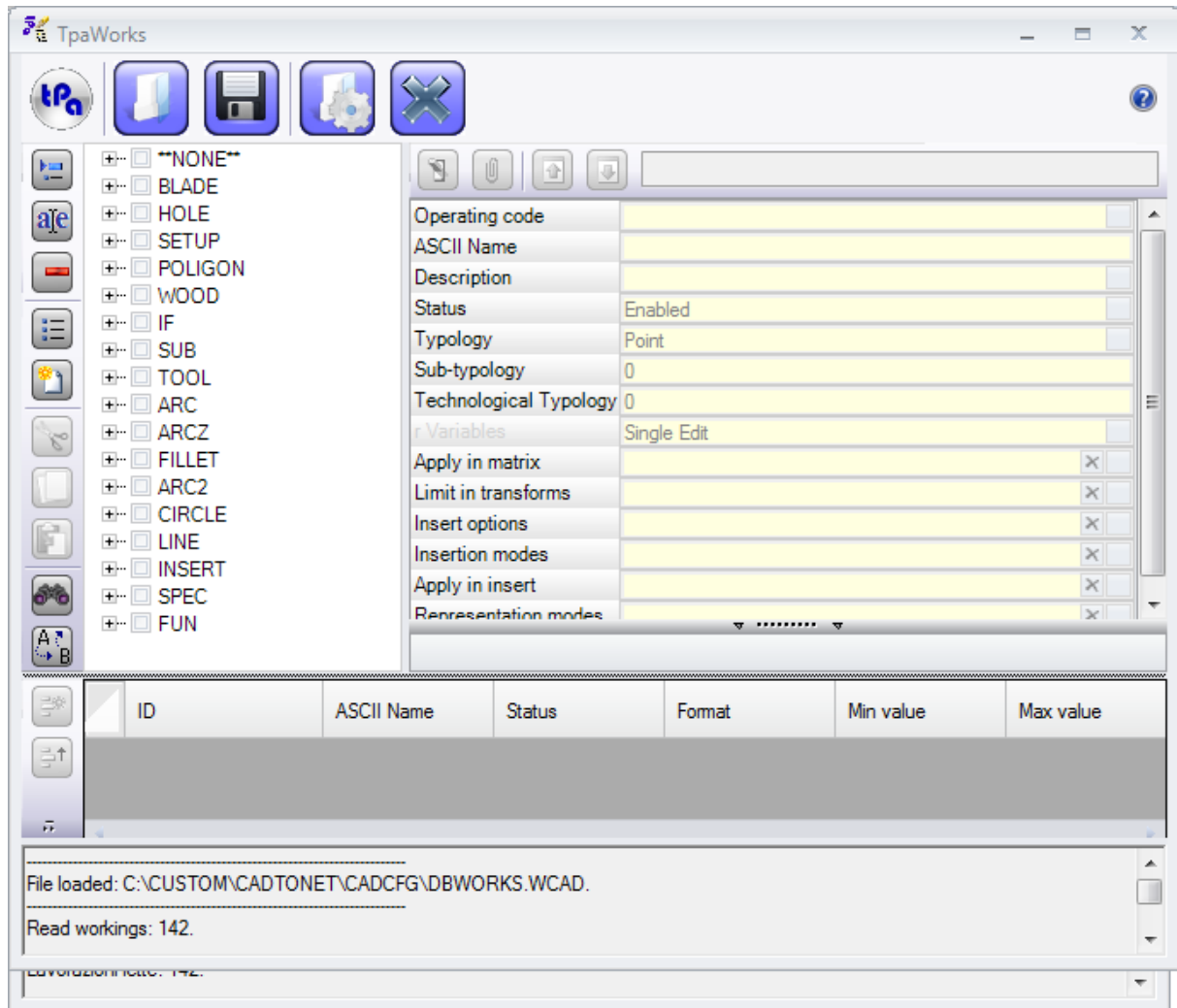
To exclude management of workings of base database it is recommended to operate on TpaCAD configuration.

**The wordings "it is recommended", "it is advised" that are used here and that we are going to read later on are to be meant as advice for a good programming and should be adopted as a rule.**

### **Also change of custom database can be blocked:**

The manufacturer can "sign" the custom database blocking its access in/during change. For "signature" operations see related unit:

This is the screen now:



The display area of particular messages provides information referred to loading result and the following are showed:

- deleted workings
- deleted parameters
- actually loaded workings

If no correction in automatic mode has been applied, the number of workings loaded from the database is showed in the display area.

The validity checks made on a workings database reading match the filters and checks that are normally active during the assignment itself of the database in TpaWorks. Changes during the reading can so result only from direct alteration cases, which are to be avoided.

With a loaded database the workings tree-path is composed by workings grouped in nodes:

"\*\*NONE\*\*", "HOLE", ...

Each main node match a button in the workings palette of TpaCAD (see TpaCAD Configuration Manual, paragraph about configuration of workings Palette).

\*\*NONE\*\* node groups \*\*NONE\*\* node groups workings that have no button assignment, that is they can not be selected in the workings palette. They are generally workings of particular use.

To select a working it is necessary to open a node and select an option in the list. A working is identified by:

- an operating code shown between square parenthesis Example: [81]
- a describing message, adjusted to the selected language Example: "HOLE"

In a node the order of workings matches the order they can be selected in the palette of TpaCAD workings.

The working selected in the tree-path is shown as current working: together with the data of current workings the assignment areas of the attributes of the working and of the parameters are compiled:

The screenshot displays the TpaWorks software interface. On the left, a tree view shows a hierarchy of manufacturing operations: \*\*NONE\*\*, BLADE, HOLE, and POLYGON. Under the HOLE category, several operations are listed, including [81] HOLE, [83] POLAR HOLE, [1008] MULTIPLE DRILLING, [1001] X-FITTING, [1002] REPEAT X, [1003] Y-FITTING, [1004] REPEAT Y, [1005] XY-FITTING, [1006] REPEAT XY, [1010] DRILLING ALONG TF, and [1011] DRILLINGS ON POLY.

The main panel on the right shows the detailed configuration for the selected operation, [81] HOLE. The configuration includes the following fields:

- Operating code: 81
- ASCII Name: HOLE
- Description: !1001HOLE
- Status: Enabled
- Typology: Point
- Sub-typology: 0
- Technological Typology: 1
- Variables: Single Edit
- Apply in matrix: (empty)
- Limit in transforms: (empty)
- Insert options: (empty)
- Insertion modes: (empty)
- Apply in insert: (empty)


At the bottom of the interface, a table lists the parameters for the selected operation:

ID	ASCII Name	Status	Format	Min value	Max value	Default \$	Dime
PRASSINC [8015]	EG	Edit	Integer	0.000	1.000		None
PRQX [1]	X	Edit	Double	0.000	0.000		[mm]
PRQY [2]	Y	Edit	Double	0.000	0.000		[mm]
PRQZ [3]	Z	Edit	Double	0.000	0.000		[mm]
PRF_TOOL [1002]	TD	Edit	Double	0.000	0.000		[mm]
PRN_MAC [201]	TMC	Edit	Integer	0.000	0.000		None
PRN_GRP [203]	TR	Edit	Integer	0.000	0.000		None
PRN_SUBGRP [204]	EM	Edit	Integer	0.000	0.000		None



## 7 How to create and open a working database file outside current plant



Select the command  that opens the window of files opening with only active typology(\*.wcad). Confirm opening of a workings database file: the same is loaded and made available in window.

This is a loading of external database, on which no modifications are possible.

For example, it is possible to recover one or more workings or parameters of one working in order to set/insert them in the database (base or custom) of the plant.

Concerning this:

- some windows of direct assignment (for example: description on working or parameter) aren't opened;
- assignment windows that are more developed are open, but only in display mode.



## 8 Printing the database



### **Print database structure**

This command prints the database, that corresponds to the tree structure displayed in the video.





## 9 Save and close the database



### **Save the current database (base or custom).**

At each saving also a copy (backup) of the previous version of the database is executed. Each new backup copy replaces the previous one.

The backup copy is featured by .bak extension and is saved in the same folder that contains the original file.

The command isn't enabled in case of opening of an external database.






### **Close the current database.**

If the base database has been modified, a confirmation window/dialog box asks if you want to save the file.



## 10 Commands referred to nodes

The commands for nodes management are in the vertical palette on the left of the workings tree-path. The commands are not enabled in case of external Database.

	<p><b>Insert a Node</b> inserts a new node in workings tree-path: It is necessary to show the name of the node:</p> <ul style="list-style-type: none"><li>• the name has got a length between 2 and 20 alphanumeric characters</li><li>• the first character must be alphabetic</li><li>• the name cannot be used (it must be of one meaning).</li></ul> <p>the name is always converted into capital:</p>
	<p><b>Rename the node</b> changes the name of the current node. The command is active only with selection of a grouping node and not for the <b>**NONE**</b>node. For the assignment of the name, above mentioned commands are valid.</p>
	<p><b>Delete the node</b> The command is active only with the selection of a grouping node <u>empty</u> and anyway not for <b>**NONE**</b> node.</p>



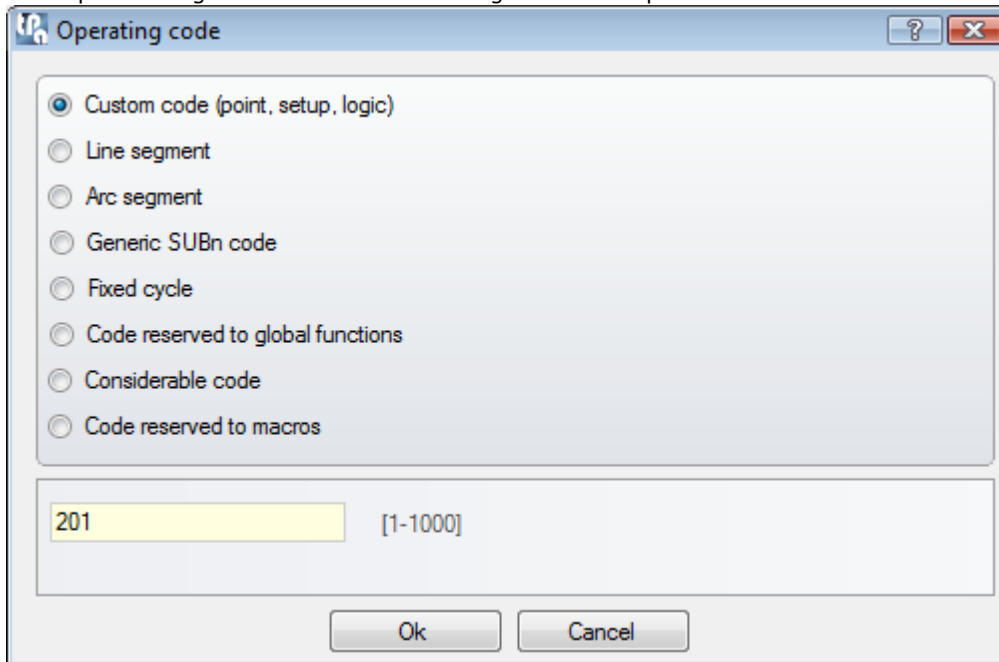
# 11 Command referred to the workings

## 11.1 Insert a working



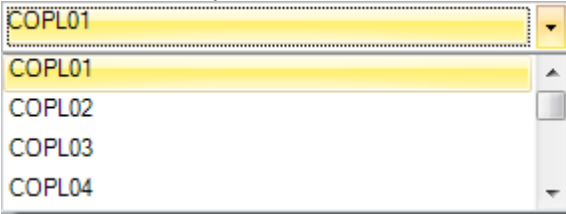
### Insert a working

Inserts a new working in the position following the current working. The command is not active if the maximum number of managed workings (1000) is already assigned or if you are working on an external Database. The window requires assignment of the new working numerical operative code:



The value must be univocal in the whole database. Chose the group of the working (items at the top) and then the operative code in the lower part of the window.

Let's see the groupings:

<p><b>Custom code</b></p>	<p>Custom workings of following typologies: point, set up, logical. The setting is direct (edit field) and can assign a value in the range: 1 -1000. <b>Workings of this group are already assigned in the base database: for new custom implementations it is recommended to set a value between 201 and 1000</b> (custom interval),so that there are no conflicts with workings that are already available in the base database. New workings of the base database will use a code free in interval 1-200). The setting field is shown with the first free value in interval. Select item <i>Code in custom interval</i>(available in opening of custom database): the first free code in custom interval (201-1000) is directly shown. Workings that are already assigned in the base database are for example:</p> <ul style="list-style-type: none"> <li>• single drilling codes</li> <li>• milling setups and blade setup</li> <li>• base insertion code.</li> </ul>
<p><b>Line segment</b></p>	<p>Profile base workings of linear typology: selection is made on a predefined list with indication of symbolic names:</p>  <p>Workings of this group are already assigned in the base database.</p>
<p><b>Arc segment</b></p>	<p>Profile base workings of arc typology: selection is selected on a predefined list</p>

	<p>with indication of symbolic names. Workings of this group are already assigned in the base database.</p>
<b>SUB generic code</b>	<p>Workings of complex typology assigned as generic calls to subroutine: selection is on the basis of a predefined list with indication of symbolic names. In this group there are:</p> <ul style="list-style-type: none"> <li>• codes of SUB type for subroutines recall</li> <li>• codes STOOL type</li> </ul> <p>Workings of this group are assigned in the base database.</p>
<b>Fixed cycle</b>	<p>Workings of complex typology assigned as fixed cycles. Setting is direct (edit field) and can assign a value in intervals: [1001–2000], [3001–4800].</p> <p><b>Workings of this group are already assigned in the base database: for new custom implementations it is recommended to set a value between [1501 - 1899] and [3001 - 4800]</b> (custom intervals), so that there are no conflicts with workings that are already in the base database. New workings in the base database will use a code that is free in the interval: 1001-1500).</p> <p>The setting field is shown with the first free value in interval. Selecting item <i>Code in custom interval</i> (available in custom database opening):</p> <p>the first code that is free in the custom interval is directly shown. Workings of fixed Cycle that are already assigned in the base database are for example:</p> <ul style="list-style-type: none"> <li>• drilling cycles: FITTING, REPEAT</li> <li>• milling cycles: ELLIPSE, RECTANGLE, POCKET</li> <li>• emptying cycles of basic geometries: ELLIPSE, RECTANGLE, POCKET</li> <li>• cycle of texts development</li> <li>• blades</li> <li>• insertions</li> <li>• ISO curves application</li> </ul>
<b>Code of global function</b>	<p>Logical workings that manage application of a custom function that can compile a returns array (see TpaCAD Configuration information). Setting is direct (edit field) and can assign a value in interval: 2701 –2800.</p> <p><b>Workings of this group can already be assigned in the base database: for new custom implementations it is recommended to set a value between 2751 and 2800</b> (custom interval), so that there are no conflicts with workings that are already in the base database. New workings of the base database will use a code that is free in the interval: 2701-2750).</p> <p>The setting field is shown with the first value free in the interval. Selecting item <i>Code in custom interval</i> (available in opening of CUSTOM DATABASE): the first free code is directly shown in the custom interval. Workings that are already assigned in the base database are for example:</p> <ul style="list-style-type: none"> <li>• solutions of geometries: POLAR SYSTEM, SYMMETRY/POINT ROTATION</li> <li>• logical solution: AND/OR MULTIPLES.</li> </ul>
<b>Considerable code</b>	<p>Workings of different typologies that have a considerable interpretation. In this group there are:</p> <ul style="list-style-type: none"> <li>• geometric codes of typologies: point, set up, line, arc</li> <li>• logical codes that can be used in program (IF, ELSEIF, ELSE, ENDF, EXIT, ERROR)</li> <li>• codes for assignment of variables &lt;j&gt;</li> <li>• codes for assignment of application points –initial and final- of subroutine</li> <li>• codes for creation of automatic face and programmed application of induced calls.</li> </ul> <p>Selection is on the basis of a predefined list with indication of symbolic names.</p> <p>Workings of this group are assigned in the base database and it is recommended not to modify them.</p>
<b>Codes reserved to macros</b>	<p>Logical workings reserved to macro programs: all codes have a considerable interpretation. In this group there are:</p> <ul style="list-style-type: none"> <li>• codes of a cycle programming: FOR, ENDFOR</li> <li>• check codes of program flow: BREAK, CONTINUE</li> <li>• codes for assignment of &lt;\$&gt; variables.</li> </ul> <p>Selection is on the basis of a predefined list with indication of symbolic names.</p> <p>Workings of this group are assigned in the base database and it is</p>

	recommended not to modify them.
--	---------------------------------

Closing of the window with confirmation can start an automatic procedure of working assignment. Let's see all possible cases:

### Automatic procedure of setting fixed Cycle

If a code of fixed Cycle is selected, it is possible to start the automatic setting for the application of a custom function:

The window Open File is opened: the folder shown is the one where the macro-programs are stored (cadcfg for the base database, tpacadcfg\custom for the custom database).

Chose a valid format file (macro-program or subroutine) and confirm the choice.

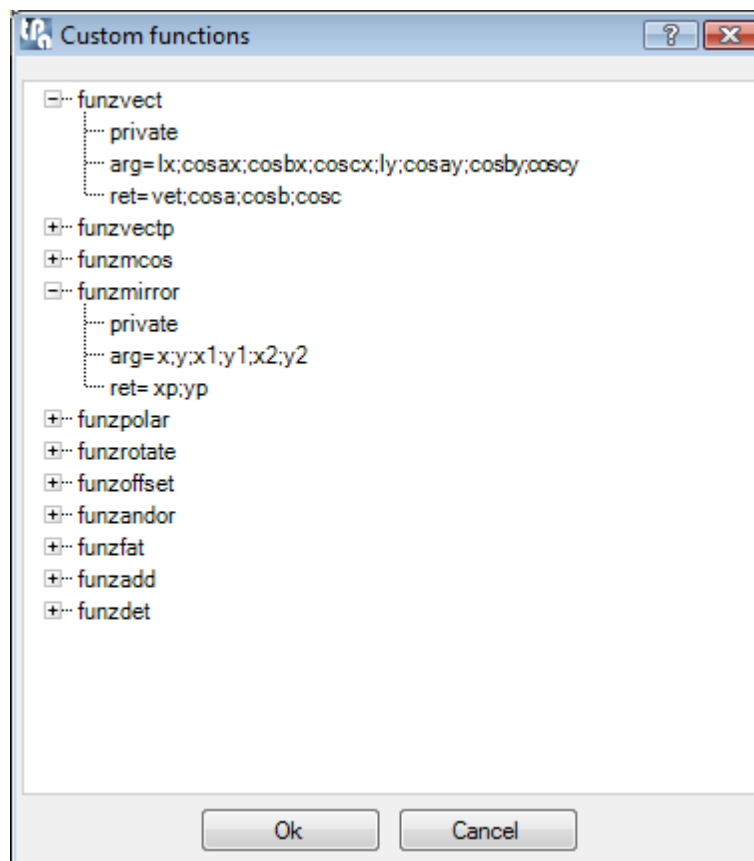
The list of parameters of the working is set with the parameters for the assignment of:

- essential parameters for the application of subroutine or macro-program (name of the subroutine, applied face)
- variabili <r> public variables of subroutine (parameters with identification number in range: 8500-8799).

Starting from the first settings it is then possible to change the working.

### Automatic setting procedure of a global function

If a Function code is selected, it is possible to start the automatic setting for the application of a custom function:



A window opens showing the list of the custom functions as they are read (by TpaCAD Configuration). Each function matches a node of a tree-path structure. Each node shows the name of the function (funzvect, funzvectp,..) and can be expanded:

- function state: public, private
- arguments ("arg=...")
- returns ("ret=...").

In the list following functions are shown:

- valid (written correctly)

- with or without declared returns.  
Chose the custom function that you want to use as a model and confirm with the button **[OK]** . The new working is set according to the declarations given for the function.  
Starting from the first settings it is then possible to change the working.

## 11.2 Enter a working with copy



enters a new working as a copy of the current working.  
The new working is inserted in the position following the current working.

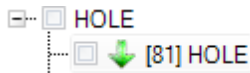
The command is not active if the maximum number of managed workings (1000) is already assigned or when you are operating on an external Database.

The window requires the assignment of the numeric operative code of the new working, as already seen for the command **Enter a working**, only now it isn't possible to change the owning group of the working: to continue with the command you need to set a new operative code and confirm the window.

## 11.3 Deselect all



**Deselect all** resets all selections on the workings tree-path.



Next to both each main node and each working a box is displayed:

- select/deselect a node box to select/deselect all workings of the node
- select/deselect the box of a working to select/deselect each single working.

Multiple selections allow to execute overall/comprehensive delay, copy or insertion operations.

## 11.4 Delete a working



**Delete workings**

This command deletes selected workings (with the active check box), otherwise it removes current working.

Deleted workings are copied in the TpaWorks local Clipboard.

In the area reserved to the signals, messages referring to each single executed operation are reported.  
The command is not active in case of external Database.

## 11.5 Copy and paste a working



**Copy workings**

This command copies selected workings (with the active check box), otherwise it removes current working.

The copy is executed in the TpaWorks local Clipboard.

In the area reserved to the signals, messages referring to each single executed operation are reported.



**Paste workings**

This command inserts available workings in the local Clipboard at the current insertion point on the workings tree-path, in the current node. It is possible move single workings with relative commands, to obtain the requested order in the of graphic selection palette.

The command isn't active if:

- workings in local Clipboard aren't available
- Maximum number of managed workings (1000) has already been assigned.
- in case of external Database

For each working to be inserted following checks are carried out:

- if a working has already been assigned with the same (operative code, ASCII name): it is possible to



chose to replace it;

- if a working has already been assigned with the same ASCII name and the operative code isn't used: it is possible to chose to replace it;
- if a working has already been assigned with the same operative code but with a different ASCII name, it is possible to enter the insertion.

In the area reserved to the signals, messages referring to each single executed operation are reported.

## 11.6 Find



### Find

Search for the first working which meets the assigned search criteria:

### Find

- **Operating code**: if the field isn't assigned, the search doesn't apply filter on the operating code (example: 81)
- **ASCII name**: if the field isn't assigned, the search doesn't apply filter on ASCII code (example: "HOLE").
- **Typology**: select box to apply search filter and select required typology in the list
- **Sub-Typology**: select box to apply search filter and set required value (whole number in interval: 0-100).
- **technological Typology** : select box to apply search filter and set required value (whole number in interval: 0-100).
- **Parameter**: select box to apply search filter and set ID of required parameter

### Search options

- **Search in all directions**: if enabled, the search is executed in the whole workings tree, otherwise only downstream of current working.
- **Applies to selected workings**:if enabled, the search is executed only among selected workings (items with active check box).

Button **[Find next]**allows to keep on searching:

- button selection is not available if none of the searching fields is set
- a message notifies the negative result of the search

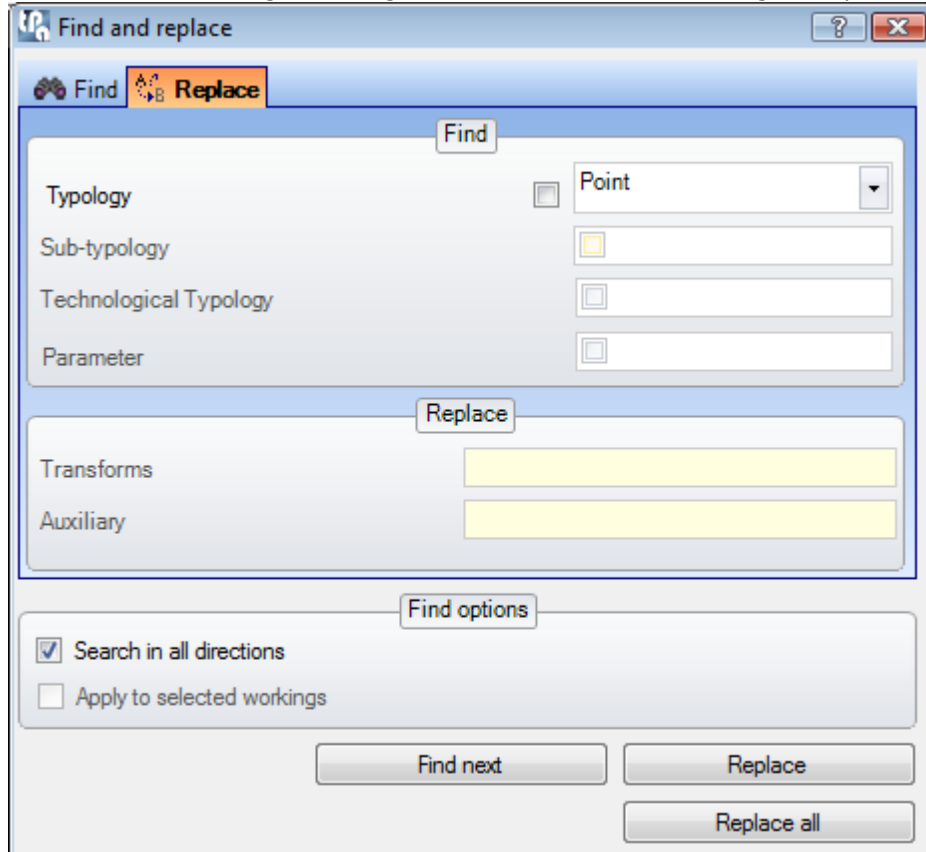
- otherwise: the working found becomes the current working.

## 11.7 Replace



### Replace

The command looks for the workings checking search criteria and makes assigned replacement:



### Find

- **Typology**: select the box to apply the search filter and select in the list the required typology
- **Sub-Typology**: select the box to apply the search filter and set the required value (whole number in interval: 0-100).
- **technological Typology** : select the box to apply the search filter and set the required value (whole number in interval: 0-100).
- **Parameter**: select the box to apply the search filter and set the ID of required parameter

### Replace

- **Transforms**: imposta i campi da assegnare per l'attributo Trasformate dei parametri della lavorazione
- **Auxiliary**: sets the fields to be assigned for the attribute Auxiliary of working parameters.

**[Find next]** allows to start or to keep on searching without any replacement. A message notifies the negative result of the search, otherwise the found working becomes the current one.

**[Replace]** applies the replacement required to the working that verify the set search criteria

**[Replace all]** replaces all workings that verify the search criteria with the new data set.

## 12 The current working

Current working is fully defined on two areas that show respectively attributes and working parameters. The assignment of a working has got two essential aims:

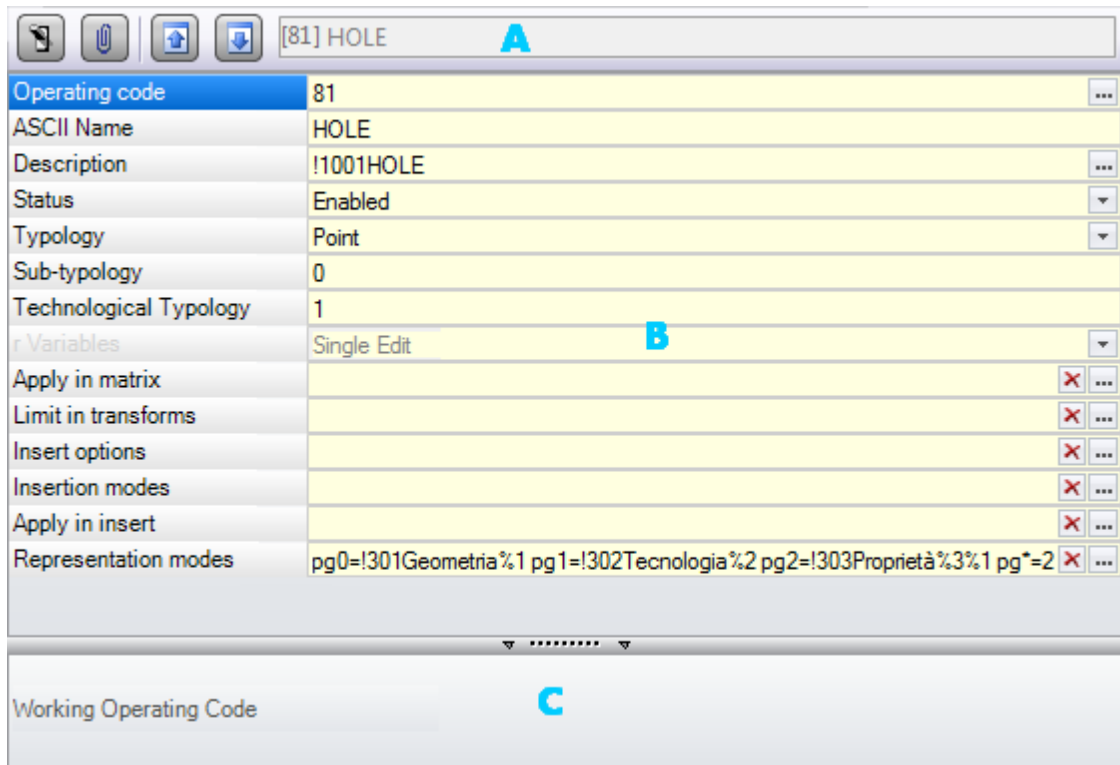
- to define use features for its operative function: for what is is useful, behaviour and features of a working when it is utilized in a program, its behaviour in application of tools, if and how it arrives in execution mode of the piece,...:
- define its direct interaction modes: how the check of data entry of the working is structured (parameters assignment, if it manages local preview, help on line,...).

In an external database no information of current working can be modified.



# 13 The attributes of the working

Attributes of a working are the set of assignments that define generically a working.



<b>A</b>	The area includes: <ul style="list-style-type: none"> <li>• a bar of commands that can be selected for current working.</li> <li>• the working title (operative code and describing message)</li> </ul>
<b>B</b>	List of attributes of current working
<b>C</b>	Display area of a describing message of current attribute.

Some of the list can be modified with edit by keyboard or selection in list (if the button of list opening is available: ). In the fields in which the button is available the parameters are set in dedicated windows.

Some fields have generic setting in string format (always in dedicated window), in which information can be stored with a meaning that can also not be defined previously. The field can assign information that is useful at different levels and not necessarily operating together with other information (program management and/or TpaCA and/or optimization application program).

This is a feature of some parameters attributes, too (see further). As proposed information is just of general use, the format of these attributes is defined in advance and rigidly.

1. The string assigns one or more entries, that are separated by the character ` ` (space). Each entry has got a (univocal) header that consists by "name item" followed by "=" . Example: "cop=";
2. the recognition of the "name item" is case-insensitive, that is without distinction between upper- and lower case letters. Examples of equivalent assignments: "cop=", "Cop=", "COP=";
3. it follows the assignment of the entry, in numerical form (whole or with comma, positive or negative) or of string.

each setting window of a field of this kind displays:

- entries of known meaning are distinct
- entries that aren't previously known (field: "Other") are grouped in a one only text field. Other:

For each entry of significant format it is shown how the entry is assigned to its concerning attributes (of working or of parameter): header and assignment and also particular evaluations and default conditions. Unknown entries must comply with above mentioned format and utilize not declared headers that have already been used.

## 13.1 Operation code

The operation code is a numerical value between 1 and 5000 that identifies the working in a univocal way.

All database workings are identified by two univocal pieces of information: the Operating Code and the ASCII Name (see next paragraph).

Change of value shows the window that has already been examined for the insertion of a new working. Change of operating code can lead to automatic assignment of other attributes of working: Typology, Sub-Typology, Subgroup typology, Geometric code, r Variables, Apply in matrix.

## 13.2 ASCII name

The ASCII Name is the name of working in ASCII representation that identifies the working uniquely.

Set a string that checks the format:

- length between 2 and 10 alphanumeric characters
- the first character must be alphabetic
- the name of two alphabetic characters beginning with letter 'W' are considered reserved names (and for this reason not allowed) (examples: "WB", "WL", "WO"...).

Valid settings are: A6, HOLE, SETUP8.

Not valid settings are: A, 5HOLE.

The ASCII name is always converted into capital.

The ASCII name is the mnemonic name that identifies a working in TpaCAD:

- in ASCII text
- for the general search and/or replacement commands it is then useful to set significant names.

If this is a careful choice, the name can be immediately helpful for the working recognition.

The ASCII name of the working doesn't depend on the active language.

## 13.3 Description

Description is a message that identifies the working. This information is memorized in a file and can be translated.

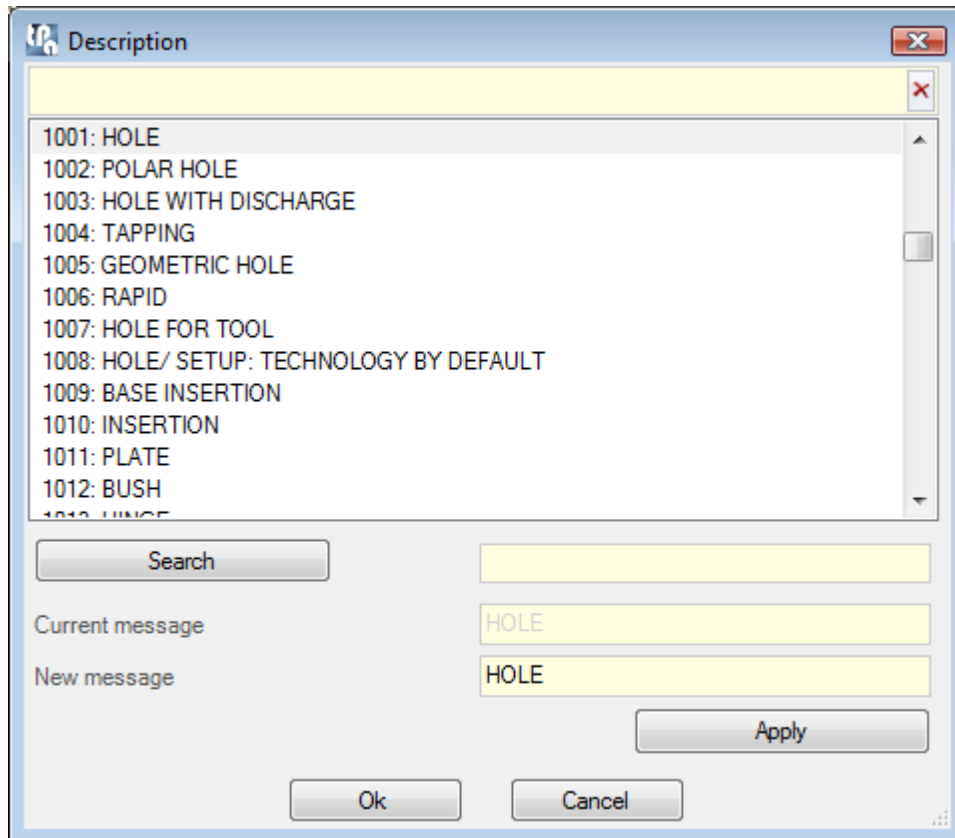
The message text can be assigned only directly if it hasn't been possible to open the messages file. On the contrary, normally the window showing messages read by files is open:

- tpacadcfg\DBWORKS.XMLng: it is the message file of base database
- tpacadcfg\custom\DBWCUST.XMLng: it is the message file created and/or modified for the custom database.

With open base database: the only messages of the first file are shown, with messages numbering from 1 to 5000: all messages, also the no-assigned ones, are shown, with modifying possibility.

With open custom database: the messages corresponding to the first file are shown (for messages with numbering from 1 to 5000) and also the ones of the second file, with numbering from 5001 a 9999. In this case:

- reference to a numbering message up to 5000: it utilizes a resource from the base file (tpacadcfg\DBWORKS.XMLng). Messages are reported with background in grey colour and compacted/compressed in the only assigned items, in correspondence to the fact that messages can't be modified;
- reference to a numbering message more than 5000: it utilizes a resource from the base file (tpacadcfg\DBWORKS.XMLng). In this case the message can be modified.



To set a direct message ( without inserting it in the file \*.XMLng) write it in the text box at the top.

To choose the message from the file:

- cancel, if necessary, writing from the text box at the top
- select the required message from the list and confirm window closing with the button **[OK]**.

In the field *Description* the message, if chosen from the messages file, is written adding at the beginning "!nn", nn is the identification number of the message in the messages file: this header that is added to the message allows translation.

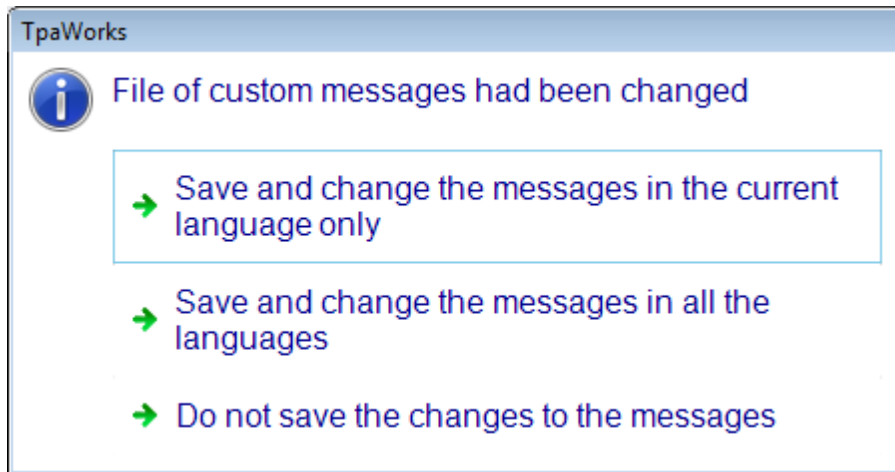
From the window it is also possible to insert or modify a message directly in file \*. XMLng:

- to modify a message: select an item in the list, assign the *New message* in the corresponding edit field and confirm adjustment with the button **[Applies]**
- to insert a message: select an empty line (without message) and go on as for adjustment of a message that is already set.

NOTE: with an open custom database it is possible to modify only the messages with numbering

**Note:**

if some messages are modified, at the exit of the database or of TpaWorks a window is shown that asks confirmation of the modifications. The window can also ask if the modifications must be applied to all languages into which the messages are translated:



The button **[Search]** enables the search, in the messages list, of the writing inserted in the box reported alongside the button.

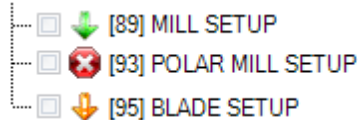
The search is enabled starting from the message that follows the current one. If any reply isn't found, the search re-starts from the beginning of the list and it ends with the message that is previous to the current one. If the writing on which the search must be made is numerical (example): "128", the search is carried out also on the numbers of the messages in the list.

## 13.4 Status

It's the working enabling status. It is possible to choose between three options:


- **Enabled**: the working is always and in anyway managed, in assignment of any work piece typology
- **Protected**: the working can be utilized only in edit phase of a macro-program.
- **Disabled**: the working is never managed.

The status of the working is reported on the corresponding node in the tree-path, as icon before the descriptive string:



- a green arrow matches the status: *Enabled*
- a Stop signal matches the status: *Disabled*
- a yellow arrow with check mark matches the status: *Protected*.

**NOTE:** it isn't a good idea to customize the base database changing the working status. It's better to use the tools of configuration available in TpaCAD. Disabling base database workings can for instance make unusable fixed cycle codes of the base database

**NOTE:** the icon on a working node can for instance be  and shows that the working is *Enabled* but can't be seen unencrypted as it utilizes a macro-program saved in an encrypted format. The situation can be:

- the base database in case of workings developed in TPA;
  - the custom database, if the manufacturer has signed the database.
- For this reason note that only an account with qualified credentials can unencrypt and modify the database.

## 13.5 Typology

It is the primary working typology: it is an important information basically for interpretation of workings. The managed typologies are six: The number and the listed items change on the basis of the working operating code:

- **Point**: it has a operating code in custom interval between 1-1000. A point working assigns a geometry (application point XYZ assigned in Cartesian or polar mode, with absolute or relative reference) and a technology (machine, group, spindle, tool, diameter). A point working is always defined by a single element
- **Setup**: it has operating code in custom interval between 1-1000. A set up working assigns a geometry



(application point XYZ assigned in Cartesian or polar mode, with absolute or relative reference) and a technology (machine, group, spindle, tool, diameter). A set up working is always defined by a single element. Generally a set up opens a profile. It assigns all or some of profile execution modes: technology, opening and closing segments, toll compensation, sped...

- **Linear:** a list of operating codes is defined that are interpreted with linear typology. Each operating code matches a particular interpretation of geometry together with the profile element is assigned. A code of linear typology can interpret a single element -a linear segment- (example: L01) or more elements - one or more linear segments or circle arcs - (an example is the working "[2219] L19:CHAMFER" composed by two linear segments);
  - **Arc:** a list of operating codes that are interpreted with arc typology is defined. Each operating code matches a particular interpretation of the geometry by means of which the profile element is assigned. A code of linear typology can interpret a single element -an arc- (example: A01) or more elements - one or more arcs or linear segments - (an example is the working "[2129] A29: Fillet" composed by a linear segment and an arc);
  - **Logical:** it has operating code in the custom interval (1-1000), or it is assigned in a list of predefined codes, or it is a code of global Function:
    - logical workings in a custom interval have an interpretation which is contextual to the execution of a piece. Examples are: programmed/scheduled stop, displacement in rapid, measurement on the work piece with probe. These working can assign a geometry and/or a technology, but the TpaCAD application program doesn't associate with them any specific interpretation;
    - workings of predefined operative code have a strictly logical functionality. They are also shown as *Logical Instructions* and they never arrive to the execution level of the work piece. Examples of *Logical instructions* are:
      - cycle instructions (IF..ELSEIF..ELSE..ENDIF; FOR..ENDFOR)
      - instruction of assignment of (j or \$) variables
      - conditional jump instructions (BREAK, CONTINUE, EXIT);
    - also the global Function Codes don't arrive to the execution level of the work piece and can be considered as *Complex logical instructions*.
- A logical instruction (simple or complex) doesn't correspond neither to a geometry nor to a technology.  
In TpaCAD: all logical workings aren't graphically shown.

- **Complex:** Sub generic working codes and fix Cycles. The term complex shows that a working can execute multiple functionality. More precisely a complex working is an aggregate of workings: generally it executes a program with any typology (program, subroutine, macro program). In case of STOOL codes: the execution is concerning not to a subroutine or macro program, but to workings that have already been programmed and called by name. Particularly, a complex working can execute (using a computer term: call) other complex workings, with a maximum nesting level of five calls.

## 13.6 Sub-typology

The sub-typology is the secondary working typology. It is useful to define under grouping of workings It may take any whole number from 0 to 100.

The attribute isn't significant for the workings of logical typology with operating code not in the custom interval.

Let's see the interpretations that are valid at the moment:

- for workings of setup typology:
  - 0 shows that it applies a vertical setup
  - 1 shows that it applies an oriented setup.
- for workings of complex typology:
  - 0 shows that it applies a cycle of point workings
  - 1 shows that it applies a cycle of millings
  - 2 shows that it applies a cycle of sawings.

In case of complex working, the information is utilized to apply the default technological settings, as they are assigned in customization of TpaCAD application program (Technology Default of Technology):

- settings referring to tabulation *Defaults for setups*: are assigned to workings with sub-typology equal to 1;
- settings referring to tabulation *Defaults for punctuals*: assigned to workings with sub-typology equal to 0.

Information arrives in work piece matrix and is available for evaluations of customized character.

### 13.7 Technological typology

The working technological typology is a field available for interpretation linked to the technology of a working.

It may take any whole number from 0 to 100.

The attribute isn't significant for the workings of logical typology with operating code not in the interval. Information arrives in work piece matrix and is available for evaluations of personalized character.

### 13.8 r Variables

The attribute is significant only for workings of complex typology (Sub generic and fixed Cycles) and defines the assignment mode of <r> public variables in application of subroutine.

Two options are possible:

- Single Edit: variables are assigned as normal parameters;
- Edit in list: variables are assigned in dedicated mode (dedicated list that is separated by remaining working parameters).

The selection of *Edit in list* it is to be assigned for the complex SUBn generic codes.

The section of *Single Edit* it is to be assigned for the remaining complex codes (fixed Cycles).

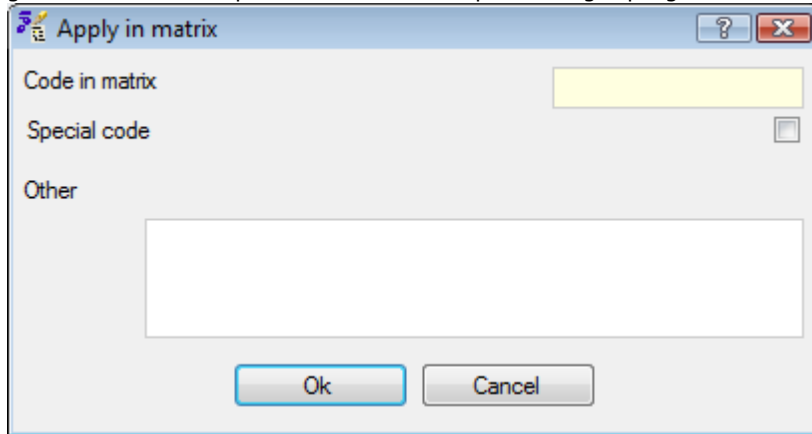
The entry is kept selected in view of particular assignments of complex typology, such as: fixed cycle with name of the macro (or of the subroutine) not set of default.

### 13.9 Apply in matrix

Field of string format for assignment of information with a significant that can be not defined previously (see what has already been written about this).

The attribute isn't significant for the workings that don't arrive to the work piece execution: of complex or logical typology not in custom interval.

The attribute assigns information of specific interest while processing a program or during its execution.

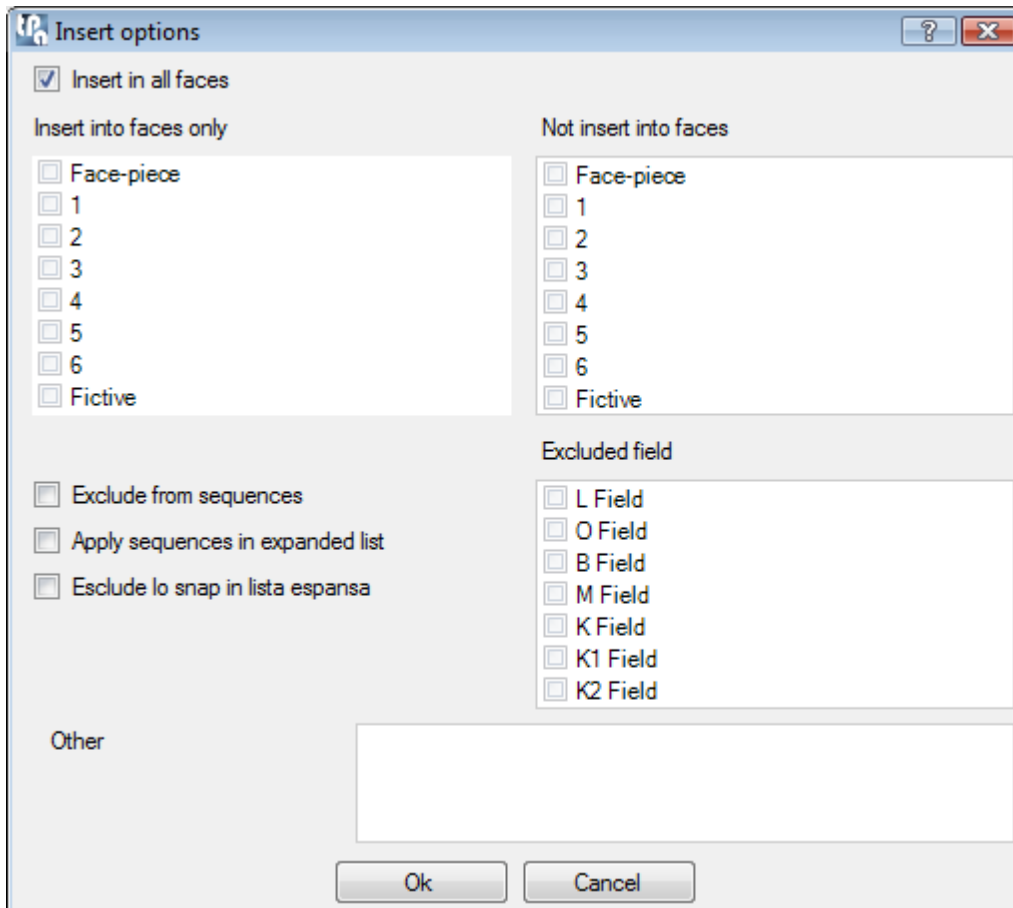


<b>Code in matrix</b>	<p>Assigns the operating code that is recorded in work piece Matrix for the working. The information is interpreted for workings with operating code between custom interval 1- 1000 that is of point, set up and logic typology. The set value doesn't need to match a working code actually assigned in the workings database. The default condition matches a null value and the operating code proper of the working is written in work piece Matrix.</p> <p>Syntax of the attribute: "cop=nn" with</p> <ul style="list-style-type: none"> <li>• nn= working (numerical) operating code</li> </ul> <p>Example: "cop=89".</p> <p>For example in the base database significant values are assigned for:</p> <ul style="list-style-type: none"> <li>• [83] POLAR DRILLING: it writes the operating code of the Cartesian hole in the matrix (81);</li> </ul>
-----------------------	---

	<ul style="list-style-type: none"> <li>• [93] SETUP POLAR MILL: it writes the operating code of Cartesian set up in matrix (89).</li> </ul>
<b>Special code</b>	<p>Select the box if the working must be handled as a special code, for application of sorting criteria. The field generally concerns custom workings of logical typology (or point typology) such as:</p> <ul style="list-style-type: none"> <li>• a programmed STOP</li> <li>• a measurement on the work piece</li> </ul> <p>In these cases the working, when inserted in a program, defines a logical separator between the first programmed lines and the last ones.</p> <p>Syntax of the attribute: "cspe=n" with</p> <ul style="list-style-type: none"> <li>• n= positive numerical value (1) if the box is selected.</li> </ul>
<b>More</b>	Here you can have information concerning particular matching criteria among workings.

### 13.10 Insert options

Field of string format in which more information can be assigned, with a significant that can be not defined previously (see what has already been written about this).  
The attribute assigns information that is particularly interesting in restrictions and working insertion modes.



#### Insertion faces

The options allow to set insertion restrictions of working, dedicated to the numbering of the faces. The entries refer to the workings:

- defined in the custom interval 1- 1000;
- complex workings: generally the codes corresponding to fixed Cycles;
- Logic instructions.

<b>Insert in all faces</b>	<input checked="" type="checkbox"/> it excludes insertion restrictions on the faces (default condition)
----------------------------	---

	<input type="checkbox"/> it allows to assign insertion restrictions on the faces
<b>Insert into faces only</b>	<p>The group of items is enabled if the item isn't selected <b>Inserts in all faces</b> . The selected entries show on which faces insertion of the working is allowed. It is possible to select each single real face, while for fictive faces selection is comprehensive. For automatic faces the choice made for fictive faces is used.</p> <p>Syntax of the attribute: "facesOn=n1,n2,..,nn" with</p> <ul style="list-style-type: none"> <li>• n1, n2,..,nn= faces numbers (0 for the work piece face, from 1 to 6 for the real faces, f character for fictive faces);</li> </ul> <p>Example: "facesOn =1,2".</p>
<b>Not insert into faces</b>	<p>The group of items is enabled if the entry isn't selected <b>Inserts in all faces</b> . The selected items show on which faces <b>not</b> it is allowed insertion of working. If they aren't assignment for <i>Work piece face</i> (see further), this group of items is valued only if items of the previous group aren't selected.</p> <p>It is possible to select each single real face, while for fictive faces selection is comprehensive. For automatic faces the choice made for fictive faces is used.</p> <p>Syntax in string format of the attribute: "facesOn=n1,n2,..,nn" with</p> <ul style="list-style-type: none"> <li>• n1, n2,..,nn= faces numbers (0 for the work piece face, from 1 to 6 for the real faces, f character for fictive faces);</li> </ul> <p>Example: "facesOn =3,5".</p>

NOTE: selections referring to fictive (and automatic) faces isn't applied with recognition of "similar face". Example: if a working can be applied only in 1, anyway can't be applied in fictive (or automatic) face, also if it is geometrically similar to face 1.

The assignments referring to the work piece face are particular, as the face itself is particular (it can insert workings in each other face). Let's see some specific cases:

The insertion is enabled in *Work piece-face* ;

- Insertion in faces (1, 2) is disabled:

It is possible to insert the working only in work piece-face and faces(1, 2) are excluded. Selections in list at the right (*It doesn't insert in the faces*) are now considered as in the other list(*Inserts only in the faces* ) it is selected only the item of *Work piece-face*.

<p><b>Insert into faces only</b></p> <ul style="list-style-type: none"> <li><input type="checkbox"/> Face-piece</li> <li><input checked="" type="checkbox"/> 1</li> <li><input checked="" type="checkbox"/> 2</li> <li><input type="checkbox"/> 3</li> <li><input type="checkbox"/> 4</li> <li><input type="checkbox"/> 5</li> <li><input type="checkbox"/> 6</li> <li><input type="checkbox"/> Fictive</li> </ul>	<p><b>Not insert into faces</b></p> <ul style="list-style-type: none"> <li><input type="checkbox"/> Face-piece</li> <li><input type="checkbox"/> 1</li> <li><input type="checkbox"/> 2</li> <li><input type="checkbox"/> 3</li> <li><input type="checkbox"/> 4</li> <li><input type="checkbox"/> 5</li> <li><input type="checkbox"/> 6</li> <li><input type="checkbox"/> Fictive</li> </ul>
<ul style="list-style-type: none"> <li>• Insertion in faces (1, 2) is enabled:</li> <li>• It isn't possible to insert the working in other faces (3,4,5,6 fictive or automatic).</li> <li>• For work piece-face there is no assignment: the insertion is enabled in work piece-face, too, but only for faces (1,2)</li> </ul>	

Insert into faces only	Not insert into faces
<input type="checkbox"/> Face-piece <input checked="" type="checkbox"/> 1 <input checked="" type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> 6 <input type="checkbox"/> Fictive	<input checked="" type="checkbox"/> Face-piece <input type="checkbox"/> 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> 6 <input type="checkbox"/> Fictive

- Insertion in faces (1, 2) is enabled:
- It isn't possible to insert the working in other faces (3,4,5,6, fictive or automatic), the work piece-face, too.

On the list at the right (*It doesn't insert in the faces*): it is possible to select only the item *Work piece-face*, while each other selection would be discarded.

Insert into faces only	Not insert into faces
<input checked="" type="checkbox"/> Face-piece <input type="checkbox"/> 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> 6 <input type="checkbox"/> Fictive	<input type="checkbox"/> Face-piece <input type="checkbox"/> 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> 6 <input type="checkbox"/> Fictive

- It is clearly enabled insertion in *Work piece face*, without any other specific selection. It is possible to insert the working only in work piece Face, without any further face restrictions

Insert into faces only	Not insert into faces
<input checked="" type="checkbox"/> Face-piece <input checked="" type="checkbox"/> 1 <input checked="" type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> 6 <input type="checkbox"/> Fictive	<input type="checkbox"/> Face-piece <input type="checkbox"/> 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> 6 <input type="checkbox"/> Fictive

- It is enabled insertion in faces (1, 2) and in *Work piece face*. It is possible to insert the working only in work piece Face and with limited faces (1, 2)

In the list on the right (*It doesn't insert in the faces*), each selection would be discarded.

Insert into faces only	Not insert into faces
<input checked="" type="checkbox"/> Face-piece <input type="checkbox"/> 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> 6 <input type="checkbox"/> Fictive	<input type="checkbox"/> Face-piece <input checked="" type="checkbox"/> 1 <input checked="" type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> 6 <input type="checkbox"/> Fictive

TpaCAD assigns the palette of workings so that in each face the workings are displayed only if they can be inserted.  
 For example in the base database it is declared that the working "BLADE X" can be applied only in face 1

and 2. This means that the working BLADE X can't be inserted in a different face, not even through call from another complex code.

### Exclude from sequences

The option **Exclude from sequences** of execution it is interpreted only if management of the Sequence field is active in TpaCAD and refers to the workings:

- defined in the custom interval (1- 1000)
- complex workings (generally the codes corresponding to fixed Cycles).

The entry must generally be activated for codes that don't correspond to actual workings: a typical case is a code (complex or point) of application of a constraint (rail, pod).

Syntax of the attribute: "fieldS=nn" with

- nn= 0 disables assignment of the sequence
- nn# 0 enables assignment of the sequence (default, in case of not assigned entry).

### Apply sequences in expanded list

The option **Apply sequences in expanded list** is interpreted only if management of the Sequence field is active in TpaCAD and refers to complex workings (generally the codes corresponding to fixed Cycles).

For complex Codes not corresponding to fixed cycles, the option is generally active. This means that it is generally necessary to assign the execution sequence for a subroutine.

For the subroutines and macro-programs that are executed with assignment of a fixed cycle:

- if the selection isn't active, it isn't necessary to assign the execution sequence. Consecutive sequence number is assigned to workings resulting from application of the complex code
- if the selection is active, who is writing the text of the macro-program cares for the assignment of the execution sequences.

Syntax in string format of the attribute: "expSeq=nn" with

- nn= 0 exclude assignment of the sequence in the expanded list (default, it matches a not selected entry);
- nn# 0 enable assignment of the sequence in the expanded list.

### Exclude the snap in expanded list

The option **Exclude the snap in expanded list** refers to complex workings (generally codes corresponding to fixed Cycles).

The option is applied in execution of quotes interactive acquisition procedure in application program TpaCAD: activate to exclude searching of snap quotes/coordinates in the expanded list of the workings. In this case: the snap is only on the application point of the complex code.

Syntax in string format of the attribute: "expSeq=nn" with

- nn= 0 enables snap functionality in the development of the working (default, it matches a not selected entry);
- nn# 0 disables functionality.

### Excluded fields

In the area named **Excluded fields** each item allows to disable assignment of a single working property: select a field to deactivate its associated property.

The item is interpreted for the workings:

- defined in the custom interval 1- 1000:
- complex workings (generally the codes corresponding to fixed Cycles).

Syntax of the attribute:

"fieldoff =\$\$\$\$" with \$\$\$\$ sequence of characters in conformity with the excluded properties.

## More

In the field *More* some significant assignments are interpreted as follows:

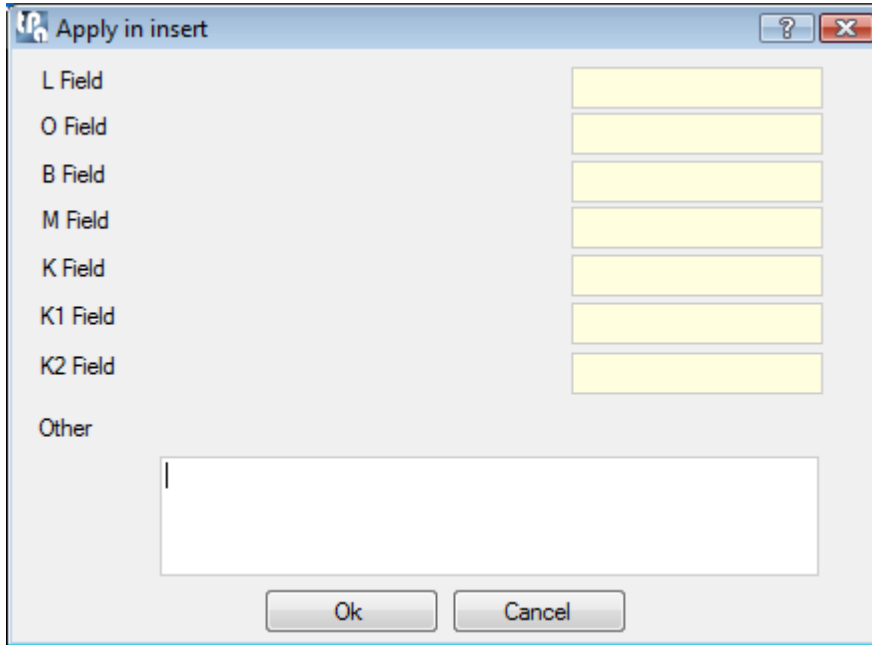
"geoxyz=n"	<p>The value is interpreted in case of application of complex workings:  "geoxyz=" field header  "n numerical value assigned to the field. Value 0 or different from 0 is interpreted.</p> <p><u>Value 1 (different from 0)</u> forces an interpretation of the application point that is the same of the point working: particularly, a no-assigned coordinate is propagated from the starting working.  <u>Value 0</u> forces an interpretation of the only assigned coordinates of the application point, particularly a no-assigned coordinate is interpreted as calculated in the text of the subroutine (or macro-program).</p> <p>[from the version of TpaCAD 1.4.2] as an alternative it is possible to assign a string field that specifies the placement of each single axis. The letters 'x', 'y', 'z' are recognized in every possible combination: Examples:  "geoxyz=xyz" amounts to setting "geoxyz=1"  "geoxyz=xy" distinguishes the interpretation between the couple of axes (x,y) and the axis z:      ≠ for the axes (x,y) it forces an interpretation of the application point similar to the case of point working,      ≠ for the z axis it forces the interpretation that corresponds to the 0 value.</p> <p><b>If the field isn't assigned: interpretation as defined in TpaCAD Configuration is applied.</b></p>
"readrsub=n"	<p>The field is interpreted in case of application of complex workings:  "readrsub=" field header  "n numerical value assigned to the field. A value 0 or different from 0 is interpreted.</p> <p><u>Value 0</u> forces public &lt;r&gt; variables of the subroutine (or macro-program) to be inherited. Particularly: a public variable that isn't set (the data-entry field is empty) is assigned following the original text.  <u>Value 1 (different from 0)</u> excludes that public &lt;r&gt; variables can be inherited. Particularly: a public variable that isn't set (its data-entry field is empty) is assigned of null value (0: if numerical; "": if string).</p> <p><b>If the field isn't assigned: interpretation as defined in TpaCAD Configuration is applied.</b></p>
"compred=n"	<p>The field is interpreted in case of application of complex workings:  "compred =" field header  "n numerical value assigned to the field. A value 0 or different from 0 is interpreted.</p> <p><u>Value 1 (different from 0)</u> forces reduced compilation in application of a subroutine (or macro-program).  <u>Value 0</u> excludes reduced compilation in application of a subroutine.</p> <p><b>If the field isn't assigned: interpretation as defined in TpaCAD Configuration is applied.</b></p>

<p>"explav=n1(t1-d1): n2(t2-d2)"</p>	<p>The field is interpreted for profile workings that calculate more sections (examples: chamfering, fillet, double arc, oval):  "explav=" field header  "n1(t1-d1)" first assignment  ":" separator for following assignment  n2(t2-d2) second assignment  ..  (by maximum 5 assignments)</p> <p>Each form "n1(t1-d1)" shows an assignment of parameter that must be applied to one or more working sections:</p> <ul style="list-style-type: none"> <li>• n1= ID of original working parameter;</li> <li>• t1= expanded segment (numerical progressive -from 1-, or typology character: 'l' for line, 'a' for arc);</li> <li>• d1 = ID of the destination parameter of expanded working.</li> </ul> <p>Let's see, for instance, working ("A29-Fillet"): <b>explav=8047(2-2008)</b>: the value of parameter [8047] of working "A29-Fillet" is assigned in automatic mode to parameter [2008] of second developed working (segment). Specifically in the working:</p> <ul style="list-style-type: none"> <li>• parameter [8047] has as a meaning "Execution speed of fillet arc";</li> <li>• the working develops a first linear segment (basic code line: L01) and a second curvilinear (basic arc of A01 code);</li> <li>• parameter [2008] of the working code A01 has as a meaning " Execution speed of segment".</li> </ul>
<p>"fieldn=n"</p>	<p>Assignment in N field N (working name) and it is interpreted, if the working field is managed:  "fieldn =" Field header  "n" numeric value assigned to the field. It is interpreted value &gt;0.</p> <p><u>Value 1</u> the <i>Working name</i> can only be seen  <u>Value 2 (&gt;1)</u> the <i>Working name</i> is hidden  (in both cases: the field holds the value as automatically assigned).</p>
<p>"fieldf=n"</p>	<p>Assignment in F field (application face) and it is interpreted, if it is in face-piece and the working field is managed:  "fieldf ="Field header  "n" numeric value assigned to the field. It is interpreted value &gt;0.</p> <p><u>Value 1</u> the <i>Application face</i> field can only be seen  <u>Value 2 (&gt;1)</u> il <i>Application face</i> field is hidden  (in both cases: the field holds the value as automatically assigned).</p>
<p>"fieldn=n"</p>	<p>Assignment in N field (working name) and it is interpreted, if the field of the working is managed:  "fieldn =" field header  "n" numeric value assigned to the field. It is interpreted value &gt;0.</p> <p><u>Value 1</u> the <i>Name of the working</i> can only be displayed.  <u>Value 2 (&gt;1)</u> the <i>Name of the working</i> is hidden.  (in both cases, the field holds the value as automatically assigned).</p>
<p>"fieldf=n"</p>	<p>Assignment in F field (application face) and it is interpreted, if in piece-face if the field of the working is controlled:  "fieldf =" field header  "n" numeric value assigned to the field. It is interpreted value &gt;0.</p> <p><u>Value 1</u> the <i>Application face</i> field can only be displayed.  <u>Value 2 (&gt;1)</u> the <i>Application face</i> field can is hidden.  (in both cases, the field holds the value as automatically assigned).</p>



### 13.11 Apply in insertion


Field of string format in which information can be assigned also with a meaning that can be not defined previously.  
 The attribute assigns information of specific interest in default settings in working insertion with exclusive use of TpaCAD.



#### Field L, Field O,....

In edit fields, default values are set that are given to the properties in insertion phase of working in TpaCAD. The entries are interpreted for the workings:

- defined in the custom interval 1- 1000:
- complex workings (generally the codes corresponding to fixed Cycles).

If the field is excluded from assignment of the working (see [Excluded fields](#) in insertion Options) the image is displayed : click the mouse on the image to display the concerning information message. In this case the set value will be ignored.

For the single fields and their set values enablings are applied as assigned in TpaCAD Configuration: particularly each single field can be not-managed or managed within an interval of limited values.

<b>Field L</b>	Value between 0 and 255. Syntax of the attribute: "fieldL=nn", nn= default value.
<b>Field O</b>	Value between 0 and 255. Syntax of the attribute: "fieldO=nn" , nn= default value.
<b>Field B</b>	Value between 0 and 255. Syntax of the attribute: "fieldB=nn" , nn= default value.
<b>Field M</b>	Value between 0 and 65000. Syntax of the attribute: "fieldM=nn" , nn= default value.
<b>Field K</b>	Value between 0 and 65000. Syntax of the attribute: "fieldK=nn" , nn= default value.
<b>Field K1</b>	Value between 0 and 255. Syntax of the attribute: "fieldH=nn" , nn= default value.
<b>Field K2</b>	Value between 0 and 255. Syntax of the attribute: "fieldJ=nn" , nn= default value.

<b>More</b>	
-------------	--

## More

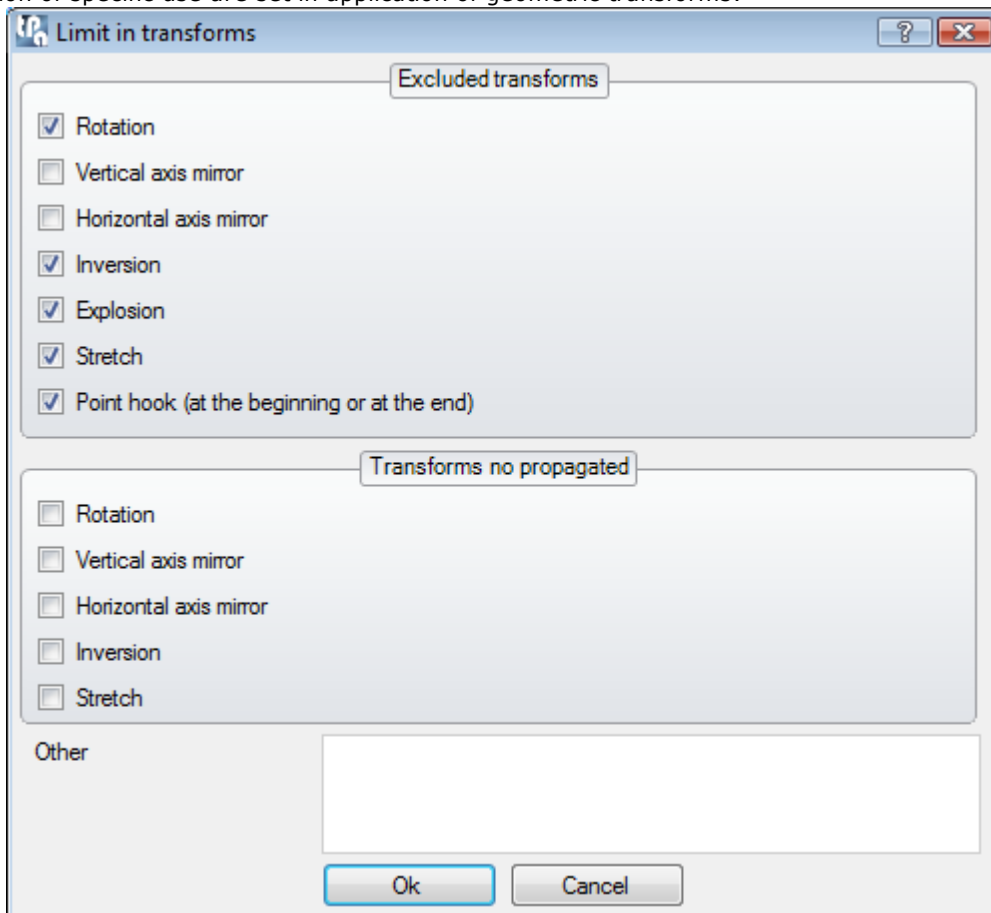
In the *More* field some significant assignments are interpreted:

"n=name%"	<p>This field sets the default value assigned to the N property (name of the working): being a non-numerical property, the value set is interpreted as a string.</p> <p>Three syntaxes illustrated below with specific assignments are recognized as follows:</p> <p><u>"n=base%51"</u> syntax interpreted: <u>"aaa" + "%" + (number between 0 and 99)</u>  A name built as a "base" + (string formatted value of the j51 variable) is set to the working. The value for the variable is evaluated with the active insertion mode (before or after the current working) and a (numerical) name of a variable between 0 and 99 (in the example: from "n=base%0" a "n=base%99");</p> <p><u>"n=base%"</u> interpreted syntax: <u>"aaa" + "%"</u>  A name built as a "base" + (numerical value not used as a string formatted value) is set to the working. The numerical value is set by looking in the entire list some inserted workings as first progressive free higher number. For example, if the names "base1", "base7", "base12" are already used, the name="base13" will be set to the working.</p> <p><u>"n=base"</u> name="base" is set to the working</p> <p>Remember that a working name is alphanumerical (letters from 'a' to 'z' and figures from '0' to '9'), beginning with a letter, in lower-case format, 16-character length max. If the name does not follow the syntax required, it is set to zero, both if it is assigned directly or as a result after the interpretation of the '%' character.</p>
-----------	---

## 13.12 Limit in transforms

Field of string format for assignment of information with a meaning that doesn't need to be defined previously (see what has already been written about this).

Information of specific use are set in application of geometric transforms:



## Excluded transforms

Assignment refers to complex workings: generally the codes corresponding to fixed Cycles. It can set exclusions in the application of geometric transforms (rotation/ x mirror/ y mirror/ inversion/...) both in insertion of the working and in tool application.

Exclusion of a transform can result from the set of geometric and/or technological considerations, rather than simply strategic or commercial. Let us see a few examples:

- working "BLADE X" (here taken as example as follows): there are both geometric and technological causes that limit its use with application of transforms;
- working of fixed Cycle developed for a specific application. For example, the explosion of the working can be blocked simply for commercial causes: in a way you don't want "to sell cheap" the effort to develop the custom macro-program. In this way the working can be used as one only block.

Assignment of the table requires of course study and attention, but it is of main importance for a "right functioning" of the working in TpaCAD environment.

Syntax of the attribute: "geoOff=c1c2..cn" with

- c1, c2,..,cn= associated character to a specific excluded transform

Example as [in the picture](#) previous: "geoOff=aiesl".

<b>Rotation</b>	Select the box to exclude possibility of rotation application. With active exclusion: <ul style="list-style-type: none"> <li>• it is recommended not to assign rotation parameters for the working (see for example parameter PRANG [ID=8044] than can be used in assignment of complex codes)</li> <li>• with direct application of the rotation Tool: the only rotation of the application point is executed.</li> <li>• with request of Rotation in nested calls of complex codes: an error is signalled.</li> </ul> Examples of possible application: sawings codes with working direction of the fixed tool (horizontal or vertical). Character of the attribute: a.
<b>Mirror on vertical axis</b>	Select the box to exclude chance to apply a symmetry around a vertical axis (x mirror) or generically oriented. With active exclusion: <ul style="list-style-type: none"> <li>• it is recommended not to assign symmetry parameters around a vertical axis (see parameters PRMIRRORX [ID=8103], PRMIRROR [ID=8106], PRMIRRORXY [ID=8105] that can be used in assignment of complex codes)</li> <li>• with direct application of symmetry Tool on vertical axis or with a generic one: no transformation is applied;</li> <li>• when Symmetry in nested calls of complex codes is required: there is error message</li> </ul> The expression "vertical axis" must be referred to the plane of the programming face: a vertical axis is always parallel to Y face axis. Character of the attribute: x.
<b>Horizontal axis mirror</b>	Select the box to exclude chance to apply a symmetry around a horizontal axis (y mirror) or generally oriented. With active exclusion: <ul style="list-style-type: none"> <li>• it is recommended not to assign symmetry parameters around a horizontal axis (see parameters PRMIRRORX [ID=8103], PRMIRROR [ID=8106], PRMIRRORXY [ID=8105] that can be used in assignment of complex codes)</li> <li>• with direct application of the symmetry Tool on horizontal axis or with a generic one: no transformation is applied.</li> </ul> The expression "horizontal axis" must be referred to the plane of the programmings face: a vertical axis is always parallel to X face axis. Character of the attribute: y.
<b>Inversion</b>	select the box to exclude the possibility/chance to apply an execution inversion. With active exclusion: <ul style="list-style-type: none"> <li>• it is recommended not to assign inversion parameter for the working (see parameter [ID=8100] that can be used in assignment of complex codes);</li> <li>• when inversion of calls nested in complex codes are required: there is error message.</li> </ul> Character of the attribute: i.
<b>Explosion</b>	Select the box to exclude possibility to apply an explosion. With active exclusion anyway it won't be possible to ask working explosion. Examples of possible application: codes of sawings or insertions. Character of the attribute: e.
<b>Stretch</b>	Select the box to exclude possibility to apply a resizing. With active exclusion:

	<ul style="list-style-type: none"> <li>• it is recommended not to assign stretch parameters for the working (see parameters [ID=8118], [ID=8119] that can be used in assignment of complex codes</li> <li>• when Stretch in calls nested of complex codes is required: there is error message.</li> </ul> <p>Character of the attribute:s.</p>
<b>Point hook</b>	<p>Select the box to exclude the chance/possibility to apply a point hook (both to the previous and to the next).</p> <p>With active exclusion:</p> <ul style="list-style-type: none"> <li>• it is recommended not to assign for the working parameters of point hook (see parameter [ID=8101] that can be used in assignment of complex codes);</li> <li>• if point Hook is required in calls nested of complex codes, it won't cause no profile hook .</li> </ul> <p>Example of possible application: sawing codes or insertions.</p> <p>Character of the attribute:I.</p>

Default condition matches all entries that aren't selected (entry "geoOff=..": not assigned).

For example let's consider working "XBLADE ":

- If the working tool can move only in X direction, rotation mustn't be allowed
- if, for example, development of workings must check horizontal direction of the tool (for example, it assures movement only along incremental X):
  - allow Mirror on vertical axis
  - parameters mustn't be assigned that correspond to the transform (for example: PRMIRRORX [ID=8103])
  - rules of geometric Transforms for parameters of typology (from 8500 up to 8799) must be assigned so that the modifications to mirror the working correctly can be showed
  - the text of the macro associated to working "BLADE X" must interpret: coordinates set to assign sawing geometry (x coordinates of beginning and end of the segment) and the variable argument of global application of the x mirror transform (submir0)
- if you want to keep protected the development of the working, the decomposition in independent blocks (explosion) mustn't be allowed
- generally it is necessary also to disable the point hook.

In TpaCAD transforms limitations that are here supposed for "X BLADE" working cause:

- with application of Rotation tool: for the working X BLADE the only one point of application is rotated in XY plane (assigned with parameters: X1 [ID=8020], Y1 [ID=8021]), while parameters of direct rotation also if configured (see parameter:PRANG [ID=8044]) aren't modified. On the contrary possible rules of geometric Transforms for parameters of typology (from 8500 up to 8799) are applied; in this case they mustn't be assigned.
  - with application of the Mirror tool around vertical axis: for working "X BLADE" X coordinate of application point is mirrored (assigned with parameter: X1 [ID=8020]). The rules of geometric Transforms assigned for parameters of typology from 8500 up to 8799 are also applied. In this particular case they must assure the actual application of the transform. Note: Parameters of direct mirror are also modified, if configured. In this case it is recommended NOT to configure
  - with request of Explosion command: working "X BLADE" is never exploded, both if directly programmed and if applied by other complex code
  - a point hook programmed at the starting of "X BLADE" working doesn't verify the hook condition between profiles
  - if the "X BLADE" working is applied by a complex code: this can't require a rotation. The request determines an error situation
  - if the "X BLADE" working is applied by a complex code and it opens its development: a point hook doesn't verify the hook condition with profile at the beginning
  - if the "X BLADE" working is applied by a complex code and it closes its development: a point hook that is required at the end doesn't verify the hook condition with continuation of the profile at the end.
- Similar considerations are applied to the other transforms.

### Transforms not propagated

Assignment refers to complex workings: generally the codes corresponding to fixed Cycles. Select a box to disable the application of corresponding transform, when the working is applied in nested calls of complex codes (example: in SUB code): wording *No propagated transforms* shows that successful application of selections is just at the level of expanded working (code *SUB generic* or of *Fixed Cycle* ). The selection of one entry of this section must be considered as an alternative to selection in section of *Transforms excluded*. Anyway: a selection is applied only if not available also in section of *Transforms excluded*.

The non-propagation of a transform is due essentially to geometric and/or technological considerations.

Let's see the example of iron fitting insertion (hinge or other fitting): the insertion can't happen with rotation. In this case, it is possible:

- to exclude rotation (See: previous table); or
- simply to block its propagation

Assignment of the table requires of course study and attention, but it is of main importance for a "right functioning" of the working in TpaCAD environment.

Syntax of the attribute: "geoOff=c1c2..cn" with  
 c1, c2,..,cn= associated character to a specific excluded transform  
 Entries in the list correspond to the section Transforms excluded, except: Explosion and point hook.

For example, let's consider the entry:

<b>Rotation</b>	<p>With active exclusion:</p> <ul style="list-style-type: none"> <li>• it is recommended not to assign rotation parameters for the working (see for example parameter PRANG [ID=8044] that can be used in assignment of complex codes)</li> <li>• with request of rotation Tool: the rotation will be executed as enabled (parameters corresponding to transform, example: PRANG [ID=8044]- and/or rules of <i>Geometric transforms</i> for the parameters of typologies (from 8500 to 8799));</li> <li>• with request of Rotation in nested calls of complex codes: just the application point in XY plane is rotated and all development of the working is <u>translated</u> integral with application point.</li> </ul> <p>Examples of possible application: codes of fitting insertion.                  Character of the attribute: a.</p>
-----------------	---

**More**

In the field *More* some significant assignments are interpreted:

"id16=nid"	<p>The value is utilized to convert fields from programs that are written with Edicad [1], in case of custom code or complex working assigned as fixed Cycle:                  "id16=" field header                  "nid" numerical value assigned to the field: it is significant if strictly positive (&gt;0).</p> <p>Value nId declares the operative code that sets the current working in Edicad database (through application program: Pastudio).</p> <p>For example:</p> <p>in working "[1061] PLATE" the field with value "id16=51" is set. When in TpaCAD a program created with Edicad is loaded, reading a working with operative code [51], code [1061] is assigned in automatic to this working.</p>
"idK=nid"	<p>The value is utilized to convert fields from programs that are written with Edicad or TpaEdi32, in case of custom code or complex working assigned as fixed Cycle:                  "idk=" field header                  "nid" numerical value assigned to the field. It is significant if positive (&gt;0).</p> <p>Value nid declares the identification number of parameter (ID) that current working must recover in K property.                  In a point working of code 300, for example, you can set field "idk=9540": when in TpaCAD a program written with Edicad or TpaEdi32 is loaded and when a working with code 300 that has assigned parameter [9540] is met, the value is recovered in field K of the working in automatic mode in TpaCAD.</p>
"idh=nid"	<p>The value is used to convert fields from programs written with Edicad or TpaEdi32, in case of custom code or complex workings assigned as fixed Cycle: the meaning is the same of field "idK=nid", only now it assigns property K1.</p>
"idj=nid"	<p>The value is used to convert fields from programs written with Edicad or TpaEdi32, in case of custom code or complex workings assigned as fixed Cycle: the meaning is the same of field "idK=nid", only now it assigns property K2.</p>

**[Note 1]** The note is useful in case of "TCN" file creation through tools that are external to Edicad,

TpaEdi32 or TpaCAD.

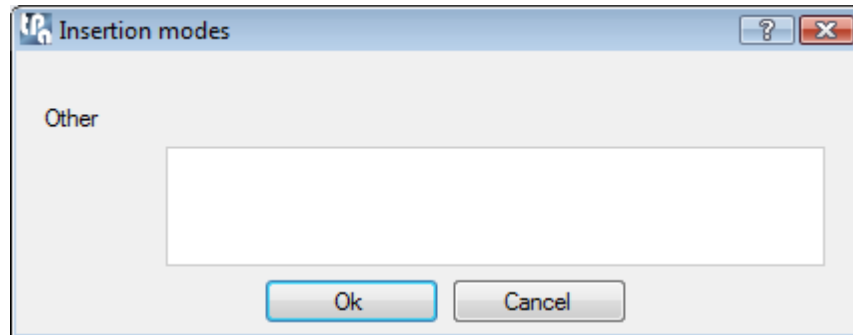
- the expression "programs written with Edicad" recognizes a file version "0.0" in the header row of the file itself;
- the expression "programs written with TpaEdi32" (that we will see after) recognizes a file version "1.0" in the header row of the file itself.

### Insertion modes

This is the field of string format in which the information can be assigned also with a meaning that isn't defined previously.

The attribute assigns information that is particularly interesting in restrictions and working insertion modes, with an interpretation aimed to the edit phase.

At the moment no entries of defined meaning are assigned.



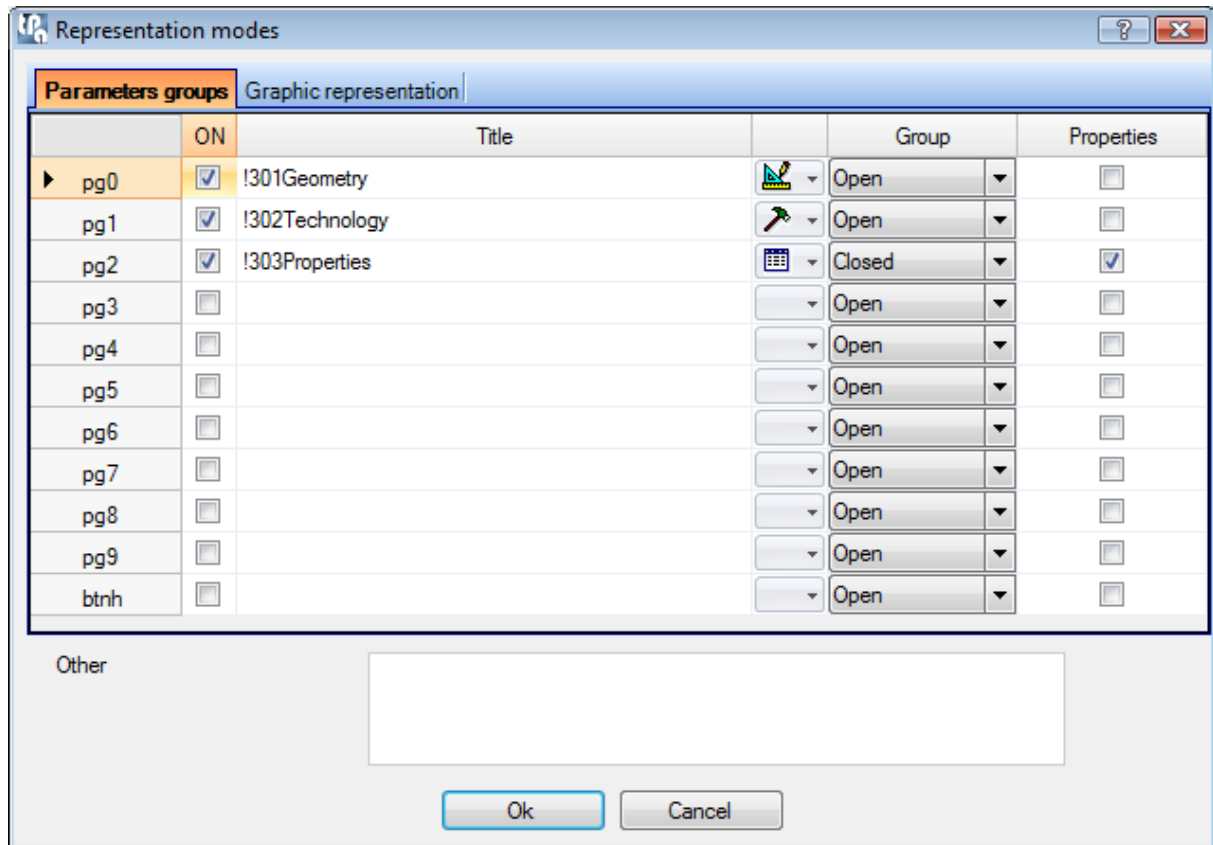
### More

"autoins=nn"	<p>This field is interpreted during the insertion with selection from the working palette or from the favourite workings:</p> <p>"autoins =" field name</p> <p>"n" numerical value assigned to the field. It is interpreted as value 0 or value not equal to 0 (positive).</p> <p><u>Value 1 (positive)</u> the insertion of the working is finished in automatic way, without the need to confirm it.</p> <p><u>Value 0 (default)</u> excludes the automatic confirmation: it corresponds to the normal functioning.</p> <p>This field is used in a high customized working database, in which the automatic insertions are normally sufficient to define ALL the workings that can be selected from the palette.</p>
--------------	--

## 13.13 Representation modes

This is the field of string format in which more information can be assigned also with a meaning that doesn't need to be previously defined.

The attribute assigns information of specific interest to the working representation modes with exclusive use in TpaCAD.



### Parameter groups

These are assignments that customize the working data entry.

Settings of the figure [Representation modes](#) can correspond to the working in the figure alongside. The groups define the nodes in which working parameters are grouped. In the figure there are 3 nodes:

- Geometry (open node)
- Technology (open node)
- Property (closed node)

The table allows to assign the groups (nodes) in which the working parameters are divided. It is possible to manage up to 10 groups, identified from pg0 to pg9. Let's examine a row of the table:

<b>ON:</b>	<input checked="" type="checkbox"/> enables the group. Discontinuous rows can be selected.
<b>Title</b>	Title of the group. It is a translatable information, if addressed in the language file (see previous cases).
<b>Icon</b>	Icon of group header. Select the icon with a mouse click. Not to display any figure click on the first icon. 
<b>Group</b>	Status of the group when the setting check of the working is assigned: <ul style="list-style-type: none"> <li>• open: the group is shown open</li> <li>• closed: the group is shown closed</li> <li>• hidden: the group isn't visible. The assignment of one or more hidden groups must be associated to the enabling of the btnh field, described as follows</li> </ul>
<b>Properties</b>	<input checked="" type="checkbox"/> shows that the group matches the expandable grouping of the working properties. In automatic mode an only one selection is managed among all enabled groups. Se no group has the field <i>Property</i> enabled, the managed properties are queued to the enabled groups.



	In TpaCAD configuration it is possible to show which properties are reported directly at the beginning of the working and which are grouped in the node that is shown here.
--	---

**Syntax of the attribute:**

"pg0=!301Geometry%1%0 pg2=!302Technology%2%0 pg3=!303Property%2%1 pg\*=3"

- "pg0=..", "pg1=..", .., "pg9=.." each entry assigns the corresponding row of the table (only the entries of the enabled groups are reported)
- "pg\*=nn" entry of indication of the property group (nn= number of the group).

Let's examine a single voice of group assignment: "pg0=!301Geometry%1%0"

- assigns group 0
- "!301Geometry": title of the group (header "!301" shows that it is indexed on the message file);  
WARNING: according to considerations relating to the kind of database and to the language currently used, it is possible that of the field only a part of the header is recorded (i.e: without the message).
- "%1" shows that display of an icon on the node is required on the left of the message: from "%1" to "%30" it matches a selected icon, "%0" matches an empty icon (no icon)
- "%0" shows that the node is open (default condition): ); "%1" for empty node is displayed.

Let's examine the assignment entry of the property group: "pg\*=3"

- if the entry shows a non-assigned node or if there isn't one, it takes the assigned node with the maximum number

The last headed row btnh manages an added field in the working (NOTE: not a node, but a single field) with following characteristics:

- no editable field or without data entry string
- the field manages opening of a dialogue box of custom type.

Management matches a guided/wizard assignment of a particular group of parameters of the working, outside the data entry check.

Three fields which are completely the same as the group table (On, Title, Icon) are assigned to the row. In these cases all parameters that have an assisted setting must be assigned in Hidden groups.

Here following the possible configuration of an active btnh field is reported:

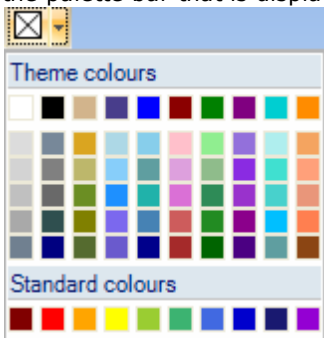


At the moment TpaCAD manages btnh field in cases of:

- COPNEWSIDE working: automatic face assignment is in dedicated window of fictive face
- complex working: parameters of emptying

**Graphic representation**

<b>Owning Node</b>	Shows the owning node in the workings tree-path The choice is possible among the assigned nodes. If the working belongs to no node: selection "***NONE**"appears.  Syntax in string format of the attribute: "pal=str" with: str= name of the owning node (default if entry isn't assigned: no owning node). Example: "pal=POLIGON"
--------------------	--

<b>Colour of graphic</b>	<input checked="" type="checkbox"/> allows to assign a particular colour for the working graphic Select the colour in the palette bar that is displayed selecting the box  The assignment isn't remarkable if the working is of logical typology.
--------------------------	--

	<p>Syntax in string format of the attribute: "rgb=nl" with nl= chosen colour. The colour can be showed in numerical format or in string. Examples: "rgb=65280", "rgb=yellow".</p> <p>If the Colour of graphic isn't assigned, the graphic of working utilizes the colours that are set in TpaCAD.</p>
<p><b>Personalized status-bar</b></p>	<p>String format field that allows to assign a personalized composition of status-bar.</p> <p>Let's take, for instance, working "[1001]" FITTING X":</p> <p>X=8020 Y=8021 Z=8022 XF=1 STEP=8512 TD=8516 the status-bar displays, for instance: "X=100 Y=600 Z=-12 XF=300 STEP=32" where 100 is the parameter value with ID=8020 600 is the parameter value with ID=8021 -12 is the parameter value with ID=8022 300 is the parameter value with ID=1 32 is the parameter value with ID=8512 12 is the parameter value with ID=8516.</p> <p>The field syntax manages up to maximum 10 assignments, separated from the space. Each assignment is of type "s=n", where:</p> <ul style="list-style-type: none"> <li>• s= displayed name (maximum length 5 characters, alphanumeric)- Ex.: "X";</li> <li>• n = ID of the parameter (ex.: 8.020).</li> </ul> <p>A particular case concerns the need to show on the status bar the value corresponding to a variable &lt;j&gt;, as a later result of the working. The typical application concerns a complex working. In this case following specifications are required:</p> <ul style="list-style-type: none"> <li>• ID =8350</li> <li>• the index of the variable must be specified in the name, with this kind of format "SS.nn" where : <ul style="list-style-type: none"> <li>• SS is a fixed string</li> <li>• nn is the index of the variable (from 0 to 99)</li> <li>• the full stop is obligatory, before the "nn" field.</li> </ul> </li> </ul> <p>Valid example: "ABC.16=8350" show the file on the status bar ".. ABC=123 ..", where the value of the j16 variable is 123 Syntax of the field: "sbar=...".</p>

## 14 Commands concerning current working



### Preview of current working

The command shows the preview of the working setting window, as it is displayed in TpaCAD. Some selections of functioning concerning the window layout can be asked. They refer to:

- Password level (Operator, Installer, Manufacturer)
- Active functionality (generic Edit, Entering or change, Technology)



### Check the working

The command checks assignments of current working:

During this check it could be necessary to carry out automatic changes (delete or adding of parameters, change of parameters attributes). Each change is displayed in its area.

Normally, assignments to current working are applied if a command on the workings palette or if another working is selected.

In case of a complex Code it is possible to update the assignment parameters of public variables of subroutine or macro-program. The following must be checked:

- the operative code matches a fix Cycle
- r Variables assignment attribute is on: Single Edit
- parameter PRSETMCRNAM is assigned, with default String set as valid and it is possible to go back to the complete addressing of a valid format file.

In this case, it is asked if you want to check correspondence with subroutine variables. Choose **[Yes]** to enable functionality. The list of parameters is updated with:

- delete of parameters that correspond to not public r variables
- adding of parameters that correspond to not listed r public variables
- adjustment of parameters typology according to changed r variables.



### Move working Up/Down

Move current working to previous or next position in the tree-path of the workings.

When moving to previous position: if the current working is already the first of a node, it is moved to the previous node (if possible).

When moving to next position: if the current working is already the last of a node, it is moved to the following node (if possible).



## 15 Working parameters

The working parameters are listed in the table reported in the lower area of the screen:

ID	ASCII Name	Status	Format
PRASSINC [8015]	EG	Edit	Integer
PRQX [1]	X	Edit	Double
PRQY [2]	Y	Edit	Double
PRQZ [3]	Z	Edit	Double
PRFI_TOOL [1002]	TD	Edit	Double
PRN_MAC [201]	TMC	Edit	Integer
PRN_GRP [203]	TR	Edit	Integer
PRN_SUBGRP [204]	EM	Edit	Integer
PRN_TOOL [205]	T	Edit	Integer

Each row assigns the attributes of a single parameter with a maximum of 300 parameter for each working.

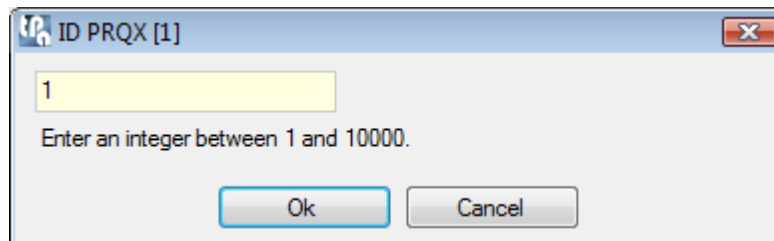
Moving the mouse cursor to a cell of the header of the table rows, an information message shows the progressive number of the corresponding row and the total number of the rows.

On the left of the table a bar of commands is reported that can be used for the management of the parameter list.

Let's examine the parameters attributes.

### 15.1 ID

The cell reports the parameter identifier in a symbolic (if it is recognized as assigned) and numeric form. in the figure the numeric identifier parameter 1 and recognized symbolic name PRQX is reported.



Set a numerical value in the interval (1-10000).

The parameters of a working must all have different ID. The ID is a univocal identifier of a parameter, together with the ASCII name.

The choice of a ID parameter must be made carefully: see documentation about detailed description of the parameters to be assigned for each working typology. Please, follow strictly provided information. Generally a parameter must be considered:

- known if a remarkable interpretation is associated to it. Generally this depends on the working typology.
- of custom use, if otherwise. If they are custom information that you want to bring in piece-matrix. **We recommend to assign an identifier in the interval [9001-9999].**

### 15.2 ASCII Name

Assigns the name of the parameter in ASCII representation and must be univocal.

Set a string that checks the format:

- length between 1 and 10 alphanumeric characters
- the first character must be alphabetic
- they are considered as reserved names, then not usable:
  - ASCII name of the working.
  - names of 2 alphanumeric characters that begin with letter 'W' (examples: "WB", "WL", "WO"...).

- for working of complex typology, the names of "Rnn" format with a numerical nn (ex.: "R45", "R1"), if utilized for parameters that don't assign r variable.

Valid settings are: P6, QX, T54, X.

Not valid settings are: 5X, 76.

The ASCII name is always converted into capital.

TpaCAD utilizes ASCII names of parameters for example for:

- make comprehensive assignments of settings
- assign technological filters of display
- they appear in ASCII text of the program, too.

It is important to adopt general criteria of assignment of ASCII codes for all workings of a database: also different parameters but of same significance (geometrical or technological) must have same ASCII name.

They include for example:

- utilize the same name for each setting parameter of machine number and each other technological parameter: group, spindle, tool, diameter, speed,...

Let's see some examples of the base database:

node	working	Machine parameter	Tool parameter
HOLE	[81]HOLE	PRN_MAC [201]	PRN_TOOL [205]
HOLE	[1002]REPEAT X	8525	8527
WOOD	[1024]OVAL	8520	8522
..			

for all these workings are assigned:

- ASCII name of machine parameter: **TMC**;
- ASCII name of tool parameter: **T**.

In assignment of a custom working that sets technology: ID of the machine parameters will be determined by <r> variables list of subroutine, but must have ASCII name: **TMC**.

- utilize the same name for each geometrical parameter (application points, tangency line,...).

The ASCII name of the working doesn't depend on the active language.

## 15.3 Status

This is a selection in a three entries list:

- **Edit**: the parameter is visible and can be modified (default);
- **Only visible**: the parameter is visible but cannot be modified
- **Hidden**: the parameter is hidden (it isn't visible, it can't be modified).

In cases of non-editable status (Only visible or Hidden) the parameter always takes the value as defined in the field *Value \$ of default*.

In attribute Auxiliaries other assignments are examined that can change the status of the parameter, but they are applied only to parameters with Status=Edit and with a possible only one restriction adding.

## 15.4 Format

This is a selection in a three entries list:

- **Double**: the parameter has a solution that is generally numerical (with significant sign and decimal values)
- **Whole**: the parameter has a numerical solution without decimal values (it applies truncation to the assigned or calculated value)
- **String**: the parameter has a solution of string type.

Examples of parameters to be assigned with format:

- Double: quotes, displacement speed
- Whole: technology identifier (machine, group, tool), counters, function selectors, rotation speed
- String: subroutines names, writing text, name of a font.

The format selection mustn't be taken by mistake instead of assignment mode that is possible for a parameter. Assignment is always allowed in a generally parametric form, on the contrary the format selection shows how to interpret the parameter. A first distinction is between:

- numeric interpretation (double, integer)
- non-numerical interpretation (string).

The setting "Pippo" is for example correct in case of string format parameter, while it isn't correct for a numerical format.

Considering the numerical formats, the value of setting "1000.567" is:

- 1000.567 in case of double format
- 1000 in case of whole format.

## 15.5 Minimum value and maximum value

The remarkable minimum and maximum value in case of parameters of numerical format (Double and Whole) are set. See TpaCAD Configuration for enabling to carry out check on values that are here assigned. With both values assigned to zero (0) check on allowed values is disabled.

Values are also utilized in setting of List control typology (see paragraph: [Auxiliaries, control Typology](#) ).

## 15.6 Default \$ values

These are two fields of string type that have multiple applications:

- in case of parameter that can't be modified (Only visible or Hidden) assigns the parameter value in a univocal way. The parameter takes always and anyway default value here set
- in case of parameter that can be modified sets starting value, in phase of [working insertion](#).

The two values are applied in case of program written in [mm] or in [inch]. The value for the inches program can't be set in case of parameter of string format so the value of the program in mm is always and only used, as in case of empty field.

### Particular case:

If the parameter assigns the name of a macro in a fixed cycle code, the management window of the files is opened, so that the file is directly selected. The possibility to manage directly the value remains by entering edit and clicking the button **[Shift]**.

To assign the default value it is possible to use the parametric programming functions. For example in working "[91] SETUP INCLINED MILL":

- parameter PRQB[5] has default value "geo[beta]",
- parameter PRQB[4] has default value "geo[alfa]",

## 15.7 Dimension

This is a selection in a three entries list:

- [none]: the parameter is non-dimensional
- [mm]: the parameter has a dimension of linear measure ([mm], [inch])
- [m/min]: the parameter has a dimension of displacement speed ([m/min], [inch/sec]).

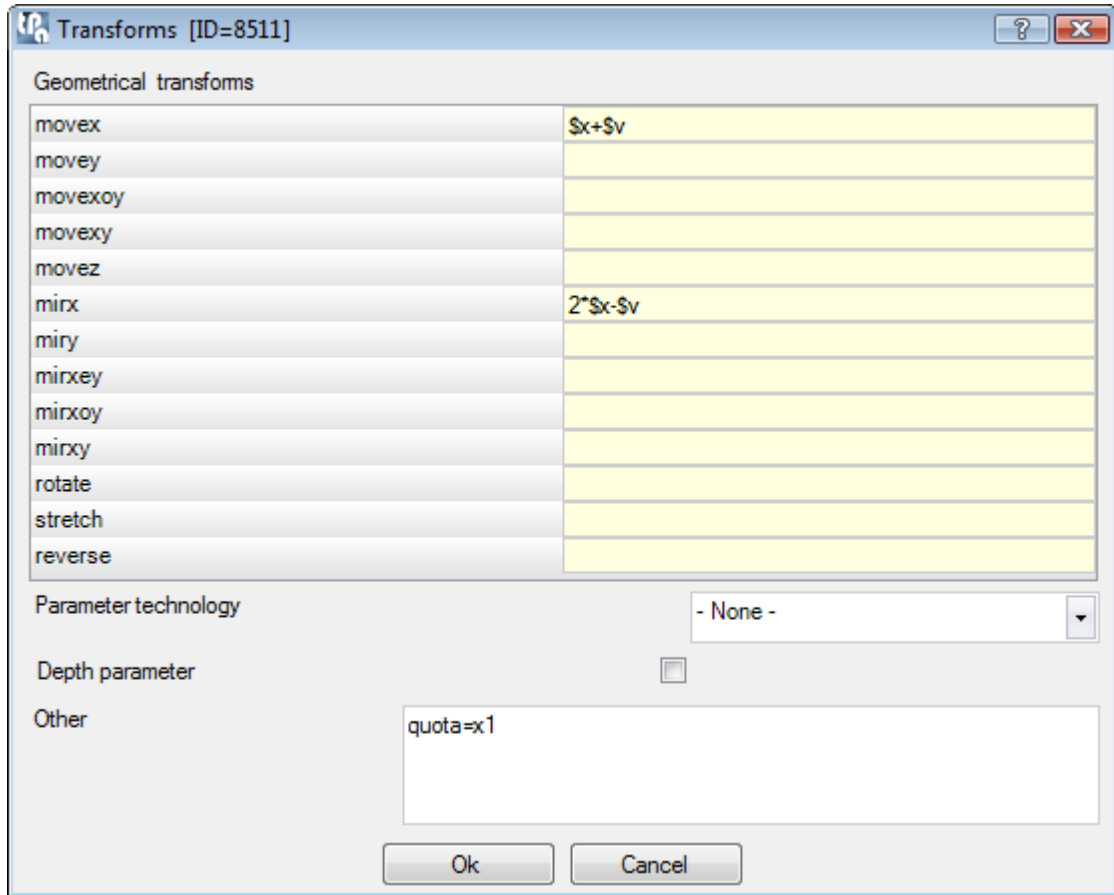
The attribute is used for conversions applied during the execution of:

- conversion tools of a program; or
- during the creation of the piece-matrix, in case a conversion of the program unit of measure is required (the program is written in [inch] and the piece-matrix is written in [mm]).

## 15.8 Transforms

The page sets fields of string format in which information can be assigned also with a meaning that isn't defined previously.

Information is set that is used in geometric transforms application and/or during assignment of particular modes, besides limitations of parameter use.



**Geometrical transforms**

The settings in the table are remarkable for assignment parameters of <r> variables in cycle workings (parameter typology: from 8500 up to 8799).

The assignment of the table for assignment parameters of <r> variables in a fixed Cycle working needs study and attention and is basically for a "correct use" of the working in TpaCAD:

the table assigns the rules that are interpreted in case of request of a specific tool of geometric transform.

An example is application of Symmetry tool. We referred to these settings in regards to working attribute **Limits in transforms**.

Let's interpret the listed items in detail:

rules of	Field name	action to be applied to parameter in case of	Example as in the picture previous:
TRANSLATION	<b>movex</b>	x translation	$x+v$
	<b>movey</b>	y translation	
	<b>movexoy</b>	x or y translation. Defines transforms that mustn't be summed in case of translation both in x and in y.	
	<b>movexy</b>	(x + y) translation. Defines transforms that must be summed in case of translation both in x and in y.	
	<b>movez</b>	z translation	
SYMMETRIES	<b>mirx</b>	x symmetry (symmetry around parallel axis to y axis)	$2x+v$
	<b>miry</b>	y symmetry (symmetry around parallel axis to x axis)	
	<b>mirxey</b>	symmetry (x + y) (symmetry around a point). Defines transforms that must be summed in case of	



		mirror (x + y)	
	<b>mirxoy</b>	x or y symmetry. Defines transforms that <u>mustn't</u> be summed in case of mirror (x + y).	
	<b>mirxy</b>	Symmetry around an inclined axis	
ROTATION	<b>rotated</b>	rotation	
SCALE	<b>stretch</b>	scale	
INVERSION	<b>reverse</b>	inversion	

Examples of assignment of table fields, as in workings database:

Value	Action	Example
mirx=-\$v	Change sign with x mirror	FITTING X step
miry=-\$v	Change sign with y mirror	FITTING Y step
mirxoy=-\$v	Change sign with x or y mirror	
mirx=180-\$v	Symmetry around angle vertical axis (value-180)	Rotation angle:
miry=-\$v	Symmetry with angle x mirror (denies the value)	Rotation angle:
mirxy=\$v-180	Symmetry	Rotation angle:
mirxoy=ifcase[\$v;=;0;1;0]	Symmetry with x mirror of Boolean value (1 becomes 0 and viceversa)	Arc rotation
mirxoy=case[\$v;1:2;2:1;0]	Symmetry with x mirror of Boolean value (1 becomes 2 and viceversa).	Tool compensation

To assign the single table fields it is possible to use parametric programming functions (for example: ifcase, abs), but the only parametric expressions admitted are:

- "\$v" = parameter setting before transform. If parameter setting is numerical (for example: 100), the value is directly set. Example: with mirx="180-\$v", the parameter setting becomes =180-100=80). If the parameter setting is parametric (example: r0), the transformation formula is maintained as it is here showed. Example: with mirx="180-\$v", the parameter setting becomes =180-(r0))
- "\$x" = auxiliary setting, with specific meaning given by the transform
- "\$y" = auxiliary setting, with specific meaning given by the transform
- "\$i" = auxiliary setting, with specific meaning given by the transform
- "\$j" = auxiliary setting, with specific meaning given by the transform
- "\$z" = auxiliary setting, with specific meaning given by the transform
- "\$a" = auxiliary setting, with specific meaning given by the transform
- "\$#nn" = auxiliary setting of relevant meaning: with nn = typology of other working parameter, the string is replaced by the parameter value before the transform (0 if the parameter isn't assigned).

#### Application criteria of assignments in case of translation

	Applies the reported criteria, with priority assigned by items sequence:	Meaning of auxiliary settings:
(x+y) + translation. (Optional) z	<ul style="list-style-type: none"> <li>• if it finds field <b>movexy</b>: applies and ignores the other assignments; otherwise</li> <li>• if it finds field <b>movex</b>: applies and ignores other assignments; otherwise</li> <li>• if it finds field <b>movey</b>: applies and ignores other assignments; otherwise</li> <li>• if it finds field <b>movez</b>: applies.</li> </ul>	"\$x"= translation x offset, "\$y"= translation y offset, "\$z"= translation z offset (0.0 if z doesn't translate).
x + translation (Optional) z	<ul style="list-style-type: none"> <li>• if it finds field <b>movexoy</b>: applies and ignores other assignments; otherwise</li> <li>• if it finds field <b>movex</b>: applies and ignores other assignments; otherwise</li> <li>• if it finds field <b>movez</b>: applies.</li> </ul>	"\$x"= translation x offset, "\$y"= translation y offset = 0.0, "\$z"= translation z offset (0.0 if z doesn't translate).
Y + translation (Optional) z	<ul style="list-style-type: none"> <li>• if it finds field <b>movexoy</b>: applies and ignores other assignments; otherwise</li> <li>• if it finds field <b>movey</b>: applies and ignores other assignments; otherwise</li> </ul>	"\$x"= translation x offset = 0.0, "\$y"= translation y offset, "\$z"= translation z offset (0.0 if z doesn't translate).

	• if it finds field <b>movez</b> : applies.	
Z translation	• if it finds field <b>movez</b> : applies.	"\$x"= translation x offset = 0.0, "\$y"= translation y offset = 0.0, "\$z"= translation z offset,

### Application criteria of assignments in case of mirror

	Applies the reported criteria, with priority assigned by items sequence:	Meaning of auxiliary settings:
Mirror (x+y)	<ul style="list-style-type: none"> <li>• if it finds field <b>mirxey</b>: applies and ignores other assignments; otherwise</li> <li>• if it finds field <b>mirx</b>: applies and ignores other assignments; otherwise</li> <li>• if it finds field <b>miry</b>: applies.</li> </ul>	"\$x"= x quote of the point of symmetry axis, "\$y"= y quote of the point of symmetry axis,
Mirror x	<ul style="list-style-type: none"> <li>• if it finds field <b>mirxoy</b>: applies and ignores other assignments; otherwise</li> <li>• if it finds field <b>mirx</b>: applies.</li> </ul>	"\$x"= x quote of the vertical symmetry axis
Mirror y	<ul style="list-style-type: none"> <li>• if it finds field <b>mirxoy</b>: applies and ignores other assignments; otherwise</li> <li>• if it finds field <b>miry</b>: applies.</li> </ul>	"\$y"= y quote of the symmetry horizontal axis
Mirror xy (inclined axis)	• if it finds field <b>mirxy</b> : applies.	"\$x"= x quote of 1° point on symmetry axis, "\$y"= y quote of 1° point on symmetry axis, "\$i"= x quote of 2° point on symmetry axis "\$y"= y quote of 2° point on symmetry axis,

### Application criteria of assignments in case of rotation

with field <b>rotated</b> assigned: applies.	Meaning of auxiliary settings: "\$i"= rotation centre x (absolute), "\$j"= rotation centre y (absolute), "\$a"= value of rotation angle in relative positioning.
--	---

If the field **rotated** isn't assigned and a translation rule x or y or (x + y) is set, the translation rule is applied.

### Application criteria of assignments in case of stretch

with field <b>stretch</b> assigned: applies.	Meaning of auxiliary settings: "\$a"= stretch factor
--	---

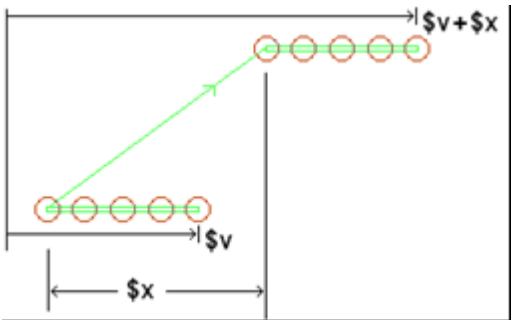
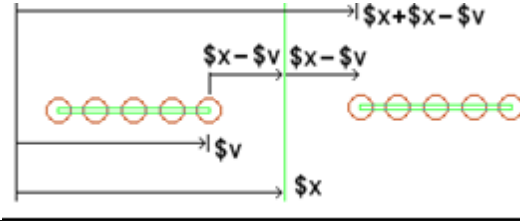
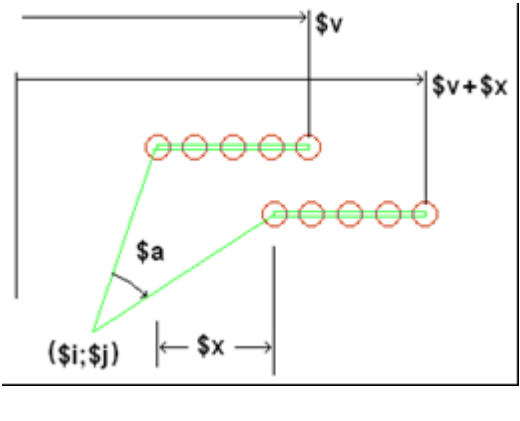
### Application criteria of assignment in case of inversion

with field <b>reverse</b> assigned: applies.	Meaning of auxiliary settings: "\$#nn"= interprets only assignments for values of other parameters
--	---

### Complete example

Each picture shows the case of application of a tool to working "[1001] FITTING X" of the base database and specifically it is showed how the end position X of the fitting is recalculated:

Translation • \$v is the original position of the end quote X • \$x is the translation offset (it measures the	"movex=\$x+\$v"
--	-----------------

<p>displacement applied to the first hole)</p>	
<p>Symmetry around a vertical axis</p> <ul style="list-style-type: none"> <li>• \$v is the original position of the end quote X</li> <li>• \$x is the position (x) of the symmetry axis</li> </ul>	<p>"mirx=2*\$x+\$v"</p> 
<p>Rotation: for the parameter a rotation rule isn't assigned, while a x translation rule is assigned. In the figure:</p> <ul style="list-style-type: none"> <li>• \$v is the original position of the end quote X</li> <li>• \$x is the translation offset (it measures the displacement applied to the first hole), as assigned by rotation.</li> </ul> <p>The rotation of a FTTING X doesn't change holes direction. On the contrary the following change:</p> <ul style="list-style-type: none"> <li>• the first hole position, with translation in X and Y, as deriving from rotation around the assigned centre</li> <li>• final X coordinate, with its following translation, with application of the rule assigned in field "movex=\$x+\$v".</li> </ul> <ul style="list-style-type: none"> <li>• (\$i;\$j) give the position of the rotation centre</li> <li>• \$a is the applied rotation angle.</li> </ul>	<p>"movex=\$x+\$v" "rotated= "</p> 

## Parameter technology

The field shows the use of technological parameters.

This is a selection in list:

- **None**: no assigned technology (default condition)
- **Generic technology** : the parameter assigns a generic technology
- **Machine**: the parameter assigns the machine number
- **Group**: the parameter assigns the group number
- **Spindle**: the parameter assigns spindle number
- **Tool**: the parameter assigns tool number
- **Tool typology**: the parameter assigns tool typology
- **Tool diameter**: the parameter assigns tool diameter
- **Translation speed**: the parameter assigns a linear displacement speed (typical examples: tool entry speed for point or setup workings, interpolation speed for arc or line workings)
- **Rotation speed**: the parameter assigns a rotation speed of the spindle (typical: in set up working).

Technology information of the parameter is interpreted:

- in application of default technological parameters (the only parameters that have instructions for use of technological parameters are assigned)
- in application of particular tools (Text generation, Emptying)
- in direct assignment of technology (example: from tool table).

Selection is available for workings: with custom code, of profile typology (line, arc), complex.

Syntax in string format of the attribute: "tecno=name" with name:

- generic: generic technology
- nmac: machine
- ngrp: group
- nsgrp: subgroup (spindle)
- tool: tool
- tiptool: tool typology
- fitool: tool diameter
- feed: displacement speed
- rot: rotation speed

### Depth parameter

Select the field if the parameter must be associated to a programming quote along depth axis (programming axis Z) and it is subjected to sign switching, if required. Sign switching can occur during creation of piece-matrix.

Syntax in string format of the attribute: "zeta=nn".

### More

"id16=nid"	<p>This field is interpreted when a program that is written with Edicad is read, in case of custom code or complex working assigned as fixed Cycle:          "id16=" field header          "nid" numerical value assigned to the field. The value is significant if positive (&gt;0).</p> <p>Value nid declares the identifier. By means of this identifier the current parameter is set in Edicad database (through application program: Pastudio).          Let's examine, for instance, working "[1001] FITTING X": for parameter [8020] ("starting X") field "id16=8510" is set.          When reading a program created with Edicad, that utilizes a working with operative code "[1001], parameter [8510] is assigned in automatic mode to parameter [8020], allowing to recover the setting even if the working format has changed.</p>				
"id32=nid"	<p>The field is interpreted when a program recorded by TpaEdi32 is read, in the cases of custom code or complex working assigned as Fixed cycle. Formalism and use reflect what is expressed for the field "id16=nid".</p>				
"addv=nn"	<p>The field is interpreted when a program recorded by a working database, that later has been modified, is read. If "nn" represents a positive value (&gt;0) and if the parameter is not already available in the program, the parameter is added automatically and to it the value set by default is assigned.</p> <p>The field must be used in case of additional parameters matching an interpretation by default, that does not correspond to the 0 value (corresponding to an unassigned parameter).          This field is mainly to be coupled to control typologies: checkbox, list.</p>				
"quote=st"	<p>The field interprets the meaning of the parameter in procedure of position interactive acquisition, which is started by the assignment window of the working:          "quote=" field header          "st" string of remarkable meaning</p> <table border="1" data-bbox="363 1563 1465 2016"> <tr> <td data-bbox="363 1563 598 1787">"x" ("y", "z")</td> <td data-bbox="598 1563 1465 1787"> <p>Positioning parameter respectively along x, y, z axis. Let's require the interactive position acquisition from a parameter with assigned field "quote=x":</p> <ul style="list-style-type: none"> <li>• position in xy face plane associates x coordinate to current parameter</li> <li>• position in xy face plane associates y coordinate to the parameter with field "quote=y" (if assigned)</li> <li>• position in face depth associates z coordinate to the parameter with field "quote=z" (if assigned).</li> </ul> </td> </tr> <tr> <td data-bbox="363 1787 598 2016">"x1", "y1", "z1"</td> <td data-bbox="598 1787 1465 2016"> <p>Positioning parameter respectively along x, y, z axis that must be utilized to identify an added position.          Let's require the interactive position acquisition by parameter when field "quote=x1" is assigned:</p> <ul style="list-style-type: none"> <li>• Position in xy face plane associates x coordinate to current parameter</li> <li>• position in xy face plane associates y coordinate to the parameter with field "quote=y1" (if assigned)</li> <li>• position in face depth associates z coordinate to the parameter with</li> </ul> </td> </tr> </table>	"x" ("y", "z")	<p>Positioning parameter respectively along x, y, z axis. Let's require the interactive position acquisition from a parameter with assigned field "quote=x":</p> <ul style="list-style-type: none"> <li>• position in xy face plane associates x coordinate to current parameter</li> <li>• position in xy face plane associates y coordinate to the parameter with field "quote=y" (if assigned)</li> <li>• position in face depth associates z coordinate to the parameter with field "quote=z" (if assigned).</li> </ul>	"x1", "y1", "z1"	<p>Positioning parameter respectively along x, y, z axis that must be utilized to identify an added position.          Let's require the interactive position acquisition by parameter when field "quote=x1" is assigned:</p> <ul style="list-style-type: none"> <li>• Position in xy face plane associates x coordinate to current parameter</li> <li>• position in xy face plane associates y coordinate to the parameter with field "quote=y1" (if assigned)</li> <li>• position in face depth associates z coordinate to the parameter with</li> </ul>
"x" ("y", "z")	<p>Positioning parameter respectively along x, y, z axis. Let's require the interactive position acquisition from a parameter with assigned field "quote=x":</p> <ul style="list-style-type: none"> <li>• position in xy face plane associates x coordinate to current parameter</li> <li>• position in xy face plane associates y coordinate to the parameter with field "quote=y" (if assigned)</li> <li>• position in face depth associates z coordinate to the parameter with field "quote=z" (if assigned).</li> </ul>				
"x1", "y1", "z1"	<p>Positioning parameter respectively along x, y, z axis that must be utilized to identify an added position.          Let's require the interactive position acquisition by parameter when field "quote=x1" is assigned:</p> <ul style="list-style-type: none"> <li>• Position in xy face plane associates x coordinate to current parameter</li> <li>• position in xy face plane associates y coordinate to the parameter with field "quote=y1" (if assigned)</li> <li>• position in face depth associates z coordinate to the parameter with</li> </ul>				

	field "quote=z1" (if assigned).
"xn", "yn", "zn"	Positioning parameter respectively along x, y, z axis that must be utilized to identify an added position, with n=numerical positive value (>0). Let's require the interactive position acquisition by parameter when field "quote=x3" is assigned: <ul style="list-style-type: none"> <li>• position in xy face plane associates x coordinate to current parameter</li> <li>• position in xy face plane associates y coordinate to the parameter with field "quote=y3" (if assigned)</li> <li>• position in face depth associates z coordinate to the parameter with field "quote=z3" (if assigned).</li> </ul>
"xa" ("ya", "za")	Positioning parameter respectively along x (y, z) axis, that must be utilized to identify a position in absolute forced programming mode. An example of use is the one for the coordinates of an arc centre that have a default interpretation in relative. The acquired position is returned with formalism "a;...".
"xa1", "ya1", "za1" .. "xan", "yan", "zan"	Positioning parameter respectively along x (y, z) axis that must be utilized to identify a position in absolute forced programming mode from and to identify an added position (n=strictly positive numerical value).
	<p>It is possible to start an interactive acquisition procedure of position as configured in the attribute <b>Auxiliaries</b> of the current parameter (examined in following paragraph).</p> <p>Let's consider, for instance, working of arc typology "[2101] A01: Pf, C, CW" (arc in xy face plane):</p> <ul style="list-style-type: none"> <li>• parameters of the centre have assignment ("quote=xa" parameter PRQI[31]=abscissa; "quote=ya" parameter PRQJ[32]=ordinate);</li> <li>• assignment parameters of starting point (PRQINX[8121]; PRQINY[8122]; PRQINZ[8123]) and end point (PRQX[1]; PRQY[2]; PRQZ[3]) of the arc don't have field "quote=.." assigned: it's a default management, if enabled.</li> </ul>
"tecnum=nn"	<p>The field is significant for parameters with <b>Parameter technology</b> set: with <i>nn</i> null or positive (&gt;=0; &lt;=10) shows that the parameter must be associated to a selection group of technology. The field is managed to allow differentiated assignments of technology, with direct selection by window.</p> <p><b>NOTE:</b></p> <ul style="list-style-type: none"> <li>• assign only for parameters with technology of tool selection: <i>nmac, ngrp, nsgrp, tool, tiptool</i>.</li> <li>• if the working manages an only one grouping: don't assign the field.</li> <li>• for the default grouping: don't assign the field or assign the field not nn=0.</li> <li>• for assignments that are common to different groupings (for example: the machine): don't assign the field.</li> </ul> <p>Let's consider, for instance, a set up working that can select 2 different technologies:</p> <ul style="list-style-type: none"> <li>• assign new parameters of the working with ID custom. Example: ID=9001 for "Group 2", ID=9002 for "Tool 2";</li> <li>• let's set technology for new parameters: <i>Group technology</i> for parameter ID=9001 ("Group 2"), <i>Tool technology</i> for parameter ID=9002 ("Tool 2");</li> <li>• for parameter ID=9002 we set also a <i>Control typology</i> of <i>Custom window</i> (see further: <i>Auxiliaries</i>): parameter setting can so occur also with direct selection by technology window;</li> <li>• Let's set now technology Groups: <ul style="list-style-type: none"> <li>◦ standard parameters of technological selection (machine, group, tool) will already be assigned to the working: for Machine (ID=201) parameter <b>not</b> let's assign field "tecnum=..", so that the setting common to both technologies is maintained; for Group (ID=203) and tool (ID=205) parameters let's assign field "tecnum=0", so that the main technological grouping is created;</li> <li>◦ for parameters ID=9001 ("Group 2") and ID=9002 ("Tool 2") let's assign field "tecnum=1", so that the secondary technological grouping is created.</li> </ul> </li> </ul> <p>The described situation allows to select two assignments (Group, Tool) in an independent way on the same machine.</p> <p>Besides Set up codes the typical use application of the field is for fixed Cycle workings.</p>
"tectip=.."	<p>The field is significant for parameters whose <b>Parameter technology</b> is set: Tool or Spindle</p> <p>The field is used when the tool table opens from working edit and can assign a filter in the selection of the tools, associated to its specific working. For example, a filter of the tool typology: "1-30", "50,51,52", "100-120".</p>

	Typology and formalism of the field are not assigned, because of the customised use. When defining them, please, observe the general setting rules.
--	---

## 15.9 Description

Descriptive message of the parameter. The description is stored in a file and can be translated with exactly the same assignment modes of [description of the working](#).

The field can be assigned only directly in table if it hasn't been possible open the messages file. On the contrary, normally the window showing messages read by file tpcadcfg\DBWORKS.XMLng, or tpcadcfg\custom\DBWCUST.XMLng (respectively for the base and custom database) is open.

## 15.10 Column in matrix

The attribute defines the destination column of the working parameter in piece-matrix.

The item isn't available for logical workings of no-custom type, that anyway don't arrive in piece-matrix.

It is a numerical field:

- the minimum value is 0: it doesn't assign a saving column in piece-matrix
- the maximum value is the number of columns that is possible to assign in matrix, as defined in TpaCAD Configuration.

**NOTE:**for the parameters common to the workings of a defined typology (example: the point workings) it isn't necessary to specify here the saving column in piece-matrix. It can be assigned in a comprehensive and only way in prototype string for working typology.

On the contrary it is necessary to assign the saving column for all added custom parameters.

If the set value has already been assigned to another parameter of the list, an advise message is displayed and the value is always accepted.

## 15.11 Auxiliaries

Field of string format in which information can be assigned also with a meaning that isn't previously defined.

The attribute is expected to assign information about parameter representation modes.

## Graphic representation

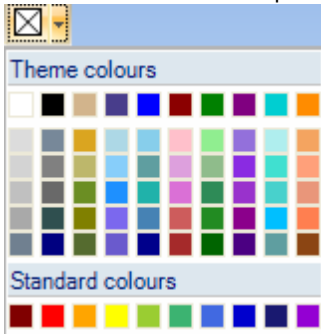
It is possible to modify directly the field in table opening the edit and clicking the button **[Shift]**.

<p><b>Owning group</b></p>	<p>Node to which the parameter belongs among the ones enabled for the working representation (it corresponds to the working attribute: <a href="#">Representation modes</a> ).          Select the check box to choose a setting: there is a numerical value from 0 to 9 at the chosen node.          The parameters assigned in the same node are queued in the order in which they are reported in the parameter table.          If the item shows a no-assigned node or if the box isn't selected, the parameter is inserted at the end of the configured groups (nodes).          Parameters aren't usually assigned in the node concerning the working properties, even if it is possible.</p> <p>Syntax in string format of the attribute: "pg=nn" with: nn= group number.          If in a node no parameter is visualized, the node itself is made invisible.</p>
<p><b>Help message (index)</b></p>	<p>Select the check box to manage a help message for the parameter. The message is memorized in a messages file and can be translated.          In the picture the cursor is on Qx line and in the messages area the help message is visualized associated to the parameter.</p>

Syntax in string format of the attribute: "hlp=nn" with: nn= message number in the messages file.

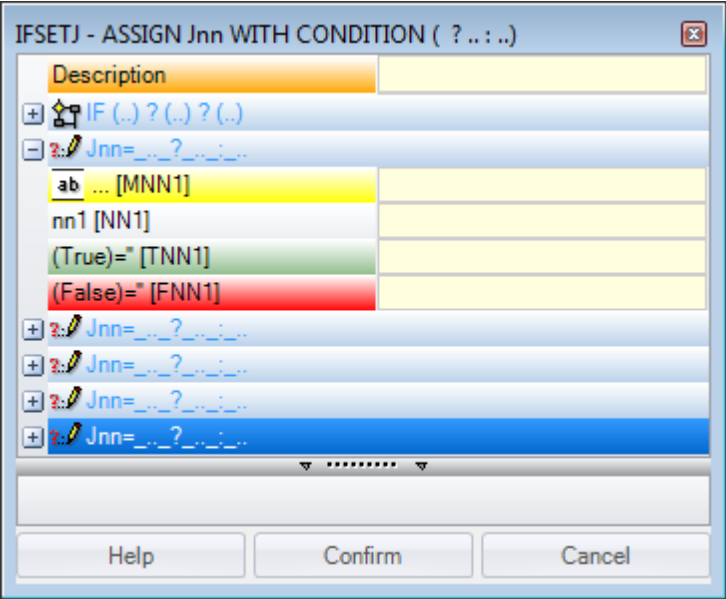
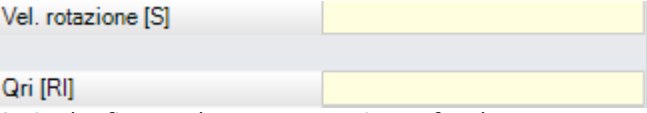
**Background colour**

Enable check box to assign a particular background colour to the parameter. Select the colour in the palette bar




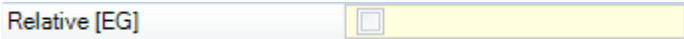
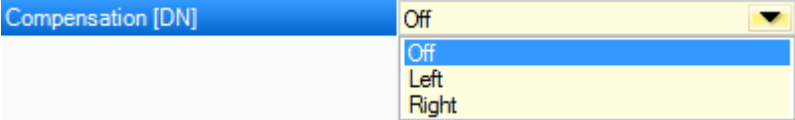
In the workings node IF of the base database, see for instance working [2015] ASSIGN Jnn WITH CONDITION (.. ? .. : ..)

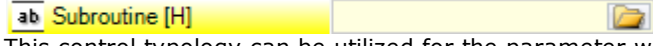
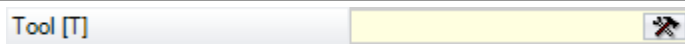


	 <p>in Jnn groups=.. ? .. : ..</p> <ul style="list-style-type: none"> <li>the setting of "j" variable that correspond to TRUE is represented with green background</li> <li>the setting of "j" variable that correspond to FALSE is represented with red background</li> </ul> <p>Syntax in string format of the attribute: "rgb=str" with: str= numerical value or norms corresponding to the colour.</p>
<b>Empty line at the end</b>	<p>Enable the check box to leave a separator line after the parameter.</p>  <p>As in the figure: the empty row is set for the parameter of rotation Speed</p> <p>Syntax in string format of the attribute: "sp=1" if it leaves a separator line.</p>

### Control typology

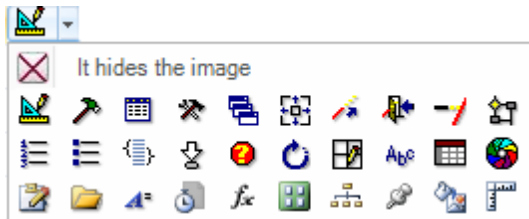
It defines the graphic control type that must be used to set the parameter . A choice between 6 different typologies is possible:

<b>Edit field</b> (default)	 <p>Direct Edit field of the parameter, only by keyboard. The setting can be generically parametric, with a format tested depending on <a href="#">Format</a> of the parameter, numerical or string. Syntax in string format of the attribute: "cnt=0".</p>
<b>Checkbox</b>	 <p>Control selected/no-selected, with values that can be assigned: 0/1 The value of the field is set in a selection box and is only numerical. Syntax in string format of the attribute: "cnt=1".</p>
<b>List</b>	 <p>Selection control in a list: setting occurs with selection of one of the items available in the list. The possible values are equal to the number of items in list. The number is taken from the attribute <b>Maximum value</b> of the parameter, increased of 1, for a maximum of items equal to 50. <b>NOTE:</b> if the <b>Maximum Value</b> set is 0, only two items in list are visualized. <b>NOTE:</b> the choice of type control <i>List</i> must match numerical (whole) parameter. <b>NOTE:</b> values progression starts from 0, at the first selection in list.</p>

	<p>Parameter setting can also occur directly, by keyboard (see further).</p> <p><u>In the example:</u> Set <i>Maximum Value</i> = 2 for the list with 3 items:</p> <ul style="list-style-type: none"> <li>• first selection matches value 0</li> <li>• second selection value 1</li> <li>• the third value 2 (= <i>Maximum Value</i> set).</li> </ul> <p>For assignment of the items in the list, the field <b>is shown. Items in list</b>, examined as follows</p> <p>Syntax in string format of the attribute: "cnt=2".</p>
<b>Browse of File Open...</b>	 <p>This control typology can be utilized for the parameter with ID=8098 (the parameter must have String format), in workings of complex typology, to assign the path of a subroutine.</p> <p>The parameter can be set directly, as <b>Edit field</b>, or the path choice can be made in window "File Open" opened by means of the button reported on the edit box. The subroutine name, together with the path, is inserted in the check/control box. The direct setting can be generically parametric, with a format tested depending on <a href="#">Format</a> of the parameter, that in the specific case must be: string.</p> <p>Syntax in string format of the attribute: "cnt=3".</p>
<b>Custom window</b>	 <p>The parameter can be directly set as <b>Edit field</b>, or from window or custom type functionality. Besides the edit box a button is visualized that, if clicked, opens the window or starts custom functionality.</p> <p>The window is personalized in accordance with the parameter ID.</p> <p>The setting can be generically parametric, with a format tested depending on <a href="#">Format</a> of the parameter, numerical or string.</p> <p><u>Cases/Examples at the moment interpreted in TpaCAD:</u></p> <ul style="list-style-type: none"> <li>• ID=8084 (PRREPORT): numerical parameter of custom error assignment in BREAK and ERROR logical instructions</li> <li>• ID=8130 (PRCOPPOINT), 8131 (PRCOPSETUP): numerical parameters of point or set up codes assignment, in complex typology workings</li> <li>• ID=8114 (PRCOUNT_PNY): numerical parameter of graphic representation of automatic face in COPNEWSIDE working (operating code: 2020)</li> <li>• ID=8117 (PRSETLAVNAM): string parameter of names assignment in programmed tool workings (STOOL codes);</li> <li>• String parameter of "r" variable assignment of complex code; in the <b>Other</b> case the field "dlgname=1" is assigned: assignment of working names, equal to the previous case.</li> <li>• numerical parameters with assignment of the field "quote=" in the attribute <a href="#">Transforms</a>: it doesn't open a window, but starts an interactive acquisition procedure of the coordinates;</li> <li>• numerical parameter with Tool or Spindle Technological assignment (see <a href="#">Workings parameters -&gt;Transforms-&gt;Parameter technology</a>): opens the technology assignment window.</li> </ul> <p>Syntax in string format of the attribute: "cnt=4".</p>
<b>Font window</b>	<p>The parameter can be set directly as <b>Edit field</b>, or it can be chosen in a window that has a font list.</p> <p>The parameter must have String format.</p> <p>Besides the edit box a button is visualized that, if clicked, opens the font window. There are two cases:</p> <ul style="list-style-type: none"> <li>• if in the box <a href="#">More</a> field "dlgfont=1" is assigned (recognizes a strictly positive value): the custom fonts are listed;</li> <li>• otherwise: the fonts installed in the system (default) are listed.</li> </ul>

If a control Typology that can open a custom window (the last three reported typology) is selected, it is

possible to choose an icon for the button besides the edit box. Click on the icon: 



If an icon is assigned: the standard one is displayed. 

**Enabling level**

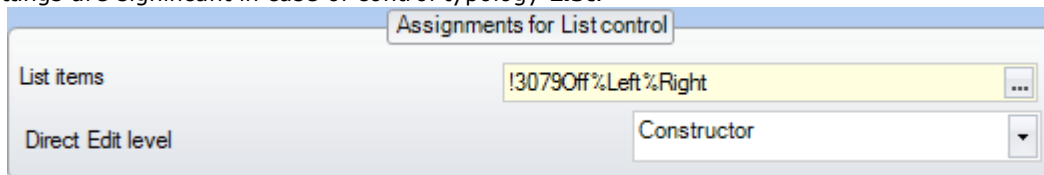
Selection in a three items list:

- **Operator**: doesn't fix restrictions to the edit possibility
- **Installer**: limits edit possibility starting from an Installer level
- **Constructor/Manufacturer**: limits edit possibility starting from the Constructor/Manufacturer level.

The field is interpreted only for parameters with Status=Edit.  
 If the access level isn't valid: the parameter is made Only visible.  
 Syntax in string format of the attribute: "keypass =nn".

**Assignments for List control**

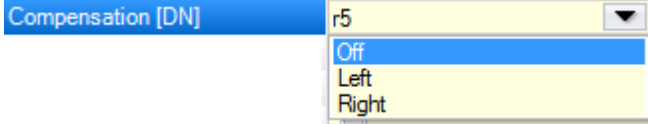
The settings are significant in case of control typology **List**.



<p><b>Listed items</b></p>	<p>The field assigns the message utilized to make up the items in the list (the message is usually memorized in a message file).  <b>NOTE:</b> the character % separate the listed items. If the message has a number of strings less than the listed items, a message in automatic mode is assigned for the missing items ("#n", with n= value assigned to the item).                  Example: message = "Off%Left%Right" corresponds to a 3 items list</p> <div data-bbox="419 1234 743 1317" style="border: 1px solid black; padding: 2px;"> <p>Off Left Right</p> </div> <p>If in the message the character doesn't appear%, the message becomes the first item of the list and the same number of messages are red for the following items, that are assigned consecutive in the messages file.</p> <p>Syntax in string format of the attribute: "items=!1250Off%Left%Right"                  • header "!1250"shows that the string is indexed in the message file  <b>WARNING:</b> according to considerations relating to the kind of database and to the language currently used, it is possible that of the field only a part of the header is recorded (i.e: without the message).  <b>Note:</b> in the message assignment it isn't possible to utilize the space. The convention of replacing the spaces with character "_" (underlined) is adopted                  Example: "items=Crf_<u>Off</u>%Crf_<u>Sx</u>%Crf_<u>Dx</u>" displays single messages: "Crf Off", "Crf Sx", "Crf Dx".</p>
<p><b>Direct Edit level</b></p>	<p>You can choose among 4 items:</p> <div data-bbox="419 1731 826 1944" style="border: 1px solid black; padding: 2px;"> <p>It doesn't allow the direct edit</p> <p>Operator</p> <p>Installer</p> <p>Constructor</p> </div> <ul style="list-style-type: none"> <li>• <b>It doesn't allow direct edit</b> : it is possible only selection in the list</li> <li>• <b>Operator</b>: the direct edit is always possible</li> </ul>

- **Installer:** allows direct edit starting from Installer level
- **Constructor:** allows direct edit starting from Constructor level.

In the figure there is an example of direct parametric setting (r5)



The setting is interpreted only for parameters with *Status=Edit* and *Enabling level valid*.

Syntax in string format of the attribute: "pass=nn" with nn level number:

- nn=-1: never allow direct edit (default if the item not available)
- nn=0: always allow direct edit (Operator Level)
- nn=1: allow direct edit starting from Installer Level
- nn=2: allow direct edit starting from Constructor Level.

### Non-operational functions

It sets one or more assignment functionalities of the working that **make not operating** the parameter. Possible choices are as follows:

- Generic Edit
- Insert, edit
- Technology

The setting is interpreted only for parameters with *Status=Edit* and *Enabling level valid*.

- **The functionality generic Edit** corresponds to the sum of the others and is used to condition management of parameters with specific enabling (as TpaCAD Configuration). At the moment TpaCAD recognizes:
  - profile Reduction parameters in tool compensation (ID=45, ID=46)
  - Variation parameter in tool compensation (ID=42): if enabled, with selection in list 2 or 3 entries, according to general Configuration and operative Mode of TpaCAD (Standard or Professional)
  - Correction parameter on Zp (ID=38): it can be On/Off
  - Inversion parameter correction side (ID=49): enabling according to general Configuration and operative Mode of TpaCAD (Standard or Professional)
  - Assignment parameters automatic face (working COPNEWSIDE; operating code: 2020): with enabling according to general Configuration and operative Mode of TpaCAD (Standard or Professional)
  - assignment parameters of the initial application point of a profile segment (ID=8121, 8122, 8123) : with enabling according to TpaCAD Personalization;
  - assignment parameters of the descriptive comments in logical instruction code (example: assignment of variables j, \$): with enabling according to TpaCAD Personalization.
- **Insert, edit** it matches the insertion phase or working modification in program list
- **The Technological functionality** matches a use of the working aimed to the only technology. In this case: the functionality must be selected for the non-technological parameters (example: quotes). In the base database, for instance, the parameters, not corresponding to technological assignment of point and set up workings, have/get activated the Technological selection. An example of use in TpaCAD is the tool *Applies set up to profile*, where it is possible set only for the working parameters of set up that don't have activated the **Technological functionality**.

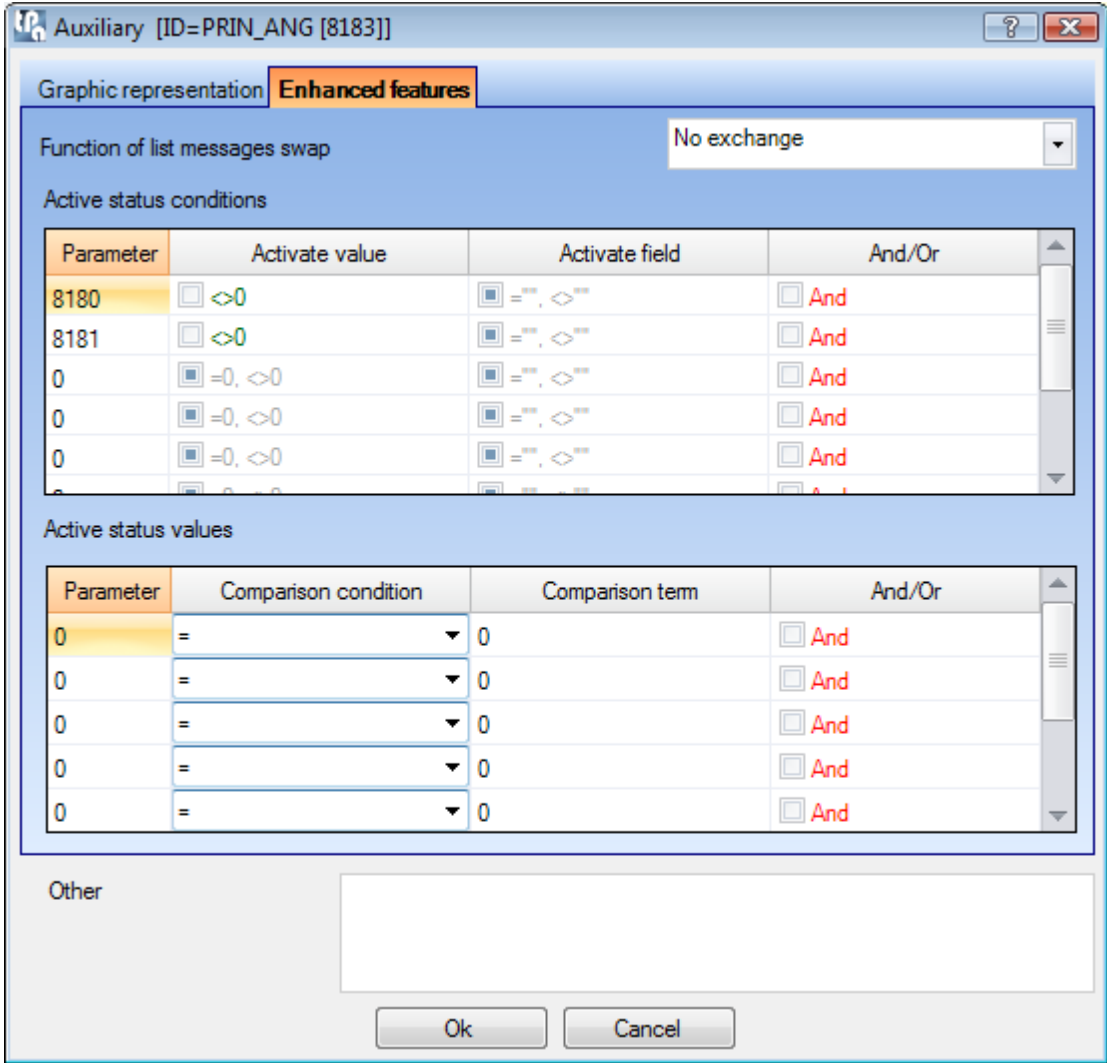
- **Visible if not operating** select the box to keep the parameter visible, even if recognized not managed according to application of the activated Functionality

**Note:** if all parameters of a group (node) are recognized not operating and with an inactive display mode, the group itself isn't displayed.

Syntax in string format of the attribute: "keyfnz=n1,n2%m":

- "n1,n2": list of the functions (one or more, separated by comma). Assigned values:
  - 1: Generic Edit
  - 2: Insertion or modification of working
  - 3: Technology assignment
- "%m" shows the visible status:
- "%0" if the not operative status matches a not visible parameter (default without assignment);
- "%1" if the not operative status matches a visible but not active parameter (it is not editable).

**Enhanced features**



**Function of list messages swap**

Selections refer to 2D (plane view) display way/way of representing of a face, so that also selection messages can be adapted according to face representation. A typical example is the selection of the rotation sense of an arc:

- chose Clockwise and see Clockwise arc
- chose Counter-clockwise and see counter-clockwise arc

Choice is made among six items:

- **No exchange**: no specific interpretation (default)
- **Rotation in xy plane**: the representation of the face plane shows the exchange of the xy rotation senses
- **Rotation in xz plane**: the representation of the face plane shows the exchange of the xz rotation senses
- **Rotation in yz plane**: the representation of the face plane shows the change of the yz rotation senses
- **Rotation in xyz plane**: the representation of the face plane shows the exchange of the xyz rotation senses
- **Vertical x-axis (and horizontal y-axis)**: the representation of the face plane shows vertical x-axis.

Let's see some examples of the base database:

<b>working</b>	<b>in the node</b>	<b>for the parameter</b>	<b>selection</b>
----------------	--------------------	--------------------------	------------------

[2101]A01 is set	ARC	Direction [EW]	Clockwise	Rotation in xy plane
[2105]A05	ARCZ		Clockwise	Rotation in xz plane
[2106]A06	ARCZ		Counterclockwise	Rotation in yz plane
[2109]A09	ARCZ			Rotation in xyz plane
[2010]SUB	SUB	Horizontal Mirror [EMX]	<input type="checkbox"/>	Vertical X - axis
		Vertical Mirror [EMY]	<input type="checkbox"/>	

If the assigned function is verified as active:

- if the check is of list typology, with 2 items in list: exchange the 2 messages.  
Message example= "Clockwise%Counterclockwise" and verified as active the exchange condition, interprets message= "Clockwise%Counterclockwise".  
**Note:** the separator character recognized inside the message: %.
- if the check is of list typology, to adapt the messages to the view, not the values.
- if the check is of list typology, with management of 3 items in list: exchange the second and the third message between them.  
Example: message= "Off%Sx%Dx" and verified as active the exchange condition, interprets: message= " Off%Dx%Sx"
- if the check is of list typology, with management of more than 3 items in list: it has no effect.
- if the check isn't a list, but it is for example an edit check, the message concerning the exchange is the one of the Description of the parameter and the exchange is always for 2 items.  
Message example="Horizontal mirror%Vertical mirror", it interprets the message= "Vertical mirror% Horizontal mirror".

Syntax in string format of the attribute: "keychg=nn" with: nn = number of the exchange function.

### Active status conditions

In the table conditions are set that can change the adjustment status of the parameter. In particular conditionings are shown in respect of other workings parameters.

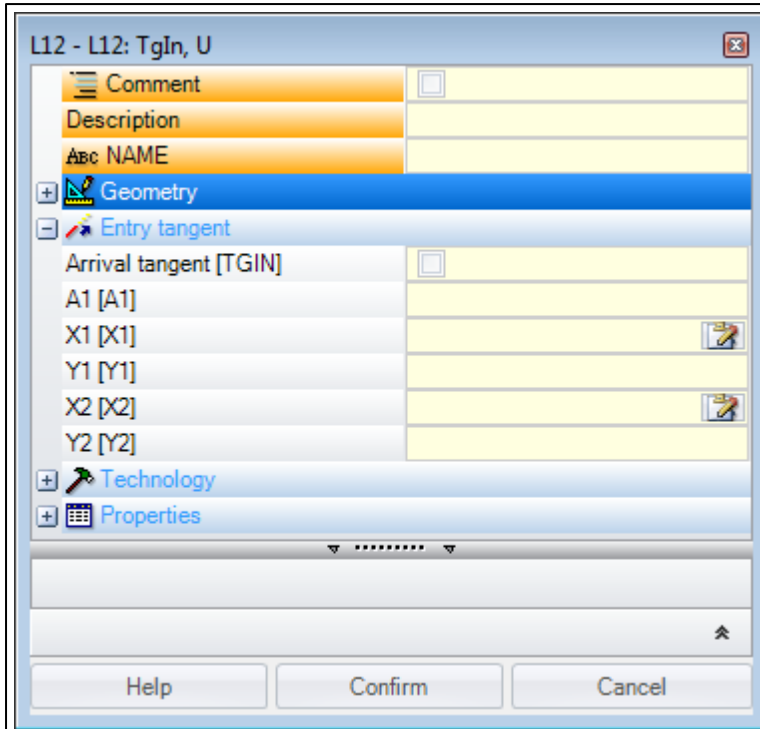
The setting is interpreted only for parameters with *Status*=Edit and *Enabling level* valid.

No conditioning is applied while editing a macro-program.

Let us see the columns of the table:

- **Parameter:** the value of the Parameter must be chosen among the ones shown in the window that is opened with a click on the box. From the list the parameters are excluded that aren't editable or already utilized in table.
- **Enables value:** sets a conditioning on the parameter value that is shown (customarily utilized to select in list or with a status box -checkbox-). The possible states are three and the switching occurs through a click on the box:
  - 0, <>0 doesn't condition on the value
  - =0 conditions on the value 0 (null)
  - <>0 conditions on values #0 (non null).
- If the setting is parametric: no conditioning on the value is set.
- **Enables field:** sets a conditioning on the setting string of the shown parameter (normally utilized for direct edit field). The possible states are three and the switching occurs through a click on the box:
  - ="" , <>"" doesn't condition on the setting
  - ="" doesn't condition on null setting (empty field)
  - <>"" doesn't condition on a non-null setting (non-empty field).
- **And/Or:** selects the way of conditioning a row/line by the following one:
  - And: as selection that it is necessary that both conditions must be verified
  - Or: as selection that only one of two conditions can be verified.

Look at an example below of the base database, for working "[2212] L12: TgIn, U", in "LINE" group:



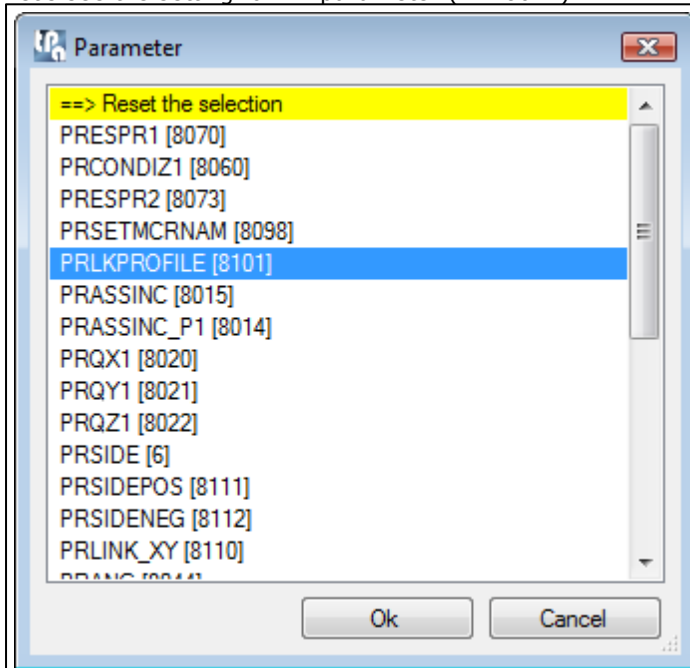
Let us examine the input Tangent node

- items from **A1** to **Y2** aren't active if the parameter is selected
- **Arrival Tangent** isn't selected, but the field **A1** isn't empty (example: A1="10"); also in this case items from **X1** to **Y2** aren't enabled.

**Let us summarize the situation for A1 and X1:**

- A1 is enabled if the arrival Tangent isn't selected
- X1 is enabled if (arrival Tangent isn't selected) and (A1 isn't set);

Let's see the setting for A1 parameter (ID=8044):



Let us examine the values of the first table row:

- **Parameter:** the ID of the arrival Tangent parameter is set (the parameter conditioning A1 state)
- **Enables value:** has value =0, to show that the current parameter state is enabled if arrival Tangent is 0 (is the value assigned to the parameter if it isn't selected)
- **Enables field:** displays "=", "<>", to show that the state of the current parameter isn't conditioned by the state of the arrival Tangent setting string

**Let us the setting for parameter X1 (ID=8020):**

Between the two rows the logical condition And is selected to select that both conditions must be verified:

Parameter	Activate value	Activate field	And/Or
8048	<input checked="" type="checkbox"/> =0	<input type="checkbox"/> "=", "<"	<input type="checkbox"/> And
8044	<input checked="" type="checkbox"/> =0, <0	<input checked="" type="checkbox"/> "=", ">"	<input type="checkbox"/> And

two rows in the table are set:

- the first row has the same values described for parameter 1
- in the second row
  - **Parameter:** displays ID of parameter A1 (it is the second parameter that conditions the state of X1)

- **Enables field:** displays ="" to show that the parameter is enabled if A1 isn't set.

A maximum of 8 conditional statements is managed that are in logical And/ Or verified in sequence.

Syntax in string format of the attribute: "keystate=v1s1c1n1:v2s2c2n2:..."

- v1: condition character on the parameter value:
  - \* doesn't assign condition on the value
  - = conditions on the value 0
  - # conditions on the value different from 0
- s1: condition character on the string set for the parameter:
  - \* it doesn't assign condition on the string
  - = conditions on the null string
  - # conditions on the non-empty string
- c1: assignment character of the conditioning type with following parameter:
  - & and (default if the character isn't assigned)
  - x or
- n1: numerical typology of the parameter
- : separator for following assignment
- v2s2c2n2 = idem for second assignment;..
- up to max 8 available assignments.

Example as parameter X1 : "keystate==\*8048:\*8044".

### Active status values

The table sets conditions that can change the parameter modification status in the same way to the previous table, but now with direct assignments on numerical settings taken by other working parameters or by global variable or by global variables of the program.

The setting is interpreted only for parameters with *Status=Edit* and *Enabling level* valid.

No conditioning is applied while editing a macro-program.

If the conditioning concerns the variables:

- the global program variables that can be used are "o", "v", "r". For the variables "r" also the "r\name" form can be used.
- the global face variables are "j".
- the TpaCAD environment global variables are "glb"and only one form of the "glb\name" type can be used.

Let us see the column of the table:

- **Comparison condition:** sets the comparison criterion of the assigned value: equal, different, greater, lesser
- **Comparison value:** assigns the value (whole,between -100000 and +100000, with which compare the parameter setting. In case of parametric setting no conditional statement/conditioning is applied.

A maximum of 8 conditionings is managed, in logical And/Or verified in sequence and it is possible to show the parameter in more rows of the table (for instance: to assess if a value is in an interval).

If the table assigns **conditioning** among (o, v, ...) variables only and if the result is **unfocused**, the current parameter is not used.

Syntax in string format of the attribute: "keystatev=v1(s1)c1n1:v2(s2)c2n2:..."

- v1: conditioning character on the parameter value:
  - = conditions equality (assesses: (parameter value)=s1)
  - # conditions non-equality (assesses: (parameter value)#s1)
  - > conditions greater value (assesses: (parameter value)>s1)
  - < conditions lesser value (assesses: (parameter value)<s1)
- s1: numerical value of comparison;
- c1: assignment character of the conditioning type with following parameter:
  - & and (default if the character isn't assigned)
  - x or
- n1: numerical typology of the parameter
- : separator for following assignment
- v2(s2)c2n2 = idem for second assignment
- ..



- up to max 8 available assignments.

#### More

"dlgfont=nn"	The field is interpreted in case of control Typology <b>Font window</b> : "dlgfont =" field header "nn" numerical value assigned to the field. The value is significant if strictly positive (>0).  With assigned positive value (example: "dlgfont=1"): the window lists the custom fonts; otherwise: the list is on the fonts that are installed in the system (default).
"dlgname=nn "	The field is interpreted in the case of Control typology <b>Custom window</b> : "dlgname =" field header "nn" numerical value assigned to the field. The value is significant if it is strictly positive (>0).  It is used for complex workings and for parameters (of "r" variable assignment, 8130 (PRCOPPOINT), 8131 (PRCOPSETUP)): the parameter assigns names of workings.
"cnt2=nn"	The field is interpreted in the case of Control typology <b>Custom window</b> : "cnt2 =" field header "nn" numerical value assigned to the field. The value is significant if it is strictly positive (>0).  It is used to manage two buttons of custom window. The value of "nn" can select the icon to represent on the second button. The utility must be previewed in a specific way in TpaCAD.

#### Cumulative selection


Some parameters of the attribute Auxiliaries can be set in a cumulative way on more parameters of the working:

- select the parameters for the cumulative assignment. Select the rows of the table clicking on the referring header, also for not in sequence rows.
- click on the header box of the Auxiliaries column.

Some settings are disabled and therefore they can't be changed in this mode. The enabled items are assigned if the same setting is displayed in all set parameters.



## 16 Configurations in TpaWorks

For TpaWorks customization select item **Options** from menu 

The displayed window shows the piece-matrix filling prototypes, as it is generated by a TpaCAD program:

	c1	c2	c3	c4	c5	c6	c7	c8	c9	c10
► Point	11	6	7	8	9	12	13	14	15	10
	COP	FACE	PROG	TIP_LAV	STIP_LAV	WO	WL	WK	WM	WB
Setup	11	6	7	8	9	12	13	14	15	10
	COP	FACE	PROG	TIP_LAV	STIP_LAV	WO	WL	WK	WM	WB
Arc	11	6	7	8	9	12	13	14	15	10
	COP	FACE	PROG	TIP_LAV	STIP_LAV	WO	WL	WK	WM	WB
Line	11	6	7	8	9	12	13	14	15	10
	COP	FACE	PROG	TIP_LAV	STIP_LAV	WO	WL	WK	WM	WB
Logic	11	6	7	8	9	12	13	14	15	10
	COP	FACE	PROG	TIP_LAV	STIP_LAV	WO	WL	WK	WM	WB

Information that is displayed here are stored in file *tpacadcfg\PROTOMX.wcad*, that is updated in installing TpaCAD (software) package: adjustments in window aren't normally possible. On the contrary assignment of custom developments must assign the saving in matrix in the referring attribute of each single working parameter.

The window shows a table with 12 rows, coupled two by two:

- the first two rows refer to the prototype for a point typology working. Following rows are:
- two rows referring to the prototype for a set up typology working.
- two rows referring to the prototype for a arc typology working.
- two rows referring to the prototype for a line typology working.
- two rows referring to the prototype for a complex typology working.
- two rows referring to the prototype for a logical typology working.

The columns of the table match the piece-matrix columns.

For each rows couple:

- the first one assigns the identification numbers/numeric identifiers of the information saved in piece-matrix
- the second one assigns the symbolic identifiers of the information saved in piece-matrix (they are optional, but can be useful in debug procedure).

### 16.1 Commands referring to working parameters

Let's see the commands bar that is displayed at the left of the parameters table.



#### Insert a parameter

Insert a parameter after current parameter.

The command doesn't work if the maximum number of parameters managed for a working (300) is already assigned.



#### Move current parameter up/down

Move current parameter respectively to previous or next position in table.

Parameters assignment order defines the showing order of setting check, respecting nodes grouping.



**Delete selected parameters**

Remove selected parameters (or the current parameter) and copies them into TpaWorks local Clipboard.



**Copy selected parameters**

Copy selected parameters (or the current parameter) in TpaWorks local Clipboard.



**Paste parameters from local Clipboard.**

Insert into the table parameters available in local Clipboard after current parameter. The command doesn't work if parameters in local Clipboard aren't available or if the maximum number of managed parameters (300) is already assigned.

## 17 The working database

The workings database is necessary for the functioning of TpaCAD. Installation of TpaCAD provides the base database, file DBWORKS.wcad in folder tpacadcfg.

The base database provides general purpose and of general interest workings:

- Drilling and set up elementary workings
- No-elementary drilling (fitting, repetitions, developments on polygons) development codes
- profile elementary codes
- profile no-elementary development codes (polygons, ellipses, pockets, texts)
- subroutines application codes
- Logic instructions.

For each specific application there is normally the necessity to define particular workings. For this reason a second custom database is managed (file DBWCUST.wcad, in folder tpacadcfg\custom), made for application needs.

**Definition of new workings or modification of existing workings must occur in custom database, so that updating of base database is assured, without losing custom information.**

Working arrangement is fully managed by TpaCAD: in particular, custom database workings can be excluded or included in TpaCAD configuration.

If there is a working both in the base database as in the custom database (same operating Code or ASCII Name), TpaCAD excludes workings management defined in base database to give greater importance to customizing.

So, if it is necessary to modify a working that is already available in the base database (adding, for example, custom parameters):

- copy the working from the base database into/in the custom one
- modify the copied working.

In the following chapters the knowledge of the workings is studied, in their general characters and in customizing modes.

### 17.1 Workings in custom codes intervals

In interval of codes (1 –1000) typology workings are assigned: point, set up and logical.

Here there are custom codes: these workings are generally defined according to the needs of a specific application.

In the database provided with the installation of TpaCAD working in this interval are assigned. These are workings:

- point ([81]HOLE, [83]POLAR HOLE)
- setup ([89]MILL SETUP, [93] POLAR MILL SETUP, [91] ORIENTED MILL SETUP, [95] BLADE SETUP)
- besides some auxiliaries codes but not available in the workings palette [2410]GEOMETRIC HOLE, [2411]GEOMETRIC SETUP, [82]TOOL HOLE, [88] MILL SETUP FOR DIAMETER).

#### Point and Setup codes

Point and setup workings have assignment and interpretation of geometry and technology. This means that:

- an application point is interpreted
- it is possible to apply a programming in relative.
- a graphic representation is assigned that shows the application point, the possible working direction and the tool overall dimensions.

Set up workings are mainly utilized to open a profile. Set up provides for technological information to be used for profiling. A set up can also be used alone, so not followed by a profile.

A point working identifies a single working. Examples of point workings are drilling workings and insertions.

To assign a new point or set up working we suggest starting copying a working that is already in the database.

Let's see the main characteristics of these workings.

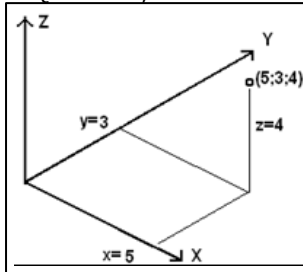
#### The application point

Workings are located in XY plane of face with depth to Z direction, perpendicularly to face plane. Values calculated for three coordinates (x, y, z) define the working point of application.

Let's open working [81]HOLE. Quotes are programmed in a system of **Cartesian coordinates**

For parameters to be assigned typology (ID column) and format (double, whole, string) are shown

- Relative: ID=8015; Whole
- Qx: ID=1; Double
- Qy: ID=2; Double
- Qz: ID=3; Double



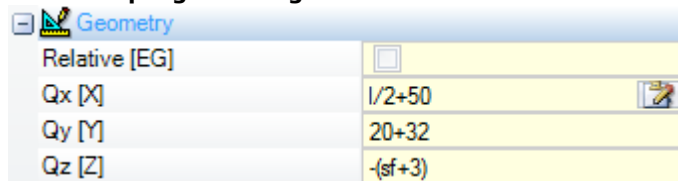
In the figure the Cartesian coordinates are shown/displayed. Coordinates directly assigned are:

- absolute from the face origin, if the case/box Relative is not selected.
- relative from the last programmed position upward, if the case Relative is selected.

If point is assigned with coordinates (x=5;y=3;z=4), but in relative mode with the last position programmed at (x=2;y=2;z=2), working shall have its own point of application at (x=7;y=5;z=6).

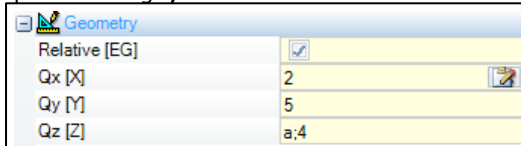
Absolute/relative selection applies to all coordinates.

**Absolute programming**



**Absolute programming**

When relative mode is selected, absolute mode can be forced on a single coordinate by placing "." before quote settinga;":

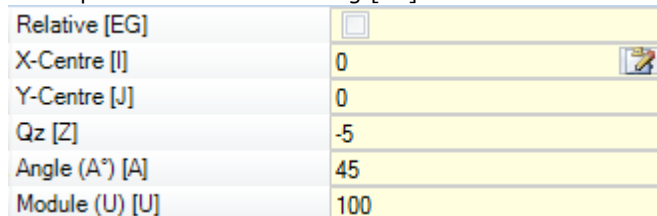


In the figure:

- coordinates X and Y are programmed in relative
- coordinate remains programmed in absolute.

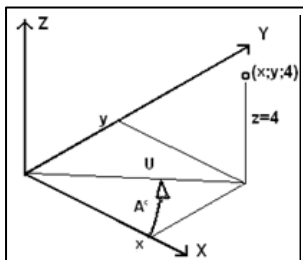
It is possible to program point or set up workings in a system of **polar coordinates** .

Let's open for instance working [83]POLAR HOLE :



For parameters to be assigned typology (ID column) and format (double, whole, string) are shown

- Relative: ID=8015; Whole
- X-Centre ID=31; Double
- Y-Centre ID=32; Double
- Qz: ID=3; Double
- Angle: ID=8044; Double
- Module: ID=8017; Double



The figure below is an example of polar coordinates. Z Coordinate is directly assigned, as in the previous case.

The position XY plane is specified by giving its distance from a centre and its angle (in degrees) in XY plane from X axis (positive angles in counter-clockwise rotation).

In the picture:

- the centre is the origin of the face (0:0);
- distance from centre is: U=100
- the angle is: A=45°.

La selection **absolute/ relative** applies now to Z coordinate and (x;y)

	coordinates of centre. When relative mode is selected , the absolute mode can be forced on a single coordinate by placing "a;" before quote setting.
--	---

### The technology

A point or set up working has a technology assigned, which depends on how working is made and in which machine, group or tool it will be executed.

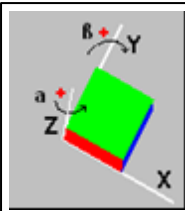
	<p>For example for a <b>point</b> we open working [81]HOLE. The parameters allow to select the technology both for Diameter and for Tool (this prevails on programming by diameters).</p> <p><b>Machine and/or Group</b>: it is generally necessary to set a value. If the fields aren't assigned value 0 is taken up.</p> <p><b>Tool typology</b> : allows to set a greater specific on the choice of the tool.</p>
<p>Diameter ID=1002; Double Machine: ID=201; Whole Group ID=203; Whole Tool ID=205; Whole Tool typology. ID=1001; Whole</p>	<p>As for example for a <b>setup</b> we open the working [89]MILL SETUP. The parameters allow to select the technology for Electrospindle and Tool.</p> <p><b>Machine and/or Group</b>: it is generally necessary to set a value. If the fields aren't assigned value 0 is taken up.</p> <p><b>Electro-spindle</b>: assigns the reference electro-spindle</p> <p><b>Tool typology</b>: it allows a greater specific on the choice of the tool.</p> <p>If in the field Electrospindle a position on the group is set that is already fitted out with a technology (of tool or tool holder), it isn't necessary to set the field Tool.</p> <p>If in the field Electrospindle a position on the group is set that isn't fitted out with a technology (it is a electro spindle position), it is necessary to set the field Tool, in order to choose which tool or tool holder fit out on Spindle (with functionality of manual or automatic tool change ).</p>
<p>Machine: ID=201; Whole Group ID=203; Whole Electro-spindle ID=204; Whole Tool ID=205; Whole Tool typology. ID=1001; Whole</p>	

It is possible to program a forcing to select the tool/ for selection of the tool, with assignment of the parameter: **Automatic tool** (ID=201; Whole). If programmed different from 0 it sets an automatic selection of the tool, by ignoring programming both of the tool number and of diameter. Selection of the tool occurs in accordance with available technology.

### Oriented tool

When a normal working condition applies, the tool is set perpendicularly to xy plane of working face. A setupworking can also assign a tool orientation from the face plane, to be defined as oriented set up. Let's open the working [91] ORIENTED MILL SETUP. The fields that define tool orientation:

Beta Angle (°) [B]	geo[beta]	(ID=5; Double): slewing angle (beta)
Alpha Angle (°) [A]	geo[alfa]	(ID=4; Double): rotation angle (alfa)



Both rotation axes have absolute programming **on the piece**.  
 With reference to the figure that represents a generic piece and the three absolute Cartesian points:

- beta rotates around the y axis
- alfa rotates around the z axis.

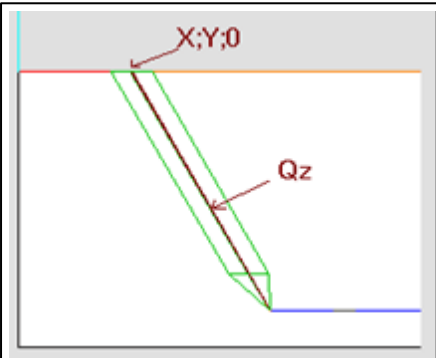
Tool rotation fields, if assigned for working, **are significant in any case**, even if they aren't set (in this case their value is 0).

With referenceto the six real faces of the parallelepiped, values for alfa andbeta are assigned as follows:

Face	Beta; Alfa	Notes
1	(0;0)	Alfa: any
2	(180;0)	Alfa: any
3	(-90;90); (90;-90)	
4	(-90;180); (90;0)	
5	(-90;-90); (90;90)	
6	(-90;0); (90;180)	

It is possible to change the programming mode of depth axis with parameter:

Orthogonal Plane Z Ref. [DZ]  (ID=47; Whole): Z reference



Orthogonal Plane Z Ref. [DZ]

with value 0 (box no-selected), programmed quote for depth (Qz: ID=3) is measured along the oriented axis of the tool (default). The depth is significant with the sign:

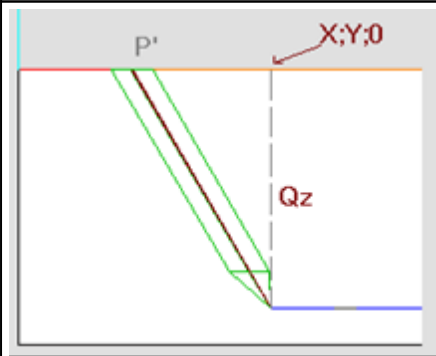
- negative value moves the drill bit from the programmed XY position along the resultant of the angles (alfa;beta);
- negative value moves the drill bit from the programmed XY position along the opposite resultant of the angles (alfa;beta);

If the (alfa;beta) angles are set correctly, so that the tool can be placed at the entry of a face:

- positive value takes the tool over the piece;
- negative value takes the working tool into the face.

The tool enters the face plane at the programmed XY coordinates, with direction assigned by the rotation and slewing angles, in accordance to the programmed depth.

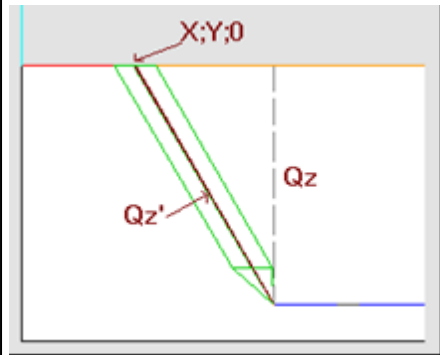
The picture shows the use of the tool into the piece, with side view in respect of the top plane of the face.  
 The forthcoming profile respects the assigned orientation on the setup.  
 This case corresponds to the default situation, which is applied to the programming in an oriented set up, even if the parameter (ID=47) is not assigned for the set up.



Orthogonal Plane Z Ref. [DZ]

with value 1 (selected box): programmed quote for the depth is measured along the plane orthogonal to the face (depth axis of the face)  
 The tool enters the face plane, with assigned direction, in P' point, so that the assigned position on the three programmed coordinates is observed.  
 The forthcoming/following profile is in any case considered as vertical profile and the orientation is useful to characterize/define set up workings.





If required, for parameter (ID=47) it is possible to interpret also value 2.

In this case: the programmed quote for depth is measured along the plane orthogonal to the face (depth axis of the face) and the tool enters the face plane at the programmed XY quote, with direction assigned and quantity ( $Qz'$ ) so that the set depth is followed.

This case is considered a variance/variant of the first examined case.

Setting of parameter (ID=47) is normally assigned with a check box (values: 0, 1) as the two cases match normally managed situations. In case it is necessary to add programming of value 2, the type of parameter data entry must be changed in list of 3 items or in field edit.

### In the only case of default mode for programming of depth axis (

Orthogonal Plane Z Ref. [DZ] ) , moreover it is possible to assign rotation axes locally, by using parameter: (ID=48; Whole):

- angle **alfa** assigns milling direction, on face plane
- angle **beta** assigns tool inclination, in respect of the perpendicular to the face plane, along the milling.

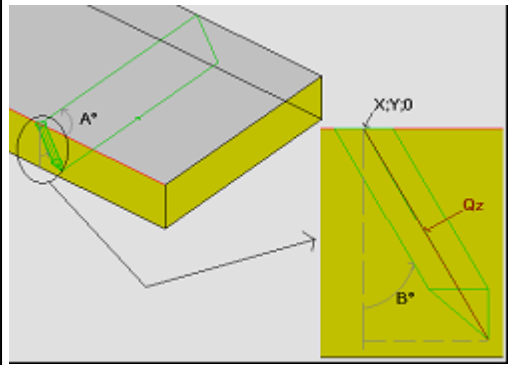
Value 0 (default)	With negative values beta: inclines to the right
1 (different from 0):	With negative values beta: inclines to the left

Parameter (ID=48), if assigned for set up working, is anyway significant, even if not set (in this case takes value 0).

Parameter (ID=48) is for instance assigned for the sawing set up, where

- the working is allowed/enabled in face 1 (top face)
- the working tool (blade) is set on the head group/unit rotated so that the previously defined working direction is followed (along x, if the blade works horizontally, along y, if the blade works vertically)

The figure below shows the sawing along Y:



At the left a possible representation in a 3D sight is shown, while at the right the set up is shown in detail:

- the quote programmed for the depth ( $Qz$ : ID=3) is measured along the oriented axis of the tool
- the slewing inclines the tool along the vertical segment.

The tool enters the face plane at the programmed XY coordinates, with direction assigned by the rotation and slewing angles, in accordance to the programmed depth.

If in the working parameter (ID=48) has value 0: this case in the figure matches a negative slewing programming (for instance: -30).



If in the working parameter (ID=48) has value 1: this case in the figure matches a positive slewing programming (for instance: 30).

### Additional information in setup

Set up workings interpret additional parameters, specific in profile assignment.

Let's open working [89]MILL SETUP :

<input type="checkbox"/> <b>Advanced technology data</b> Point hook [EGL] <input type="checkbox"/> Multiple setup [MLT] <input type="checkbox"/> Emptying profile [EMP] <input type="checkbox"/> Geometric profile [GEO] <input type="checkbox"/>	<b>Assignment parameters:</b> <ul style="list-style-type: none"> <li>• (ID=8101; Whole): hook among distinct programmed profiles</li> <li>• (ID=8096; Whole): multiple set up</li> <li>• (ID=8095; Whole): profile set up generated by an emptying procedure (tool or code) or to be considered as this</li> <li>• (ID=8094; Whole): geometric profile set up.</li> </ul>
---	---

<div style="border: 1px solid black; padding: 5px;"> <p> <b>Advanced technology data</b></p> <p>Compensation [DN]      Left ▼</p> <p>Compensation radius [D]      <input type="text"/></p> <p>Contouring [DNC]      Default ▼</p> <p>Reduce the profile [DNR]      <input type="checkbox"/></p> <p>Step by step compensation start      <input type="checkbox"/></p> <p>Step by step compensation end      <input type="checkbox"/></p> <p>Start compensation from setup [      Default ▼</p> </div>	<p>These parameters aren't assigned in case of blade set up.</p> <p>Assignment parameters of tool correction: here all possible options are provided.</p> <ul style="list-style-type: none"> <li>• <b>Correction</b> (ID=40; Whole): 0=Off; 1=Left side; 2= Right side. If it is required not to apply correction when it is made with piece-matrix: assign ID=8134 (for instance: see BLADE SETUP)</li> <li>• <b>Radius of compensation</b> (ID=36; Double): to set if different from tool radius. According to TpaCAD Configuration: the parameter can assign a variation of the compensation to be applied to the tool radius;</li> <li>• <b>Contouring</b> (ID=39; Whole): 1=inserts fillets in compensation; 2= brings fillets to intersections; otherwise (not available or different value) it doesn't affect general setting (default)</li> <li>• <b>Reduces profile</b> (ID=45; Whole): 1=it does not apply reductions; 2= applies reductions, if necessary; otherwise (not available or different value) it doesn't affect general setting (default);</li> <li>• <b>Step-by-step compensation start up</b> (ID=8135; Whole): it requires step-by-step start up if 1 (or anyway different from 0)</li> <li>• <b>Step-by-step compensation exit</b> (ID=8136; Whole): it requires step-by-step exit if 1 (or anyway different from 0)</li> <li>• <b>Starts compensation from set up</b> (ID=38; Whole ): 1=applies compensation; 2=adds a linear segment from the programmed set up on the starting point of compensation; otherwise (not available or different value) it doesn't affect on general setting (default).</li> </ul>
<div style="border: 1px solid black; padding: 5px;"> <p> <b>Advanced technology data</b></p> <p>Compensation [DN]      Left ▼</p> <p>Compensation radius [D]      <input type="text"/></p> </div>	<p>In the figure at the left an example of minimum assignment is shown, in which it is possible to enable compensation (Off/ Left/ Right) and set the compensation radius.</p> <p>In case of oriented set up, it is possible to configure the parameter</p> <ul style="list-style-type: none"> <li>• <b>Correction on Beta</b> (ID=41; Whole):             <ul style="list-style-type: none"> <li>• 1=with programmed tool or deduced/got (automatic or default tool) and without direct programming of compensation radius, applies compensation with interpolation along the slewing axis</li> <li>• =0, or tool selected by diameter, or with direct programming of compensation radius: no action.</li> </ul> </li> </ul>

Entry and exit segments	
Incoming line [INEN]	Managed
Typology [INTP]	Line
Length/Radius [INLL]	50
Path A (*) [INA]	
Starting Z (Ps) [INZ]	
Speed movement [INF]	
Outgoing line [OUEN]	Managed
Typology [OUTP]	Line
Length/Radius [OULL]	
Path A (*) [OUA]	
Final Z (Pe) [OUZ]	
Speed movement [OUTF]	

Setting parameters of enter/exit segments to profiles:

- **Enter segment** (ID=8180; Whole): enables if 1 (different from 0);
- **Typology** (ID=8181; Whole): 0= linear segment; 1= arc that arrives at the left of the profile; 2= arc that arrives at the right of the profile, 3=arc generally in the space
- **Length/Radius** (ID=8182; Double): length of linear segment or of arc radius
- **A path** (ID=8183; Double)
- **Z starting** (ID=8184; Double): starting depth of the segment
- **Movement speed** (ID=8190; Double): execution speed of the segment

- **Exit segment** (ID=8185; Whole): enables if 1 (different from 0);
- **Typology** (ID=8186; Whole): 0= linear segment; 1= arc that starts at the left of the profile; 2= arc that starts at the right of the profile, 3=arc generally in the space;
- **Length/Radius** (ID=8187; Double): length of linear segment or of arc radius
- **A path** (ID=8188; Double)
- **Z final** (ID=8189; Double): final depth of the segment
- **Movement speed** (ID=8191; Double): execution speed of the segment

## Custom parameters

A custom working must normally transfer custom information in piece-matrix. In this case:

- assign a typology of custom parameter, between 9001 e 9999:
- for the parameter: assign a saving column in piece-matrix among the free ones.

## Custom logical codes

A custom logical code interprets neither geometry nor technology. This doesn't prevent from configuring parameters that assign for instance a geometry. Anyway this doesn't activate any automatic interpretation.

For example it is possible to assign for the working the parameters referring to the application point of a point working, but:

- it isn't possible to apply a programming in relative.
- coordinates aren't a reference point for positioning upstream
- no graphic representation is associated to the working.

In program execution a custom logical working activates a custom procedure that doesn't match a working execution programmed on a piece.

Possible examples are:

- limits setting (rails and/or pods)
- measuring on the piece (for instance: with probes or set point procedures with piece movement)
- displacements in rapid
- programmed stop
- procedure of pieces blocking/release.

## Direct logical conditioning

A point or logical custom working can directly assign a logical conditioning of one only term, with a formalism translatable as:

**IF (e1 condiz1 e2) { }**

The working will be executed only if the condition is verified as TRUE.

IF (...) ? (...) ? (...)		<ul style="list-style-type: none"> <li>• <b>(e1)</b> (ID=8070; Double): first term of comparison</li> <li>• <b>?</b> (ID=8060; Whole): comparison condition:            &lt; has value 0; &lt;= has value 1            &gt; has value &gt;= has value            = has value &lt;&gt; has value</li> <li>• <b>(e2)</b> (ID=8073; Double): second term of comparison</li> </ul>
(e1) [ESP1]	r2	
? [TST1]	=	
(e2) [ESP2]	0	

This condition is anyway verified as TRUE if:

- the parameters themselves aren't configured for the working;
- the parameters are configured but they don't assign any condition.

However direct logical conditioning is verified, the possible exclusion of the working can occur only on application of logical conditions.

## 17.2 Profile codes

A profile code executes one or more interpolation segments, whose geometry is determined in accordance with assigned settings during use.

TpaCAD assigns profile codes with a distinction in two typologies: linear- and arc-.

The contextual helps of each profile codes provide a detailed description of geometries that each code calculates: refer to these.

It often occurs that the number of profile codes is wanted to be reduced. These codes are seen by the final user in the working palette of TpaCAD. To do this, the list of workings of each button in TpaCAD must be customised, as a normal procedure (see TpaCAD Configuration manual). The workings status mustn't be directly modified in the database.

A working excluded from display in graphic palette is in fact available for development of complex codes that can utilize it. If the working is excluded in the working database, an error situation occurs, for instance if the working is used in a basic macro.

## 17.3 Logical instructions

Logical instructions are particular simple workings to which no execution in machine corresponds.

Logical instructions are always available:

- IF..ELSEIF...ELSE..ENDIF structure
- ERROR instruction
- EXIT instruction
- application points of subroutine
- assignment of <j> variables
- global functions.

Instructions that are available only in macro text must be added to these:

- FOR..ENDFOR structure
- CONTINUE, BREAK instructions
- assignment of variables <\$>.

Each logical instruction has a specific interpretation, that can't be modified.

Logical instructions are assigned in the base database and mustn't be modified.

## 17.4 Global functions

Another question are the global Functions: in fact, it is normal that an application requires to assign one or more instructions of this group on the custom database, for the use of custom functions.

A global Function can assign directly a logical conditioning up to 3 terms, with the formalism that can be translated as already valid for an IF instruction.

The working will be executed only if the condition/s is verified as TRUE.

### Function identification

The name/number of the function is assigned by parameter: PRSETMCRNAM [ID=8098], that must be configured with:

- format: string
- visibility: hidden with default value assigned.

Parameter PRSETMCRNAM must assign the complete name of the custom function (Example: "funzmirror").

### Assignment of function call arguments

Arguments passed to the function are set using parameters from PRSETMCRVAR [8500] to PRSETMCRVAR+29 [8529], by maximum 30 arguments and only in correspondence with real arguments of the function:

PRSETMCRVAR assigns first argument (arg1)

PRSETMCRVAR+1 assigns second argument (arg2)

..

PRSETMCRVAR+29 assigns 30th argument (arg30).

Parameters typology (double or integer) must be assigned according to the meaning of each single custom function argument.

<table border="1"> <tr><td>arg..</td><td></td></tr> <tr><td>x</td><td></td></tr> <tr><td>y</td><td></td></tr> <tr><td>x1</td><td></td></tr> <tr><td>y1</td><td></td></tr> <tr><td>x2</td><td></td></tr> <tr><td>y2</td><td></td></tr> <tr><td>(retn)</td><td></td></tr> </table>	arg..		x		y		x1		y1		x2		y2		(retn)		<p>Let's resume for example funzmirror function (it is studied in TpaCAD Configuration manual):</p> <ul style="list-style-type: none"> <li>• x: (ID=8500: double)</li> <li>• y: (ID=8501: double)</li> <li>• x1: (ID=8502: double)</li> <li>• y1: (ID=8503: double)</li> <li>• x2: (ID=8504: double)</li> <li>• y2: (ID=8505: double)</li> </ul>
arg..																	
x																	
y																	
x1																	
y1																	
x2																	
y2																	
(retn)																	

It is recommended to configure the only parameters that are necessary for the prototype declared for the function. For example:

- for a function that reads 5 arguments: configure the parameters from PRSETMCRVAR to PRSETMCRVAR+4
- for a function that reads 10 arguments: configure the parameters from PRSETMCRVAR to PRSETMCRVAR+9
- for a function that reads a variable number of arguments, for a no-defined maximum: configure parameters from PRSETMCRVAR to PRSETMCRVAR+29.

### Assignment of function return elements

Return elements that are processed by the custom function are assigned to the same number of <j> variables using parameters from PRSETMCRNI [8800] to PRSETMCRNI +29 [8829], by maximum 30 elements:

PRSETMCRNI: assigns the index of jnn variable that gets the first return element

PRSETMCRNI +1: assigns the index of jnn variable that gets the second return element

..

PRSETMCRNI +29: assigns the index of jnn variable that gets the 30th return element.

The value returned by the function can on the contrary be assigned to one of the jnn variables using parameter: PRN\_JNI [8980].

Parameters typology is: integer.

<table border="1"> <tr><td>ret..</td><td></td></tr> <tr><td>funzmirror =&gt;j..</td><td>69</td></tr> <tr><td>xm =&gt;j..</td><td>70</td></tr> <tr><td>ym =&gt;j..</td><td>71</td></tr> </table>	ret..		funzmirror =>j..	69	xm =>j..	70	ym =>j..	71	<p>Let's consider funzmirror function as an example:</p> <ul style="list-style-type: none"> <li>• <b>funzmirror =&gt;j..</b>: (ID=8980) assigns index of &lt;j&gt; variable that sets the return (result) of the function (here: j69);</li> <li>• <b>xm =&gt;j..</b>: (ID=8800) assigns index of &lt;j&gt; variable that sets transform coordinate x (here: j70);</li> <li>• <b>ym =&gt;j..</b>: (ID=8801): assigns index of &lt;j&gt; variable that sets transform coordinate y (here: j71).</li> </ul>
ret..									
funzmirror =>j..	69								
xm =>j..	70								
ym =>j..	71								

It is recommended to configure the only parameters that are necessary for the prototype declared for the function. For example:

- if the function processes 5 return elements: set parameters from PRSETMCRNI to PRSETMCRNI +4
- if the function processes 10 return elements: set parameters from PRSETMCRNI to PRSETMCRNI +9
- ..
- if the function processes 30 return elements: set parameters from PRSETMCRNI to PRSETMCRNI +29.

It is possible to assign a maximum number of jnn variables up to the real number of elements that the function really processes.

## 17.5 Complex codes

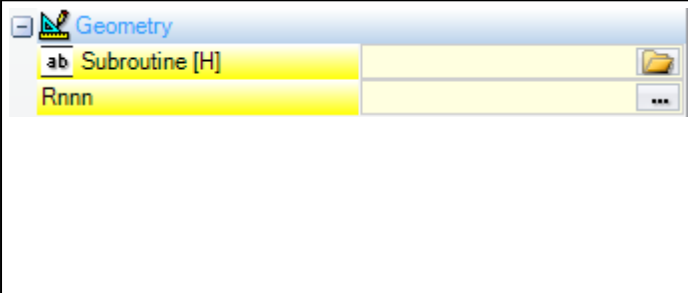
The use of a complex code allows to recall and execute a working scheme defined in another program, subroutine or macro-program.

The complex codes that can be configured are the following:

- **[2010 - 2014], [2030 - 2049]:** complex codes of generic type (codes: SUBn, STOOL). These codes are available to specialize a generic call of subroutine in a different way and they are assigned in the base database. As this codes are configured in a complex way, it is recommended not to modify them. Please, remember that normal use of this code can't recall a macro-program but: this functioning can be activated from TpaCAD configuration, but its use must be only for debug.  
These codes are used, too, to configure STOOL workings: in this case a subroutine call isn't executed, but the workings to be applied in the complex code are chosen among the first ones programmed and they are called by name.
- **[1001 - 2000], [3001 - 4800]:** codes of direct calls. These are codes available to configure subroutine or macro-programs calls with a formalism that is assigned according to the specific meaning that the call has.

In the base database different codes of this type are assigned: in some cases these are complex workings, that can be used as a basis to create custom codes.

For new codes use the interval of values reserved to custom applications.

	<p>Complex codes of SUBn type call the subroutine that is indicated in working insertion.</p> <p>As the subroutine to be applied isn't shown, it isn't possible to define previously assignment of &lt;r&gt; variables to carry out: that is why the codes (SUBn) doesn't configure parameters that are interpreted for r variables of subroutine. A complex code of generic type implements a generic call of function- or subroutine-type.</p>
---	--

The direct calls codes call, on the contrary, the subroutine or macro specified in the definition of the code itself. The final user applies the working ignoring the subroutine or macro to which it refers. A code of direct call configures what, according to computer terms, can be shown as a call to macro, considered as a close and self-consistent procedure,

A direct call code applies a wholly defined block, that can be customized only in the application parameters that are configured in the workings database, just as a generic point or set up code.

### Direct logical conditioning

A complex code can assign directly a logical conditioning up to 3 terms, with the formalism that can be translated as already seen for an IF instruction.

The working will be executed only if the condition is verified as TRUE.

This condition is anyway verified as TRUE if:

- the parameters themselves aren't configured for the working;
- the parameters are configured but they don't assign any condition.

If direct logical conditioning isn't verified as TRUE, the complex code doesn't interpret any application of sub-routine and macro, including possible induced calls. It is only about given assignment of positioning.

### How the subroutine and the macro are identified

The subroutine (or macro) name is assigned with parameter. PRSETMCRNAM [ID=8098], that must be configured with:

- **typology**: string
- **visibility**: visible and editable (for SUBn codes) or hidden with assigned default value (for direct calls codes).

Let's see working "FITTING X":

ID	Nome ASCII	Stato	Formato	Valore Min	Valore Max	\$ di default
▶ PRSETMCRNAM [8098]	H	Nascosto	Stringa	0.000	0.000	repeatx.tmcrc

Default string="repeatx.tmcrc" matches the path of addressing for a file with macro extension (tmcrc), that is: tpcadcfg\mcr\.

In case there are programs folders (product) and TpaCad (tpcadcfg) configuration folders both resulting from Albatros folder, let's remember that:

- macro-programs extension mustbe TMCR;
- custom macro-programs MUST be recorded starting from: tpcadcfg\custom\mrc;
- don't assign TMCR extension to subroutines or programs;
- record subroutines starting from folder product\sub.

Let's see some assignment examples of subroutine or macro-program:

to recall a	stored as	assign default string to	matches...
base database macro-program	tpcadcfg\mcr \mcr1.tmcrc	mcr1.tmcrc	addressing path related to a base macro-program
custom macro-program	tpcadcfg\custom\mcr \mio\mcr1.tmcrc	..\custom\mio \mcr1.tmcrc	addressing path for a macro-program in custom folder
subroutine	product\sub\ante\anta1 (door leaves\door leave1)  product\ante\anta1	ante\anta1 (door leaves\door leave 1)  ..\ante\anta1 (\door leaves \door leave1)	addressing path related to a subroutine file

A subroutine and macro-program path has a valid format, if it respects following criteria:

- maximum length of the name 256 characters
- maximum length of the extension 256 characters
- characters recognized as invalid for assignment of name or extension: " #%;/\\" (space, hash key, percentage, point and comma, bars)
- recognized characters as separators for directory: \' or \'
- manages Universal Naming Convention (UNC).

### How <r> variables of subroutine and macro are assigned

Parameters must be configured only for direct calls codes

Let's examine "FITTING X" working: the working has attributer **variables** assigned as single Edit, to show that <r> variables of the macro are known and inserted in parameters list of the working.

A code of direct call must configure **all and only parameters matching public variables** in the original text of the subroutine. Interpreted parameters have typology:

- PRSETMCRVAR [8500] for r0 variable assignment
- PRSETMCRVAR +1 [8501] for r1 variable assignment
- ..
- PRSETMCRVAR + 299 = PRSETMCRVARN [8799] for r299 variable assignment.

Parameters matching no-public r-variables are ignored.

Let's see "FITTING X" among working parameters:

ID	ASCII Name	Status	Format	Min value	Max value	Default \$	Dimension	Transforms	Description
8516	TD	Edit	Double	0.000	0.000		None	tecno=fitol	!3024Diameter
8525	TMC	Edit	Integer	0.000	0.000		None	tecno=nmac	!3021Machine
8526	TR	Edit	Integer	0.000	0.000		None	tecno=ngrp	!3022Group
8515	TP	Edit	Integer	0.000	0.000	1	None	tecno=tiptool	!3026Tool typology

- parameter [8516] matches r16 public variable and sets drilling diameter
  - parameter [8525] matches r25 public variable and sets the machine number
  - ..
- for all macro public variables.

during the application of the complex code it is possible to let empty the field corresponding to a public variable. In this case one of following situations occurs:

- null value is applied (0 for numerical variable, "" for string variable)
- the assigned value is applied for the variable in the subroutine text.

Its real behaviour depends on entry **Inherits <r> variables in Macro/Sub application** in TpaCAD Configuration:

- the first described situation matches the case of not selected entry
- the second situation matches the case of selected entry


Anyway it is possible to force selection for a single complex code, assigning entry "readrsub=n" in working attribute **Options in insertion** :

- to apply null value to not set variables: assign "readrsub=0";
- to apply value as written in the subroutine: assign "readrsub=1".

Please, remember that in the subroutine and macro text it is possible to verify if a variable has actually been assigned and to operate appropriate choices. Let's see about this use of prempy[..] parametric programming function.

For new applications of TpaCAD, that don't replace previous editors (Edicad, for example) or that anyway mustn't recover previously recorded programs, it is better not to activate the selection **Inherits <r> variables in application of Macro/Sub**: in this way the behaviour of all parameters is maintained for the subroutine variables. Functioning selection, already operative in TpaEdi32, is maintained for processing compatibility of programs that are written with previous application programs, as Edicad. Edicad always inherits the assigned value for the variables in the text of the subroutine (or macro).

### Subroutine placement mode

<p>Relative [EG] <input checked="" type="checkbox"/></p> <p>Rel&lt;- [EG ] <input checked="" type="checkbox"/></p> <p>X1 [X] </p> <p>Y1 [Y]</p> <p>Z1 [Z]</p> <p>Relative; ID=8015; Whole  X1: ID=8020; Double  Y1: ID=8021; Double  Z1: ID=8022; Double</p>	<p>The point of application is programmed in a system of <b>Cartesian coordinates</b> , where it is possible to assign the coordinates in absolute or relative mode. Placement is located in XY plane of face with depth to Z direction, perpendicularly to face plane: Calculated values for the three coordinates (x, y, z) define the <b>application point</b> (point that is shown as P1).</p>
<p>Rel&lt;- ID=8014; Whole</p>	<p>If relative mode is active and working is preceded by another complex code (macro or SUB working) Rel &lt;-.field is also evaluated.</p> <ul style="list-style-type: none"> <li>• If selected, point of application P1 is considered as relative to point of application (P1) of previous working.</li> <li>• otherwise: point of application P1 is considered relative to the last working point calculated for the previous working.</li> </ul>

For not assigned coordinates of point P1 (empty field) it is possible to have two different placement modes:

**To** translation isn't applied with respect to the position as resulting from the original development of the subroutine or:



**B** propagation of the coordinate as the first one assigned is applied.

Selection of functioning mode occurs according to/through entry **Applies P(x,y,z) of subroutine as for point Workings** TpaCAD Configuration:

- case (A) matches the case of not selected entry
- case (B) matches the case of selected entry

Anyway it is possible to force selection for a single complex code, assigning entry "geoxyz = n" in working attribute **Insertion options** :

- "geoxyz=0", to force case (A);
- "geoxyz=1", to force case (B).

Also for selection of functioning **Applies P(x,y,z) for subroutines as well as for point workings** , operating in TpaEdi32, it can be necessary to consider compatibility, in particular for programs recorded with Edicad. Edicad never applies propagation of coordination assigned upstream: if the coordinate isn't assigned, placement of subroutine (or macro) is determined only by its internal development. For new application of TpaCAD, that don't replace previous editors (Edicad, for example) or that anyway mustn't recover previously recorded programs, it is better not to activate the selection. This way the application of the application point of complex codes is maintained uniform to the one of other working typologies.

On the above shown P1 point (application point) it is translated:

- a significant point of the subroutine overall rectangle, if parameter (ID=8132) has value >0 (the rectangle is significant in x and y):

<table border="1"> <tr><td>Relative [EG]</td><td><input type="checkbox"/></td></tr> <tr><td>Rel&lt;- [EGl]</td><td><input type="checkbox"/></td></tr> <tr><td>X1 [X]</td><td>If/2</td></tr> <tr><td>Y1 [Y]</td><td>200</td></tr> <tr><td>Z1 [Z]</td><td>-5</td></tr> <tr><td>Locate the extents rectangle</td><td>Do not apply</td></tr> <tr><td></td><td>Do not apply</td></tr> <tr><td></td><td>Centre in XY</td></tr> <tr><td></td><td>X- Y-</td></tr> <tr><td></td><td>X- Y+</td></tr> <tr><td></td><td>X+ Y-</td></tr> <tr><td></td><td>X+ Y+</td></tr> </table>	Relative [EG]	<input type="checkbox"/>	Rel<- [EGl]	<input type="checkbox"/>	X1 [X]	If/2	Y1 [Y]	200	Z1 [Z]	-5	Locate the extents rectangle	Do not apply		Do not apply		Centre in XY		X- Y-		X- Y+		X+ Y-		X+ Y+	<p>this possibility is in assignment window of SUB application code of subroutine at entry <b>Places/ positions overall rectangle</b> (ID=8132; Whole). This is a multiple selection field:</p> <ul style="list-style-type: none"> <li>• 0 (It doesn't apply): it doesn't affect positioning;</li> <li>• 1 (Centre in XY): the overall rectangle centre of the subroutine is taken to P1;</li> <li>• 2 (X- Y-): in P1 the minimum overall point is taken both to x and to y;</li> <li>• 3 (X- Y+): in P1 the overall point is taken minimum to x and greater to y;</li> <li>• 4 (X+ Y-): in P1 the overall point is taken greater to x and lesser to y;</li> <li>• 5(X+ Y+): in P1 the overall point is taken greater both to x and to y;</li> <li>• &gt; 5: as in case 1;</li> </ul>
Relative [EG]	<input type="checkbox"/>																								
Rel<- [EGl]	<input type="checkbox"/>																								
X1 [X]	If/2																								
Y1 [Y]	200																								
Z1 [Z]	-5																								
Locate the extents rectangle	Do not apply																								
	Do not apply																								
	Centre in XY																								
	X- Y-																								
	X- Y+																								
	X+ Y-																								
	X+ Y+																								

- otherwise: the application point (code: COPMCRPNT), if assigned in the subroutine text;
- otherwise: the first working point of the subroutine.

Note that parameters of application point aren't always assigned in complex workings (all or some of them/partially or wholly) and also parameters of relative programming: everything depends on what the subroutine must execute.

<table border="1"> <tr><td></td><td>Geometry</td></tr> <tr><td>Starting X (Ps) [X]</td><td></td></tr> <tr><td>Final X (Pe) [XF]</td><td></td></tr> <tr><td>Step [ST]</td><td>32</td></tr> <tr><td>Qy [Y]</td><td></td></tr> <tr><td>Zp [Z]</td><td></td></tr> </table>		Geometry	Starting X (Ps) [X]		Final X (Pe) [XF]		Step [ST]	32	Qy [Y]		Zp [Z]		<p>When assigned, moreover, parameters of application point have a description relevant to developed geometry. Let's take, for instance, working FITTING X":</p> <ul style="list-style-type: none"> <li>• parameter ID=8020 matches starting X field</li> <li>• parameter ID=8021 matches Qy field</li> <li>• parameter ID=8022 matches Zp field</li> </ul>
	Geometry												
Starting X (Ps) [X]													
Final X (Pe) [XF]													
Step [ST]	32												
Qy [Y]													
Zp [Z]													

### Point hook

Let's open working [2010]SUB:

<table border="1"> <tr><td>Point hook [EGL]</td><td><input type="checkbox"/></td></tr> <tr><td>(ID=8101; Whole):</td><td></td></tr> </table>	Point hook [EGL]	<input type="checkbox"/>	(ID=8101; Whole):		<p>For complex codes, too parameter of is managed <b>point hook</b>. If hook parameter is set enabled (value 1):</p>
Point hook [EGL]	<input type="checkbox"/>				
(ID=8101; Whole):					

	<ul style="list-style-type: none"> <li>• it prevails on placement for translation through point P1</li> <li>• it deletes management of placement point as for set ups, it takes for three coordinates (xyz) an application in relative with null displacements with respect to the final application point of previous working. So profile execution could go on.</li> </ul>
--	--

For fixed Cycles:configure the parameter if the complex code develops a profile that can be hooked.

**How a face to be applied is chosen**

The face of the subroutine to be applied is assigned using following parameters: PRSIDE [ID=6], PRSIDE1 [ID=8097]

- PRSIDE set and with a value calculated different form 0 and -1: applies the subroutine face with number = value PRSIDE(note (1)) + value PRSIDE1);
- otherwise interprets **induced calls (automatic)**: each face of the program applies the corresponding subroutine face(face 1 of the program calls face 1 of the subroutine, etc..). Some conditions must be verified (otherwise it applies the subroutine face):
  - point hook mustn't be required
  - the complex code must be applied in a piece with program typology (that is: not subroutine or macro)
  - the application level is the basic one (it isn't a secondary call of complex code, in an expanded level)
  - a complex code of text development isn't recognized (see further):
 If the subroutine is programmed in piece-face, other evaluations (?) to be done are:
  - first of all: the subroutine face that is applied is now the same as the one assigned in the property "Subroutine application face";
  - automatic induced calls aren't applied if subroutine development calculates a COPNEWSIDE or COPSUBSIDE code;
  - induced calls aren't applied if the complex code is applied to an automatic face.

Note (1): if a custom numbering of the real piece faces is assigned and if PRSIDE parameter is editable. PRSIDE value is interpreted as custom face number. Otherwise: PRSIDE value is interpreted directly as a face internal identification number.

Development of automatic induced calls applies following rules:

- are applied also for fictive faces but not in piece-face
- are applied at the end to workings of referring faces
- are program lines that can't be modified directly: these are inserted and/or deleted and/or modified automatically, modifying the main call (otherwise: **master**). In TpaCAD: program lines added for induced calls are maintained/kept hidden.
- they respect logical conditioning of main calls
- a possible selection of quotes placement is reset in relative
- if the setting of the main call uses <j> variables (for example: to assign Y coordinate Y of P1): the same values are used to resolve the settings and the development of induced calls.

Let's see some examples set in the base database:

	parameter PRSIDE	
X FITTING	hidden; default value=1	The parameter is visible and chooses which face must be developed in the macro-program: here, face 1. The code excludes induced calls.
X REPEAT	hidden; default value=2	The parameter isn't visible and chooses which face must be developed in the macro-program: here, face 2. The code excludes induced calls.
SUB	editable	The parameter is visible and the operator chooses the face of the subroutine to be applied (if it lets the field empty: it activates the mechanism of induced calls)

In induced calls development further assignments are available.

### Selection of induced faces

Induced call application can be selective: Let's open working [2010]SUB:

ab Induced faces [SON]	3,5	<ul style="list-style-type: none"> <li>Induced faces:(ID=8111): if set, it indicates faces involved in induced call. In the figure: "3;5" setting indicates application of induced calls in faces 3 and 5 <u>only</u></li> <li>Excluded faces:(ID=8112: string) if set, it indicates faces not involved in induced call.</li> </ul>
ab Excluded faces [SOFF]		

In both fields: list the numbers of the faces split up by ";" (semicolon).

Setting of *Induced faces* it prevails on that of *Excluded faces*.

For the parameters it is also possible to set a numerical typology (whole). In this case: it is possible to show a single face.

If a custom numbering of the real piece faces is assigned: also the values that are set here are interpreted as custom numbering of the faces.

### Placement mode of induced calls

Let's open working [2010]SUB.

XY Induced [SXY]	Default	<p>XY induced (ID=8110; Whole): it allows to choose among different assignment modes the application point P1 in induced calls. This is a multiple selection field:</p> <ul style="list-style-type: none"> <li>0 (Default): the field does not affect positioning (it applies mode assigned in TpaCAD configuration);</li> <li>1 (Adapt XY): it adapts the point of application</li> <li>2 (Go through XY=): for each induced call, it forwards fields as set up in master call</li> <li>3 (Do not go through XY): for each induced call, it forwards non-set up fields.</li> </ul>
	<ul style="list-style-type: none"> <li>Default</li> <li>Adapt XY</li> <li>Forward XY =</li> <li>Do not forward XY</li> </ul>	

For a detailed description of the options, please, see the chapter referring to TpaCAD Configuration.

In application of induced call a propagation starting from upstream call application is never applied, even if applied in master call.

### Application of geometric transforms

When a subroutine or macro-program is applied, some geometric transforms can be activated, applied in the order below.

#### Inversion

Let's open working [2010]SUB.

Invert [EINV]	<input type="checkbox"/>	<ul style="list-style-type: none"> <li>(ID=8100: whole): if enabled, it involves the inversion of the execution order for the developed workings: the last block becomes the first one and s.o.</li> </ul>
---------------	--------------------------	--

It is executed as first transform and deletes management of placement point.

The transform also reverses the settings of tool correction (right or left ) for each set up.

The parameter mustn't be configured if complex code:

- it applies on his turn a complex code for which inversion isn't allowed.
- or it has inversion limits itself.

#### Rotation

subroutine rotation can be set configuring **one** of the parameters shown here following (the search comes out when it finds a parameter, even if not set in the reported sequence):

Rotation angle (°): [A]		PRANG [ID=8044; double]:
-------------------------	--	--------------------------

it programs rotation angle with a value in degrees and tenths of degree;

- PRANG30 [ID=8032; whole]: rotation angle (in degrees) is calculated = integer of parameter \*30

- PRANG30 [ID=8033; whole]: rotation angle (in degrees) is calculated = integer of parameter \*45
- PRANG90 [ID=8034; whole]: rotation angle (in degrees) is calculated = integer of parameter \*90

**Subroutine rotation**

- It is set with rotation angle programmed (in degrees and decimal degrees) in face XY plane from X axis.
- it occurs around sub-program point of application.
- it isn't applied if development of subroutine implies execution of arcs in a not-xy-plane and if working of operative code COPA10 (arc for 3 points in plane xyz) doesn't result assigned.

In the base database only parameter PRANG [ID=8044] is configured: the use of remaining parameters is to be limited to the case of complex codes of particular use, as constraint placements (pod benches) that don't need application of particular profile tools among the ones in TpaCAD.

The parameter mustn't be configured if the complex code:

- applies on his turn a complex code for which rotation isn't allowed
- or it has inversion limits itself.

**Mirror**

Let's open working [2010]SUB:

Horizontal Mirror [EMX]	<input type="checkbox"/>	<ul style="list-style-type: none"> <li>• Horizontal mirror (ID=8103; whole): symmetry execution around parallel axis of Y (face) axis (Cartesian representation: vertical axis)</li> <li>• Vertical mirror (ID=8104; whole): symmetry around parallel axis of X (face) axis (Cartesiann representation: horizontal axis)</li> </ul>
Vertical Mirror [EMY]	<input type="checkbox"/>	

If both items are selected, options are summed up.

Alternatively, to both parameters it is possible to configure parameter:

- PRMIRROR [ID=8106; whole]
  - 0 doesn't execute symmetries
  - 1: symmetry around parallel axis of y (face) axis
  - 2: symmetry around parallel axis of X (face) axis
  - 3: symmetry combined around horizontal + vertical axis

or:

- PRMIRRORXY [ID=8105; whole]: symmetry combined around horizontal + vertical axis.

In base database are configured only parameters: [ID=8103], [ID=8104].

For both parameters a is set activeExchange function of the message in list (in assignment of parameter attribute Auxiliaries), so that the descriptive message is adapted to the graphic representation of the face.

With application of x or y mirror (note:not x+y): the transform reverses also the settings of tool compensation (right or left) of each set up.

The parameter mustn't be configured if the complex code

- applies on his turn a complex code for which rotation isn't allowed
- or it has inversion limits itself.

**Scale**

Let's open working [2010]SUB.

<input type="checkbox"/> Scale factor		<ul style="list-style-type: none"> <li>• Enable (ID=8118; whole): if selected, it enables transform application;</li> <li>• Factor (ID=8119; double): reduction or amplification factor (minimum programmable: 0.001). Following situations are interpreted:                     <ul style="list-style-type: none"> <li>• less than 1: reduction applied</li> <li>• higher than 1: amplification applied</li> <li>• =1: no action.</li> </ul> </li> <li>• 3d scale (ID=8120; whole), it also</li> </ul>
<input type="checkbox"/> Enable [EFAT]	<input type="checkbox"/>	
<input type="checkbox"/> Enable [FAT]		
<input type="checkbox"/> [ZFAT]	<input type="checkbox"/>	

	enables in-depth application (face Z axis). Selection is compulsory if the subroutine also runs arcs assigned on a plane different from xy.
--	---

The transform applies a reduction or amplification factor to the subroutine.

The parameter mustn't be configured if the complex code:

- applies on his turn a complex code for which scale application isn't allowed
- or it has scale limits itself.

### Repetitions in subroutine running

Complex codes manage two different modes of sub-program automatic repetition:

- Free distribution
- Distribution in matrix

In the base database, among complex codes of generic type:

- SUB code performs the multiple application with free repetition
- SMAT code performs the multiple application with matrix repetition.

It is possible to assign both modes (as preferred) in an only complex code.

### Repetitions with free distribution

<p>Let's open working [2010]SUB.</p> <table border="1"> <tr> <td colspan="2">Repetitions</td> </tr> <tr> <td>Repetitions [NN]</td> <td>2</td> </tr> <tr> <td>X-Offset [NX]</td> <td>100</td> </tr> <tr> <td>Y-Offset [NY]</td> <td>0</td> </tr> <tr> <td>Z-Offset [NZ]</td> <td></td> </tr> <tr> <td>Rel&lt; [NGO]</td> <td><input type="checkbox"/></td> </tr> <tr> <td>Point hook [NGLR]</td> <td><input type="checkbox"/></td> </tr> <tr> <td>Offset A (°) [NA]</td> <td></td> </tr> </table>	Repetitions		Repetitions [NN]	2	X-Offset [NX]	100	Y-Offset [NY]	0	Z-Offset [NZ]		Rel< [NGO]	<input type="checkbox"/>	Point hook [NGLR]	<input type="checkbox"/>	Offset A (°) [NA]		<ul style="list-style-type: none"> <li>• Repetitions (ID=8109; whole): number of repetitions from <u>add</u> to base application. The minimum (powerful?) value is 1.</li> <li>• Offset X, Y, Z (ID=8023, 8024, 8025; double): relative displacements applied to each repetition: the value are added at each repetition (with their sign);</li> <li>• Rel &lt; (ID=8107; whole): if selected applies offsets to application starting point of previous repetition;</li> <li>• Point hook (ID=8108; whole): if selected, it hooks each repetition to the previous one. In this case, it ignores settings concerning Offsets X, Y, Z and Rel-field &lt;;</li> <li>• Offset A(°) (ID=8045; double): it sets rotation increase by applying it to each following repetition. Applied starting value is given by rotation field in base application (parameter: PRANG). If for example base rotation executes a rotation by 30°:             <ul style="list-style-type: none"> <li>• if Offset A(°) isn't set: all repetitions rotate by 30°;</li> <li>• if Offset A(°)=10°: 1st repetition rotates by 40°, 2nd repetition rotates by 50°,...</li> </ul> </li> </ul>
Repetitions																	
Repetitions [NN]	2																
X-Offset [NX]	100																
Y-Offset [NY]	0																
Z-Offset [NZ]																	
Rel< [NGO]	<input type="checkbox"/>																
Point hook [NGLR]	<input type="checkbox"/>																
Offset A (°) [NA]																	

Any mirrored transform assigned for base application is also applied to repetitions. In particular, transforms are also applied to the corresponding offsets:

- Horizontal Mirror: it also mirrors the offset set along the horizontal axis
- Vertical Mirror: it also mirrors the offset set along the vertical axis.

Any scale and/or inversion transform assigned for base application is also applied to repetitions.

**Repetition with matrix distribution**

<p>Let's open working [2012]SMAT:</p> <table border="1"> <tr><td colspan="2">Repetitions</td></tr> <tr><td>Rows [NL]</td><td></td></tr> <tr><td>Columns [NC]</td><td></td></tr> <tr><td>Column spacing [NTX]</td><td></td></tr> <tr><td>Row spacing [NTY]</td><td></td></tr> <tr><td>Rel&lt;- [NGO]</td><td><input type="checkbox"/></td></tr> </table>	Repetitions		Rows [NL]		Columns [NC]		Column spacing [NTX]		Row spacing [NTY]		Rel<- [NGO]	<input type="checkbox"/>	<ul style="list-style-type: none"> <li>• (ID=8116; whole, hidden): default value=1 is activated matrix repetition mode;</li> <li>• Rows, Columns (ID=8113, 8114; whole): number of rows and columns of repetitions matrix. The minimum value to be enabled for repetitions is 1, in both fields. The total number of applications made is given by (Row * Columns) product, included base application.</li> <li>• Distance between rows (ID=8024; double): distance between the rows of the matrix;</li> <li>• Distance between rows (ID=8023; double): distance between the rows of the matrix;</li> <li>• Rel &lt;- (ID=8107; whole): if selected, it applies the rows and columns offsets to the application starting point of previous repetition.</li> </ul>
Repetitions													
Rows [NL]													
Columns [NC]													
Column spacing [NTX]													
Row spacing [NTY]													
Rel<- [NGO]	<input type="checkbox"/>												

NOTE: columns are to be considered always assigned in X axis direction and rows in Y axis direction, anyway face representation is oriented.

Please, note that for parameters Rows, Columns, Distance among rows, Distance among columns an Exchange function of the message in list is set active (in assignment of parameter attribute Auxiliaries), so that the descriptive message is adapted to the graphic representation of the face.

Any mirrored transform assigned for base application is also applied to repetitions. In particular, transforms are also applied to the corresponding offsets:

- Horizontal Mirror: it also mirrors the offset set along the horizontal axis
- Vertical Mirror: it also mirrors the offset set along the vertical axis.

Any scale and/or inversion transform assigned for base application is also applied to repetitions.

**Repetitions with distribution for choice**

As already told, it is possible to assign the choice of the repetition mode in an only one complex code. Let's open working [2014]STOOL:

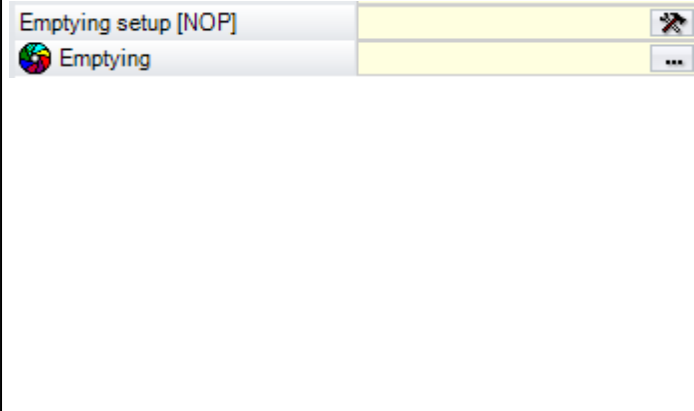
<table border="1"> <tr><td colspan="2">Repetitions</td></tr> <tr><td>Matrix repetition mode [NM]</td><td><input type="checkbox"/></td></tr> <tr><td>Keep repetitions only [RONLY]</td><td><input type="checkbox"/></td></tr> <tr><td>Rel&lt;- [NGO]</td><td><input type="checkbox"/></td></tr> <tr><td colspan="2">Repetitions [NN]</td></tr> <tr><td>X-Offset [NX]</td><td></td></tr> <tr><td>Y-Offset [NY]</td><td></td></tr> <tr><td>Z-Offset [NZ]</td><td></td></tr> <tr><td>Point hook [NGLR]</td><td><input type="checkbox"/></td></tr> <tr><td>Offset A (*) [NA]</td><td></td></tr> <tr><td colspan="2">Rows [NL]</td></tr> <tr><td>Columns [NC]</td><td></td></tr> <tr><td>Column spacing [NTX]</td><td></td></tr> <tr><td>Row spacing [NTY]</td><td></td></tr> </table>	Repetitions		Matrix repetition mode [NM]	<input type="checkbox"/>	Keep repetitions only [RONLY]	<input type="checkbox"/>	Rel<- [NGO]	<input type="checkbox"/>	Repetitions [NN]		X-Offset [NX]		Y-Offset [NY]		Z-Offset [NZ]		Point hook [NGLR]	<input type="checkbox"/>	Offset A (*) [NA]		Rows [NL]		Columns [NC]		Column spacing [NTX]		Row spacing [NTY]		<ul style="list-style-type: none"> <li>• Matrix repetitions (ID=8116; whole):</li> <li>• selected: activates matrix repetition mode</li> <li>• not selected: activates free repetition mode</li> <li>• Rel &lt;- (ID=8107; whole): is still valid for both repetition modes;</li> <li>• Grouped fields from the frame assign free repetition mode (as already seen);</li> </ul> <p><u>Change</u> parameters that assign the distance between rows and columns, in matrix repetition mode:</p> <ul style="list-style-type: none"> <li>• Distance between rows (ID=8027; double): distance between the rows of the matrix</li> <li>• Distance between columns (ID=8026; double): distance between columns of the matrix.</li> </ul> <p>These parameters typologies could be used also in SMAT code, in place of typologies (ID=8024, 8023). In this case: SMAT had not to configure both parameters couples.</p>
Repetitions																													
Matrix repetition mode [NM]	<input type="checkbox"/>																												
Keep repetitions only [RONLY]	<input type="checkbox"/>																												
Rel<- [NGO]	<input type="checkbox"/>																												
Repetitions [NN]																													
X-Offset [NX]																													
Y-Offset [NY]																													
Z-Offset [NZ]																													
Point hook [NGLR]	<input type="checkbox"/>																												
Offset A (*) [NA]																													
Rows [NL]																													
Columns [NC]																													
Column spacing [NTX]																													
Row spacing [NTY]																													


## Activation of emptying

A complex code can generate a cycle of emptying, applied to profiles that are developed by the subroutine.

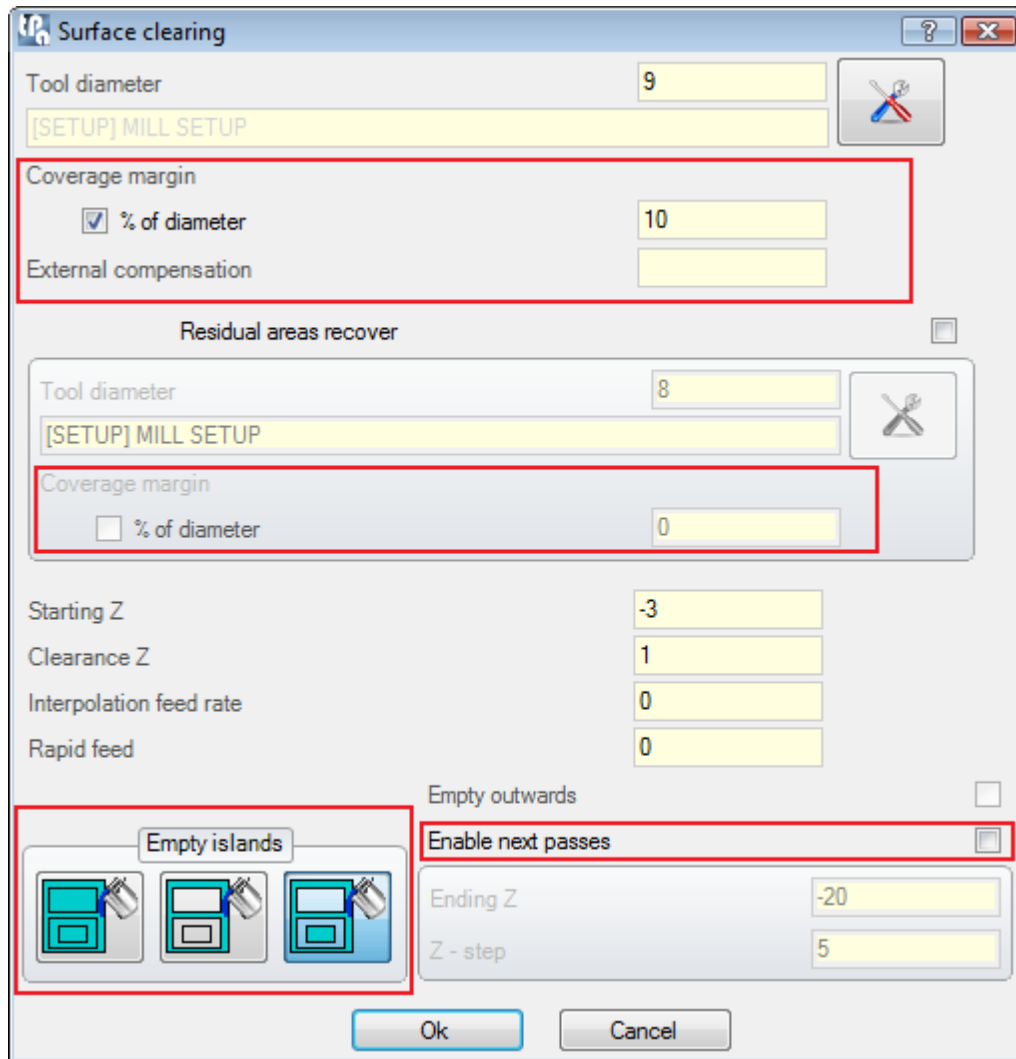
The procedure examines all expansions resulting from subroutine application and recognizes the profiles defining close areas. The criteria defined for the emptying tool are applied to these profiles.

Emptying codes are managed only in Professional mode and if the overall enabling of Emptying tool is activated.

<p>Let's open working[2013] SEMPTY:</p> 	<ul style="list-style-type: none"> <li>• Enable emptying (ID=8140; integer): if selected it enables emptying procedure to subroutine development;</li> <li>• Maintain only emptying (ID=8149; integer):             <ul style="list-style-type: none"> <li>• if selected delete subroutine original workings (<b>all</b>: not only those which have determined an emptying). In this case: an error is given when emptying profiles aren't generated</li> <li>• not selected (value 0) maintains also subroutine original workings</li> </ul> </li> <li>• Emptying set up (ID=8131; integer):             <ul style="list-style-type: none"> <li>• if set: assigns operative code of set up workings for generated profiles</li> <li>• otherwise (empty field): the procedure uses the set up code of default for TpaCAD application program</li> </ul> </li> </ul>
---	---

 **Emptying**: the item opens an assisted setting window which is similar to the case of application of emptying command. In the database all parameters assigning application criteria of emptying are available in parameters list. Simplification is only for working use. Let us see in detail:

Setting parameters of emptying criteria (areas in the frame)



#### Parameters of main emptying:

- Coverage margin (ID=8143; integer): accepts values between 10.0 and 90.0 (in % of main tool diameter); if not set: it assumes value 10;
- External compensation (ID=8168; integer): exit on main compensation (subtracts to tool radius). Accept a positive value  $\geq \text{epsilon} * 10$  and  $\leq \text{tool radius}$ . If it is assigned valid: the first correction is executed to the value of (tool radius - value). This way the correction can exit from the original profile (if the value is the same of the tool radius, it can follow the original profile). This functioning excludes the evaluation of residual areas recovering. The assignment is valid only for the generation of the main emptying profile and not for the islands management;

The first technology button sends back to the assignment of the emptying main tool.

#### Parameters of residual areas recovering management:

- Residual areas recovering (ID=8159; integer): if selected enables recovering of residual areas;
- Coverage margin (ID=8167; integer): accepts values between 10.0 and 90.0 (in % of secondary tool diameter); if not set: it takes value 10;

The second technology button sends back to the assignment of the residual areas emptying tool.

#### General parameters

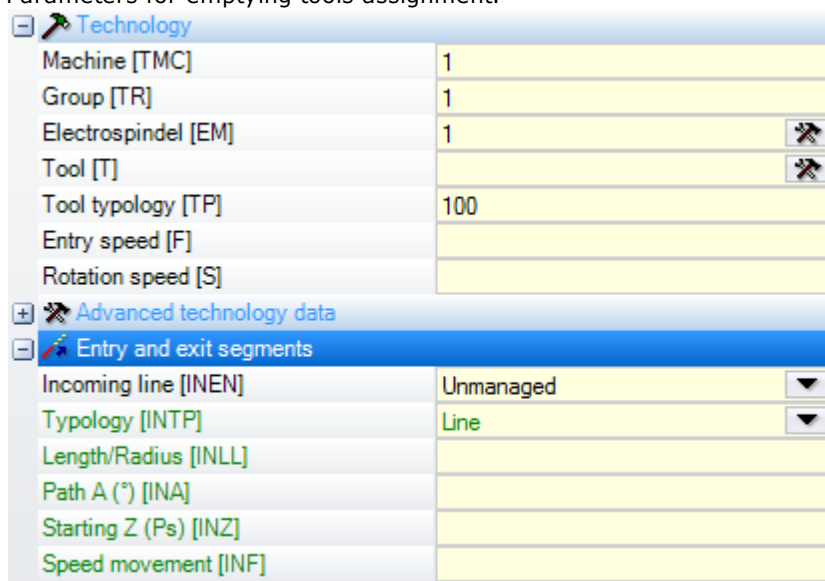
- Emptying of the islands (ID=8141; integer).
  - 0=ignore islands (default)
  - 1=empty external to the islands
  - 2=alternate emptying
- Empty outwards (ID=8142; integer):
  - 0 (not selected): develop emptying profiles inwards;
  - 1(selected): develop emptying profiles outwards (only if islands Emptying =0)

#### Parameters of technological settings of profile



- Enable next passages (ID=8148;integer):
  - 0 (not selected) execute an only one passage (at quote Z= initial Z)
  - 1 (selected) or if the parameter isn't assigned: it values end Z (ID=8145) for execution of next passages;
- end Z (ID=8145; double): end depth of emptying (depth of last passage). If set:
  - ignored if it has value equal to initial Z (less than epsilon);
  - if it has a sign different from initial Z: it determines an error;
  - if it has a value different from initial Z:
    - for each emptying more passages are executed (minimum 2);
    - the values of initial and final Z are exchanged if the initial depth exceeds the final one;
- Z feed (ID=8146; double): depth feed in next passages. The parameter is valued only if it must execute more passages:
  - ignored if not set or if with null value (less than epsilon). In this case: for each emptying two passages are executed: the first with initial Z and the second with end Z;
  - otherwise: set feed applied to each next passage until the end depth is reached.

Parameters for emptying tools assignment.



	Main tool	Secondary tool	Note
Machine	ID=8150; integer	ID=8160; integer	
Group	ID=8151; integer	ID=8161; integer	
Electric spindle	ID=8152; integer	ID=8162; integer	
Tool	ID=8153; integer	ID=8163; integer	
Tool type	ID=8154; integer	ID=8164; integer	
Tool entry speed	ID=8155: double	ID=8165: double	(1)
Rotation speed	ID=8156; integer	ID=8166; integer	(2)
Incoming line	ID=8174; integer	not managed:	(3)
Segment type	ID=8175;integer		
Segment length	ID=8176: double		
Followed angle	ID=8177: double		
Initial Z	ID=8178: double		
Segment speed	ID=8179: double		

Note(1): tool entry Speeds assign set up working parameter shown with parameter Technology: displacement speed (parameter attribute: Transforms)

Note(2): tool rotation Speeds assign set up working parameter shown with parameter Technology: rotation speed (parameter attribute: Transforms)

Note(3): assignment of segment entry at profile can be configured only for the main emptying profile and follows what has been written about set up working. Entry segment is always tangent and can be:

- linear
- arc on the left of the profile
- arc on the right of the profile
- arc in space development.

A diameter of value no less than (epsilon \*20.0) must correspond to the main tool: otherwise there is an error.

A diameter of value no less than (epsilon \*20.0) must correspond to the secondary tool (to recover residual areas). Anyway it must be less than the main tool diameter: otherwise no procedure of residual areas recovering is activated.

### Text generation

A complex code can generate profiles of text writing

Enabling modes are defined here following:

- the complex code applies a TrueType.tmc; macro
- the macro application assigns the workings:
  - (obligatory) a set up working (otherwise there is an error);
  - (optional) a COPL01 profile working.

I codes of text generation are managed in TpaCAD only if in Professional mode and with an active complex enabling of text Generation tool active.

Check of above mentioned points starts in automatic procedure of text generation.

Please, remember that the procedure of text generation:

- does not manage induced calls;
- doesn't resolve codes programming: COPNEWSIDE, COPSUBSIDE, complex codes (macro or subroutine).

Set up working of sub/macro is used to assign the set up of each generated profile with all set technological assignment (tool, correction, speed,...).

If it is necessary to assign a technology only to the profile segments (speed of interpolation): it is possible to assign it to linear typology working.

The macro application complex code must make available some significant parameters with assignment of the same number of public <r> variables.

This means that: the macro variables are assigned as parameters of the same complex code (parameters type: PRSETMCRVAR,..).

Let's see the <r> variables made available by the macro:

r..	Type	Meaning
r0:	<b>string:</b>	Text
r1:	<b>string:</b>	font name(TrueType)
r2	double	capital letters height
r3	double	distance between characters
r4	integer	italics flag
r5	integer	bold flag
r6	integer	enabling flag font test TrueType
r7	double	space character width
r8	Integer	System font (if = 0) or custom (if > 0)
r9	Integer	For custom font: value > 0 requires application of spline curve

Also technology parameters are normally assigned with <r> variables of subroutine that can be assigned

together with the ones of normal use, above mentioned.

The called macro-program can manage error situations with proper ERROR or BREAK codes:

- character height too small or exceeding,
- assigned technology for set up not valid,
- ...

as each normal macro text.

When technology elements are acquired (set up and in case profile workings): all workings resulting from the subroutine application are deleted and the application inserts only the profiles due to the generation of the given text.

Let's open working[1100] TEXT:

ab Text [TX]	
ab Font [FN]	
Font height [HC]	100
Font spacing [L]	0
Space width [SP]	20
Italic [IT]	<input type="checkbox"/>
Bold [BL]	<input type="checkbox"/>

- Text (ID=8500= PRSETMCRVAR; string): text string. If it is not assigned: no writing is generated;
- Font (ID=8501= PRSETMCRVAR+1; string): string. There are error situations if:
  - the name is assigned on more than 32 characters;
  - the name doesn't match a valid font;
  - the font generator couldn't assign a valid True Type font on the basis of the set name;
- Letters height (ID=8502=PRSETMCRVAR+2; double): capital letters height. If not set or if set with a value less than (epsilon\*50): uses (epsilon\*50)
- Distance between characters (ID=8503= PRSETMCRVAR+3; double)
- Space character width (ID=8507= PRSETMCRVAR+7; double)
- Italics (ID=8504= PRSETMCRVAR+4; whole): if selected requires generation of text in italics
- Bold (ID=8505= PRSETMCRVAR+5; whole): if selected requires generation of text in bold

The working doesn't configure parameter (ID=8506= PRSETMCRVAR+6; whole) above mentioned of *Enabling font text TrueType*: the value is used as assigned in the macro-program text:

- if 0: in font assignment it is possible to choose a font that isn't TrueType, if the font generator can assign a valid TrueType font, on the basis of the set name
- if 1 (different from 0): the font name needs to match a TrueType font.

The only parameters of text working [1100] that set the written generations functionalities have been examined here. Anyway here are also the parameters referring to;

- text positioning
- geometric transforms (profiles inversion, rotation, symmetries).

Using complex codes that generate texts, please, note that font availability is given by operative system and not from single application (TpaCAD or other) and can so change in different installations.

## Generation of text and emptying at the same time

Let's start with the situation examined before, that is the comprehensive enabling of text generation tool.

Applying of an emptying cycle to the generated text can also be considered: the parameters defining emptying modes must be added to the complex code.

In the base database for instance we can open the working: [1101] TEXT EMPTYING.

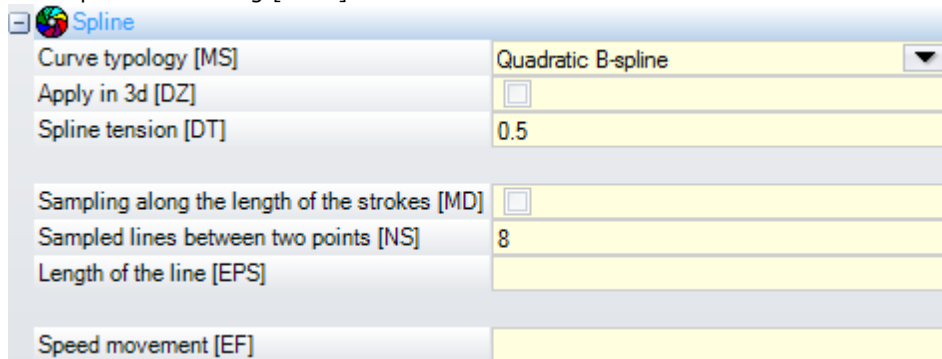
## Generation of spline curves

A complex code can generate B-Spline curves, applied to profiles that are developed by the subroutine. The codes of Spline curves generation are managed in TpaCAD only if in Professional mode and with an active comprehensive enabling of Spline generation tool.

It wasn't necessary to configure in the base database codes that develop B-Spline curves associated to

subroutine application. There is the working [1042] STOOL: SPLINE, operating as programmed tool (see further). At the moment we simply say that the subroutine applied and processed by the code is the working applied upstream.

Let's open the working [1042] STOOL: SPLINE



- **(ID=8170; whole, hidden): value of default=1** à enables procedure of B-Spline curves generation;
- **(ID=8171; whole, hidden): value of default=1** à deletes the original workings of the subroutine (**all**: not only those that have determined a Spline). There is an error in case curves haven't been generated.

value =0 or not configured parameter: it would maintain original workings;

- Curve typology (ID=8172; whole):
  - 0=Quadratic B-spline
  - 1=Cubic B-spline
  - 2=Cardinal spline
- Apply in 3D (ID=8196; whole): select to enable curve solution also following the depth coordinate
- Curve tension (ID=8192; double): set curve tension (value from 0.0 to 1.0), used in case of curve *Cardinal Spline*
- Sample on the length of the curve (ID=8197; whole): select to enable sampling with splitting of the distance between two following vertices of the original polyline. Otherwise: sampling is assigned as a number of sampled segments
- Sampled segments between two points (ID=8173; whole): set the number of sampled segments between two reference points of the original curve;
- Segment length (ID=8193; double): set the length between two following sampling;
- Movement speed (ID=8157; double): assigns the working parameter of COPL01 profile shown with a parameter Technology: displacement speed (parameter attribute: Transforms

Enabling to generation of B-Spline curves is ignored if a text or emptying generation is already activated.

### Transforms application codes with programmed workings

A complex code can also apply one or more transforms not to a subroutine, but to workings that are directly programmed in the program: working[1042] STOOL: SPLINE is an example of that.

Let's start with an example. You want to program a writing in a frame and then apply an emptying to the external frame, respecting the internal islands.

If writing and frame are both fixed: you can program single entities and then apply emptying tool.

If the writing and/or the frame aren't fixed (for example: change of the writing height and, consequently, of the frame by acting on a <r> variable or on the face dimensions): each time that a variable or dimension is changed the emptying tool should be re-applied. As we have already seen, the problem can be solved applying a subroutine code that manages the emptying. Frame and writing are assigned in a subroutine and the working application [2013] EMPTY to the subroutine determines its emptying.

Anyway it is possible to solve the problem without creating the subroutine and applying the emptying to the same frame and writing programming level, adapting the level to the variation of the profiles to be emptied:

This occurs creating a complex code that operates as programmed tool. The same code is applied not to a subroutine, but to the programmed workings to the same level. In particular: search among upstream programmed workings.

This management is conditioned by a comprehensive enabling for codes of this typology and, in our example, also by the comprehensive enabling of the emptying tool.

STOOL codes are managed in *TpaCAD* only if in CAD mode and with active comprehensive enabling to


codes of this typology and, in our specific example, also comprehensive enabling of the emptying tool.

Configuration modes of these workings are defined here following:

- the complex code has configured PRSETMCRNAM parameter (ID=8098): string typology, hidden and with default value = "\*" (asterisk);
- the complex code has configured PRSETLAVNAM parameter (ID=8117): string typology, editable.

**In these two verified cases:** the complex code recognizes to operate not on a sub/macro but on the workings found upstream of itself. Let's see on which workings it operates:

- none: in case of macro edit Actual development will be only performed during the application of the macro.
- workings as assigned in PRSETLAVNAM parameter:

. The syntax by means of which the parameter is interpreted is of type "name1;name2;...", with its only limitation set by the maximum number of characters that can be assigned (100):

- name1 is the name of the first working or of the first workings group;
- name2 is the name of the second working or of the second workings group;
- ..

If the setting is empty, no working is involved in the transform.

Assignment of the field is assisted: the button

 opens a window showing the list of the names that can be set valid.

The syntax for the names limits the length to 16 characters, the interpretation doesn't react to the use of small or capital letters (for example: "AA" is equivalent to "aa") and only alphanumeric characters are valid.

 the name of a working is assigned in the property "Name".

If programming takes place in piece-face: the only workings assigned to the same face of the complex transform code are accepted.

In any case, comment, logical or complex workings, for which it is not possible to create exploded views are excluded.

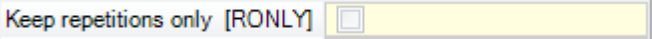
If the transform requires:

- an emptying (that is: assign parameter (ID=8140) of value different from 0), or
  - a generation of Spline curves (that is: assign parameter (ID=8170) of value different from 0)
- only profile workings are involved.

It is possible to configure each geometric transform already seen for a generic complex code that loads a subroutine. Let's summarize here these peculiarities:

- don't interpret parameter of Point hook (ID=8101);
- don't interpret parameters referring to subroutine face selection and to induced faces management;
- don't interpret parameters referring to subroutine variables (PRSETMCRVAR,...);
- if the transform requires an emptying (that is: assign parameter (ID=8140) of value different from 0): delete original workings (it maintains only emptying profiles);
- if the transform requires generation of Spline curves (that is: assign parameter (ID=8170) of value different from 0): delete original workings (it maintains only the profiles corresponding to Spline);
- if the transform requires application of repetitions, interpret parameter (ID=8115; integer).

The parameter is, for example, configured in working [2014] STOOL:



- not selected (0): maintains workings corresponding to the copy of original workings;
- selected (different from 0): delete the first copy of original workings.

## Application of ISO curves

A complex code can apply and display profiles defined in ISO text.

Enabling modes are defined here following:

- the complex code applies an Iso.tmcr macro-program;
- the macro-program application assigns the workings:
  - (optional) a set up working (if not assigned: it uses the geometric set up code COPGSET [2411]);
  - (optional) a profile working COPL01 (if not assigned: it uses the geometric line code COPGLINE [2412]);
  - one or more logical custom or point workings

Check of above mentioned points start in automatic procedure of writing and display of ISO curve. Let's remember that the reading and display procedure of the curve:

- does not manage induced calls;
- does not resolve codes programming: COPNEWSIDE, CPSUBSIDE, complex codes (macro or subroutine).

### Why si an ISO file applied?

Application of an ISO curve (that is: polyline) has the typical function to integrate the processing of a 4 or 5 axes interpolated curve (three axes of Cartesian system with one or two rotating axes) with a piece program.

In this case: TpaCAD environment has the function of formalizing the ISO curve application (that is: it defines its assignment modes) and of visualizing the application itself. Real curve processing occurs downstream of TpaCAD application program, by means of a 5 axes module of optimization.

That is: TpaCAD provides always graphic representation of the curve, but it doesn't generally integrate the curve itself in the piece-matrix. In piece-matrix information about curve application is provided on the basis of specifications of each single application. In particular these specifications:

- are in the macro-program iso.tmc that is always associated to the complex code, as we have already seen;
- are defined in the working/s (custom or point logical) that come out from the macro text to the piece-matrix;
- but in particular:

they have got an interpretation and application correspondence with the custom Optimisation module and/or with the plant cycle (that is PLC cycle). The specifications must be set during the plant design so that each particular functioning is clearly defined.

Anyway it is possible to use this functionality of TpaCAD to apply an ISO curve as defined with 3 axes and integrate it directly in piece-matrix. In this case there is a XYZ profile, only defined with a different format from the one of a normal subroutine.

### How a macro-program is defined

Set up working of macro is used to generate the set up of generated profile with all set technological assignment (tool, correction, speed,...).

If it is necessary to assign a technology also to the profile segments (speed of interpolation): it is possible to assign it to linear typology working (for example with a r variable).

These technological assignments aren't important if the application of the ISO curve in TpaCAD environment is useful only visually.

The complex code must make available some significant parameters with assignment of the same number of (public) <r> variables.

This means that: the macro variables are assigned as parameters of the same complex code (parameters types: PRSETMCRVAR,...).

Let's see the <r> variables made available by the macro:

r..	Type	Meaning
r0:	integer	select the way of assignment of ISO program (path or number)
r1:	string:	path of ISO program
r2	integer	number of ISO program
r3	string:	relative path and/or extension of ISO program
r4	integer	loading flag of ISO curve
r5	integer	flag of curve placement mode
r6	double	offset/quote X of curve placement
r7	double	offset/quote Y of curve placement
r8	double	offset/quote Z of curve placement
r9	integer	offset/quote X application flag
r10	integer	offset/quote Y application flag
r11	integer	offset/quote Z application flag
r12:	integer	Z quote interpretation flag in ISO curve
r13	integer	loading flag of ISO curve run-time

<b>r14</b>	integer	interpretation flag of arcs centres (I, J, K fields in G2/G3 codes), to differentiate between setting in absolute or incremental mode. Interprets values: 0: interprets coordinates as assigned in TpaCAD configuration 1 : interprets coordinates always and only in incremental programming 2 :(>1) interprets coordinates as G90/ G91 functions
------------	---------	---

Also technology parameters are normally assigned with <r> variables of subroutine that can be assigned in addition the ones of normal use, above mentioned.

The called macro-program can manage error situations with proper ERROR or BREAK codes:

- assigned technology for set up not valid,
- .. .

as each normal subroutine (or macro) text.

When the technology elements are acquired (set up and in case profile workings): profile workings resulting from macro application are deleted and the application:

- maintains the remaining workings (logical, custom or point) of the original macro,
- to these it adds the profile deriving from the ISO program interpretation.

Now let's see in detail possible functioning modes of a complex code in an ISO curve displaying.

### How the ISO file is identified

The file ISO name can be assigned choosing between two formalisms, through the r0 macro variable:

- r0=0: the ISO file is assigned with indication of pathname, in r1 variable (*typology*: string);
- r0#0: the ISO file is assigned with a number, in r2 variable(*typology*: integer).

The r0 variable [ID=8500] is generally assigned hidden with set default value.

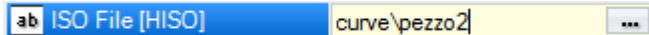
#### If it is r0=0:

[ID=8501], that must be configured with:

- *typology*: string
- visibility visible and editable (if ISO ISO to be applied isn't fix) or hidden with assigned default value (if ISO file to be applied is always the same).

In case of parameter visible and editable: it is possible to assign a control **typologyFile Browse-button Open**, so that

choose the complete path, as for the case of a subroutine assignment (see for example the code: SUB



More specifically:

- the file search is pre-set in the standard storage folder of the subroutines;
- the available files *typology* corresponds to "All files (\*.\*)".

The assigned string for the parameter corresponds to the concerning addressing path for a subroutine, that is: *product\sub\*. In the example, the ISO file has addressing: *..\product\sub\curves\piece2*".

#### If it is r0#0

[ID=8502], that must be configured with:

- *typology*: integer;
- visibility visible and editable (if ISO file to be applied isn't fix) or hidden with assigned default value (if ISO file to be applied is always the same).

The parameter value [ID=8502] assigns the name of the ISO file numerically: the file search starts in this case, too, from *product\sub\*, it is possible now to pre-define a sub-folder and an extension for the file. To this aim:

[ID=8503], that must be configured with:

- *typology*: string
- visibility hidden with assigned default value. Syntax of default value must correspond to one of the cases shown here following with specific examples (let's imagine the number of file ISO=5):

default value	Meaning	Addressing of Iso File
""	The parameter is ignored	"..\product\sub\5"

"iso"	assign file extension	"..\product\sub\5.iso"
"*.iso"	assigns file extension (character * is replaced with the name of the file). The case is exactly the same as the previous one	"..\product\sub\5.iso"
"curves13 integer\*"	assigns a sub-folder of product\sub\ (character * is replaced with the name of the file)	"..\product\sub\curves\5"
"curves\*.iso"	assigns a sub-folder of product\sub\ and file extension (character * is replaced with the name of the file)	"..\product\sub\curves\5.iso"
"..\curve\*.iso"	assigns a sub-folder of product\ and file extension (character * is replaced with the name of the file)	"..\product\curves\5.iso"

### How it is shown to load the ISO file

Let's see how is it possible to load ISO files:

- the variables r4 and r13 of the ISO macro are valued;
- moreover the compilation context is valued/considered (Edit or Run).

In Edit context: the ISO file is always loaded.

In Run context (executive):

- r4#0: loads the ISO file, that arrives in piece-matrix;
- r4=0:
  - r13=0 (default): don't load the ISO file (example of use: in optimisation for piece execution, if ISO file is loaded externally);
  - r13#0: load the ISO file, that arrives in piece-matrix (example of use: in optimisation for a graphic preview).

Variable r4 must be assigned in macro code [ID=8504]:

- typology: integer;
- set value=0 o #0.

If it is useful, variable r13 must be assigned in the macro file (ISO.TMCR):

- typology: integer;
- not reassignable
  - for example =prun1 (so that the value can be changed in piece optimisation).

### ISO file placement mode

A first consideration must be done about interpretation of Z quotes of ISO file:

- Z quotes of ISO file referred to absolute Cartesian system of the piece. In this case: [ID=8512], that must be configured with:
  - typology: integer;
  - set value=**0**.

Zed quotes are assigned with translation of the value of (-sf);

- Zed quotes of the ISO file referred to the system of face 1 of the piece. In this case: [ID=8512], that must be configured with:
  - typology: integer;
  - set value=**1** (#0).

Zed quotes are assigned as in ISO file.

ISO file placement can occur following two modes, on the basis of how the complex working is configured:

- placement following the normal application rules of a complex code. In this case: [ID=8505], that must be configured with:
  - typology: integer;
  - visibility: hidden with default value=**0**.

In particular, in the working starting placement parameters will be configured:



Relative [EG]	<input type="checkbox"/>	<p>The placement mode is in a system of Cartesian coordinates , where it is possible to assign the coordinates in absolute or relative mode.</p> <p>In case of not set coordinate, functioning mode occurs through item "P(x,y,z) of subroutine as point" in page "General assignments of the piece" or with exactly the same specific that can be assigned for the complex code.</p> <p>To the position identified as <i>Application point</i> the starting point of ISO profile is taken.</p>
Rel<- [EGl]	<input type="checkbox"/>	
X1 [X]	lf/2	
Y1 [Y]	200	
Z1 [Z]	-5	
<p>Relative: ID=8015; Whole                  X1: ID=8020; Double                  Y1: ID=8021; Double                  Z1: ID=8022; Double</p>		

- placement referring to starting coordinates of ISO file. In this case: [ID=8505], that must be configured with:
  - typology: integer;
  - visibility: hidden with default value=**1**.

In this case it is recommended not to assign for the complex code placement parameters:

Relative: ID=8015; Whole  
 X1: ID=8020; Double  
 Y1: ID=8021; Double  
 Z1: ID=8022; Double

Placement of the curve is now assigned by r macro variables:

[ID=8506] (corresponds to r6 variable: *typology*: double),  
 [ID=8509] (corresponds to r9 variable: *typology*: integer): assign placement in x of the curve;

[ID=8507] (corresponds to variable r7: *typology*: double),  
 [ID=8510] (corresponds to variable r10: *typology*: integer): assign placement in y of the curve;

[ID=8508] (corresponds to variable r8: *typology*: double),  
 [ID=8511] (corresponds to variable r11: *typology*: integer): assign placement in z of the curve.

For each of the three coordinates, first parameter (example for x: (ID=8506: double)

- translation offset applied to starting point of ISO profile, if the corresponding flag has value 0 (example for x: [ID=8509]).
- absolute position to which the starting point of ISO profile is translated, if corresponding flag has value 1 (different from 0) (example for x: X1 [ID=8509]).

<p> Geometry</p>		<p>It is an example of a complex code possible configuration:</p> <p><b>Absolute X quote</b> : ID=8509; whole  <b>X: offset/quote</b>: ID=8506: double</p> <p><b>Absolute Y quote</b>: ID=8510; whole  <b>Y: offset/quote</b>: ID=8507: double</p> <p><b>Absolute Z quote</b>: ID=8511; whole  <b>Z: offset/quote</b>: ID=8508: double</p> <p>In the example the starting point of ISO profile is displaced in xy plane to coordinates (lf/2; hf/2), while z coordinate doesn't change with respect to original ISO file.</p>
Absolute X-coordinate [ABSX]		
X: offset/ coordinate [X]	lf/2	
Absolute Y-coordinate [ABSY]		
Y: offset/ coordinate [Y]	hf/2	
Absolute Z-coordinate [ABSZ]		
Z: offset/ coordinate [Z]		

### How is the ISO file interpreted

Here following you can see a valid ISO file piece with the interpreted fields in bold letters.

(123)  
**G150 M16 T1**

```
G70
(FLAT 20MM 2F EC HSS)
G0 X-627.857Y0Z312.249 T1 B13.135 A0 ;...(comment)....
G48
G1 X-2.272Y0Z-9.738 P0.22724Q0R0.97384 F6000 T1
G1 X888.346Y0Z-217.56 P0.22724Q0R0.97384 T1 B13.134
G1 X898.083Y0Z-219.832 P0.22722Q0R0.97384 T1 B13.134
...
M2
```

- The file is recognized as valid if the first row starts with one of the strings **%** (percentage), **(** (open round parenthesis), **;** (semicolon), **G150**, **G**;
- rows that start with **%** (percentage), **(** (round open parenthesis), **;** (semicolon): aren't interpreted;
- the character **;** (semicolon), set in a row of the file: makes the next row part a row for comments;
- the default unit of the ISO file is [mm]: to assign the programming unit of the file in [inch] it is necessary to assign the field **G70** before of the first **G0** (and not in not interpreted rows). In the example: we find **G70** on the third row. In case of an ISO file unit different from the one of the program: the ISO profile is converted into a unit of the program

Profile interpretation starts with the first row of **G0** (in our example it is the fifth row) and on this row interpret the fields:

- **(X, Y, Z)** as starting coordinates of the profile
- **(B, A)** as starting values of the rotation axes (then reported on profile set up, if assigned with rotating axes)
- **G90 / G91** for programming of absolute coordinates/ incremental.

Each row following the first one of the profile can assign:

- a linear interpolation segment **G1** and here following fields are interpreted **(X, Y, Z)** as end coordinates of the linear segment, **(B, A)** as rotating axes on the segment and **G90/G91** for programming of absolute or incremental coordinates. A not assigned coordinate is propagated starting from the previous segment. Information on rotating axes is assigned on the linear segment only if it is configured in the workings database
- a segment of circular interpolation **G2/3** (respectively: clockwise/ counter-clockwise) and here are interpreted the fields **(X, Y, Z)** as end coordinates of the linear segment, **(I,J)** as coordinates of the centre, **(B, A)** as rotating axes on the segment and **G90/G91** for programming of absolute or incremental coordinates. A not assigned coordinate is propagated starting from the previous segment. Information on rotating axes is assigned on the circular segment only if it is configured in the workings database The coordinates of the centre are in absolute or incremental mode (following variable r14). A not assigned I or J sets the corresponding value of the segment starting point. If both coordinates of the centre aren't set: interpret a segment of linear interpolation.
- a new profile beginning **G0**.

Possible lines with other codes **G** are ignored.

Program interpretation ends at the end of the file or if field is interpreted **M2**.



## **Tecnologie e Prodotti per l'Automazione**

Via Carducci 221  
I - 20099 Sesto S.Giovanni (Mi)  
Tel. +390236527550  
Fax. +39022481008

[www.tpaspa.it](http://www.tpaspa.it)

[info@tpaspa.it](mailto:info@tpaspa.it)