

Excel Automation: Generation of drawing inserting macros from Excel file

User documentation

Excel Automation

1.	PURPOSE.....	3
2.	FEATURES	3
2.1	CREATE NEW EXCEL FILE FOR AUTOMATION	4
2.2	GENERATION OF DRAWINGS.....	4
3.	MACROS EDITION	5
3.1	MACRO MANAGER / PALETTE	5
3.2	MACRO TYPES	5
3.3	MACRO VARIABLES	5
4.	STRUCTURE OF EXCEL FILE	8
4.1	KEY ROW FOR COLUMN IDENTIFIER.....	8
4.2	FIELD NAMES	9
4.3	MACRO DEFINITION FIELDS.....	9
4.4	SPECIAL FIELDS.....	9
4.5	DATABASE FIELDS	9
4.6	COLUMNS FOR VARIABLES	12
4.7	PROCESSING EXCEL FILE, ERRORS AND WARNINGS	13
5.	USER RIGHT / SPECIAL EDITION.....	16

Excel Automation

1. Purpose

The purpose of this Excel Automation is to automatically generate drawings, inserting macros in the drawing, following the instructions from an Excel file.

These instructions are:

- Name and position of the **macro** that you want to insert.
- Name and description of the **drawing** where you want to include the macros.
- Name and description of the **book** for this drawing.
- Name and description of the **location** associated with the book, the drawing and where all the objects will be inserted.
- Name and value of **variables** used to define data in macros to insert

The specifications for each macro include some variables that will be changed by their values during the insertion process.

2. Features

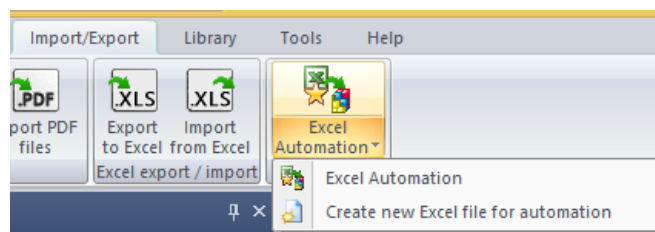
Practically, to be able to generate custom drawing from Excel, the user needs to:

- Save macros corresponding to the part of drawing
- Edit macros to set variables for macro values to be replaced at inserted
- Define an Excel template with the format he wants to use and the necessary information to define macro to insert.

Once this preparation work is done, user can create new Excel file from template, fill the content to define which macros to insert and generate drawings.

From ribbon menu, in Import / Export tab, two features are available for Excel automation.

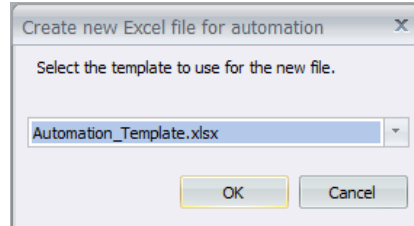
- Creation of a new Excel file from a predefined template.
- Generation of drawings from an Excel file.



Excel Automation

2.1 Create new Excel file for automation

This feature will look for Excel template and propose to create one new file, copying the selected template.



Once a template is selected, the user is asked for destination of new file, then it is open to let user fill the file for generating drawings.

Templates location:

The templates listed are the Excel files existing in the folder "XlsAutomation\Template" from application data folder. The user is free to copy and modify the existing templates to create one with the formatting and the details he wants.

Supported file extension are

- .xlsx – Excel workbook
- .xlsb – binary Excel workbook
- .xls – Excel 97 -2003 workbook
- .xlsm – Excel workbook with VBA macros.

2.2 Generation of drawings

When he clicks on button Excel automation, the user is asked to select an Excel file describing macros to be inserted. Once validated, process is started and generation of drawings begins. In the end, a report is displayed to inform about possible errors or warnings.

Excel Automation

3. Macros edition

Note: here when talking about “macro” it means macro from electrical project, not to be confused with VBA macro that is the script you can use in Excel to reproduce a sequence of actions.

In Excel file, the user will specify:

- Macro to insert.
- Where to insert macro: book, drawing, and coordinates in drawing.
- Set general information about book, drawing, location, function.
- Set values for macro variables.

To be able to do that, the macros must be edited, especially to define **variables**.

3.1 Macro manager / palette

New class “Drawing automation” is available in macro manager to help user to organize the macros he wants to use for Excel automation.

3.2 Macro types

When saving a macro for Excel automation, the user must have to ensure the type of macro will be compatible with the type of drawing he wants to use (mixed scheme by default), otherwise error and warnings could be raised.

Excel automation support only the following types:

- Scheme macro
- Line diagram macro
- Mixed scheme macro

Undefined macros (done before version 2016) are not supported.

3.3 Macro variables

When the user saves a macro, he is able to edit it and defines some “**variables**” in order to modify the characteristics of the macro before insertion from Excel file.

Variables are surrounded by **%** character with no specific limit on the name. It is recommended to avoid whitespace like **%My variable%** and use explicit variable names **%CableMark%** better than **%Variable1%**.

When editing the macro, the user will modify object property to set the variable to be able to replace by the value according to information described in Excel file.

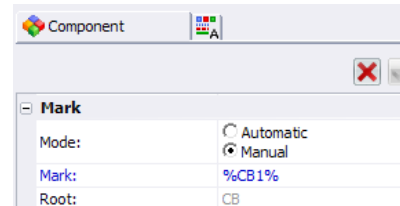
One variable can be used several times in a macro, but should be used for the same information.

Excel Automation

3.3.1 Variable for objects mark

Variable can be used for all the marks of objects with mark: locations, functions, components, cables, equipotential, wires, etc...

When editing the macros, select object properties, set mark mode as manual, to be able to replace the existing mark by the variable.



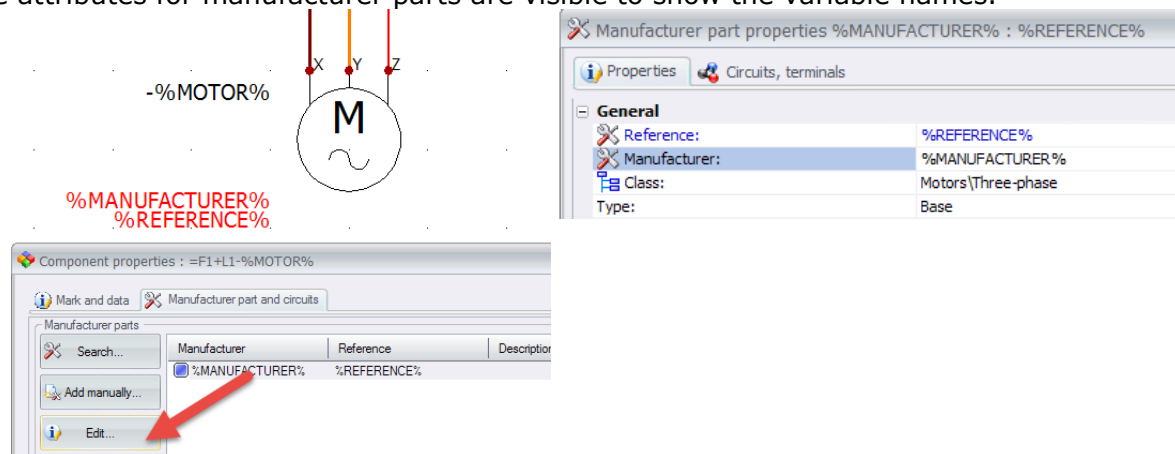
3.3.2 Variable for manufacturer reference

For manufacturer parts, the user needs to define at least one or two variables:

- For reference only, if want to keep existing manufacturer
- For reference and manufacturer, to be able to choose any part.

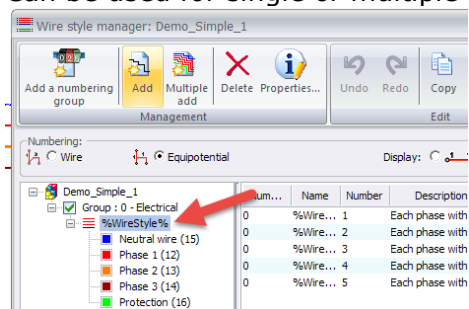
Having a variable only for manufacturer not for reference could work but not really make sense. Using edit on part properties, variables have to be set for Reference and / or manufacturer.

When editing macro to set variable on a manufacturer part, it would be useful to ensure the attributes for manufacturer parts are visible to show the variable names.



3.3.3 Variable on wire style name

Variable can be used also to change the wire style of a macro. Can be used for single or multiple wire style.



Excel Automation

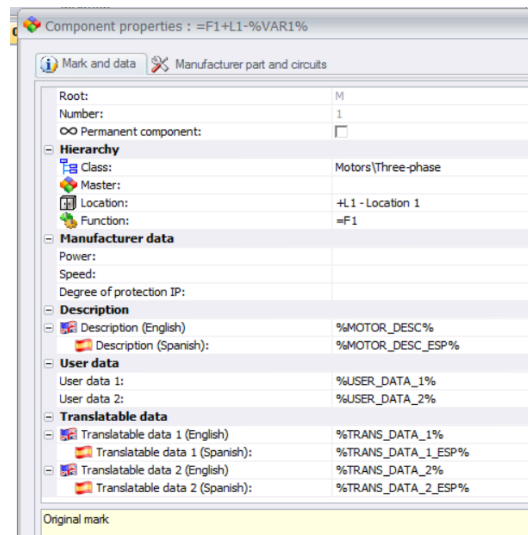
3.3.4 Variable on translatable data and user data

Variable can be used also for any object that has a mark for:

- * Description
- * Translatable data
- * User data

You can see that the description has a variable in both languages of the project.

Any translatable data for any language can contain variables.



3.3.5 Formulas

User could also enter a "formula" in any of the places that the automation look for a variable in macro. The automation will recognize a formula if the text in this places is between the characters "{" and "}". The content between those two characters is a formula similar to any formula we have in the application. Variables in the formula MUST be surrounded by the character %, in the same way as the one in the Excel file: **%VARIABLE%**.

Only the variables on the Excel file will be available for the parser of the formula. As for any formula, variables must match exactly in the formula. Difference in capital letter for example will make formula fail.

Example of formula

{%Root% + "_" + %Order%}

In Excel, if you set variables **%Root% = "ABC"** and **%Order% = "10"**, data in macro will be replaced by **"ABC_10"**.

For the same formula, if you set **%ORDER% = "10"**, evaluation of formula will fail, due to case difference in variable name.

If the parser gives any error evaluating the formula, the automation process will show this error on the final dialog of the results:

"Error while parsing a formula during insertion of the macro Test_Formula_Error: Some parse error."

Excel Automation

4. Structure of Excel file

The provided templates are just example, they can be fully customized to user needs, for example

- to add / remove columns
- change formatting
- define validation
- show list boxes to fill the data easily
- add buttons, toolbar to run VBA script (vba macros)
- Contain VBA script to define wizard to fill easily the Excel file

Only the first **visible** sheet of the Excel file will be processed, others will be ignored, so other pages can be used to define additional data, like list of macro user can insert... There is no restriction on the sheet name.

4.1 Key row for column identifier

To identify the purpose of each column, there must be a row containing key information. This row is the first one starting with # character in first column.

Changing text in this row can prevent generation to work properly, so it is recommended to hide the row once required modification are done, and keep visible only the user friendly titles.

	A	B	C	F	G	H	I	J
1	Macro			File			Book	
2	Macro	X Position	Y Position	Mark	Type	Description	Mark	Description
3	#mac_name	#mac_posx	#mac_posy	#fil_title	#fil_filetype	#fil_tra_0.11	#bun_tag	#bun_tra_0.11
4	Demo_Simple_1	30	260	10	12	Automation Demo	5	Automation
5	Demo_Simple_2	120	260	10	12	Automation Demo	5	Automation
6	Demo_Simple_2	160	260	10	12	Automation Demo	5	Automation
7	Demo_Simple_2	200	260	10	12	Automation Demo	5	Automation
8	Demo_Simple_2	240	260	10	12	Automation Demo	5	Automation
9								

All the rows before the key row are not processed. They can be used to indicate the content expected, in users language, can be merged, or apply any formatting.

All the rows after the key row will be processed for inserting macro.

A row without macro name will produce a warning, nothing will be inserted and process will continue.

Each cell of the key row must:

- Start with a #, like **#mac_name**, to indicate a special information or a database field.
- Or be surrounded by %, for variable, like **%VAR1%**.

Position of columns does not matter, only the title given in key row is important.

Column with nothing in Key row will be ignored, and produce no warning.

Excel Automation

4.2 Field names

Columns from the Excel sheet can be associated to different types of information:

- Macro definition fields.
- Special fields.
- Database fields.
- User data / translatable data
- Macro variables.

All columns are identified by text starting by # or surrounded by % for variables.
Columns without such identifier are not processed.

4.3 Macro definition fields

The fields for macro definition will give instructions about the macro to insert:

- Macro name
- Coordinate of macro insertion, in drawing units.

There is no corresponding fields in project database, so those data will not be added to project database, just used to generate the drawings.

If those fields are not empty for a row, warning will be raised and row will not be processed
If those fields are missing from the Excel sheet, the whole processe is cancel and error is raised.

FIELD NAME	DESCRIPTION	REQUIRED
#mac_name	Name of the macro.	Yes
#mac_posx	X and Y position where the macro will be inserted in drawing coordinates.	Yes
#mac_posy		Yes
#mac_insert	If the column is present, not hidden and contains a 0 or is empty, the macro will not be processed.	No

4.4 Special fields

There is one special field name, **#title_lang** that is only like a comment. If you add a column with this header it will not be processed. It's like a comment.

It is reserved for a future use to be able to define column title in several languages to be able to switch from a language to another.

4.5 Database fields

Database fields are optional fields corresponding to data from project database. They are usually named as the field in the database table. Only the fields from the following tables are supported:

Object	Table	Field prefix	Required field
Drawing	tew_file	fil	#fil_title (always required)
Book	tew_book	bun	#bun_tag (required if object used)
Location	tew_location	loc	#loc_text (required if object used)
Function	tew_function	fun	#fun_text (required if object used)

The only field that is required is **#fil_title** that is the name of the drawing where the macro will be inserted. For the other tables, the mark (tag) of the corresponding table is required, if you use any fields from this table. For instance, if you insert any location fields, then the field #loc_text is required.

Detailed and up to date list of available field is defined in document about project database structure, so as the content of some fields.

Excel Automation

4.5.1 Excluded database fields

By default, any field corresponding to an information visible in user interface could be used. The internal data user cannot edit manually are excluded, such as:

- Database object ID.
- Read only information.
- Related to automatic mark of object.

Here is the list of the only supported field, others are excluded.

Object	Table	Field	Description
Book	tew_bundle	bun_manual	Manual or automatic mark
Book	tew_bundle	bun_tag	Mark of the book
Drawing	tew_file	fil_filename	Name of the file on the disk
Drawing	tew_file	fil_filetype	Drawing type
Drawing	tew_file	fil_manual	Manual or Automatic mark
Drawing	tew_file	fil_title	Title displayed in treeview (mark)
Drawing	tew_file	fil_titleblock	Title block file name
Function	tew_function	fun_markmanual	Lock for mark
Function	tew_function	fun_tagpath	Mark path (full mark)
Function	tew_function	fun_tagroot	Root for mark
Function	tew_function	fun_text	Mark
Location	tew_location	loc_tagmanual	Lock for mark
Location	tew_location	loc_tagpath	Mark path (full mark)
Location	tew_location	loc_tagroot	Root for mark
Location	tew_location	loc_text	Mark

4.5.2 User data / translatable data

User data and translatable data can be also added as a field, but as they are not directly in object table, the field definition is a bit different. It is formatted the following way for translatable data:

#ttt.tra_nn.xx

Where

- **ttt** is to replace by the table prefix (fil, bun, loc, fun).
- **tra** is for translatable data (do not modify).
- **nn** is to replace by index of data you need (0 for description).
Up to 14 for translatable data.
- **.xx** is to replace by the language code, usually .l1 (L1) for main project language
Supported language code are:
From project languages (L + 1 to 3): "l1", "l2", "l3"
- Standard language code are: en, es, fr, it, ru, ko, ja, pt..., zh, zh-tw

Example, for the description of the drawing in the project main language: **`#fil.tra_0.l1`**.

For user data, the syntax is similar, but without language to specify.

#ttt.use_datann

Where

- **ttt** is to replace by the table prefix (fil, bun, loc, fun).
- **use_data** is for user data (do not modify).
- **nn** is to replace by index of data you need, from 0 to 19 for user data.

Excel Automation

Example, for drawing user data:
#fil.use_data0.

If the project is configured only with one language and you add any column with language code .l2 or .l3 then the process will give you a warning like this:
"The language specified on column %d: %s, is not valid. The project does not contain text using this language."

4.5.3 Most common database fields

The most commonly used fields for drawing are listed below.

FIELD NAME	DESCRIPTION	REQUIRED
#fil_title	The mark of the drawing.	Always
#fil_filetype	The type of the drawing (scheme line diagram)	No
#fil.tra_0.xx	The description of the drawing, where xx is language code	No
#fil.use_data0	User data of drawing.	No

Supported values for drawing type are:

Schematic	0	Scheme mixed (default)	12
Line diagram	1		
Cover page	5		
2D cabinet layout	9		

Warning:

If you insert a drawing type of "Cover page", then the name of the macro should be empty. This is provided only to let the user to enter a cover page, for presentation purpose, in the process of automation. If you set the macro name in a row with this type of drawing, the automation will give you a warning at the end of the process.

For other objects, the most commonly used columns are for mark (tag) and description in current language (using l1 for language code).

FIELD NAME	Object	DESCRIPTION	REQUIRED
#bun_tag	Book	Mark	If any book used
#bun.tra_0.l1	Book	Description in current project language.	No
#loc_text	Location	Mark	If any book used
#loc.tra_0.l1	Location	Description in current project language.	No
#fun_text	Function	Mark	If any book used
#fun.tra_0.l1	Function	Description in current project language.	No

4.5.4 Set automatic mark on objects

By default, all the created objects, drawings, books, location and functions will be created with manual mark. To have automatic mark, then you should insert the field xxx_tagmanual or xxx_manual and set it to 0. In this case, the drawing mark will be generated by the product according to the formula defined in project settings.

If you set the tagmanual to 1, then the column mark has no effect, the name will be automatic and the user will only be able to insert one macro in this drawing, because the next line will be a different drawing.

Excel Automation

4.6 Columns for variables

Once the way to insert macro and database field are defined (drawing, book, location and function), it is possible to specify the variables to replace by their values when inserting macros. There are two possible ways to define variables and corresponding values. Both can be used in the same Excel file:

- Generic column for any variable
- Column specific to a variable

For both methods, the name of variable is not case sensitive. We can have variable **%MyVariable%** in macro, and **%myvariable%** in Excel and it will match.

But, if variables are used in formulas (see chapter **3.3.5**), then case must be respected.

4.6.1 Generic column for any variable

With this method, we will have two consecutive columns, named (in this order):

- #mac_var_name
- #mac_var_value

They respectively indicate the name of the variable as defined in macro and the value you want to use instead. Several couples of columns **#mac_var_name / #mac_var_value** can be used in the same Excel file.

Example, in this case where we insert the macro Test0, the variable %motor1% will be changed by M5 and the variable %motor2% will be changed by M6.

#mac_name	#mac_var_name	#mac_var_value	#mac_var_name	#mac_var_value
Test0	%motor1%	M5	%motor2%	M6
Test1	%var1%	V3	%var2%	V4

This method is more flexible and requires two times the number of variables you can find in a macro. But it requires to write the name of variables every time it is used.

4.6.2 Column specific to a variable

With the second method, the column name is assigned to a specific variable.

Example, in this case, the first line will:

- insert the macro **Test0**
- change the variable **% motor1%** by **M5**
- change the variable **% motor2%** by **M6**

With the second line:

- insert the macro **Test1**
- change the variable **%var1%** by **V3**
- change the variable **%var2%** by **V4**

#mac_name	%motor1%	%motor2%	%var1%	%var2%
Test0	M5	M6		
Test1			V3	V4

For this method, only one column is required by variable. But you need one column for each existing variable name. This method is convenient when a variable is used in many macros, but will require lot of columns to manage all the variables.

Excel Automation

4.7 Processing Excel file, errors and warnings

4.7.1 General test before processing the Excel file

Some general tests are done even before starting to process the Excel file for generation. Here is the list of possible errors:

Case	Type	Output	Message
Must have project open	Error	Message box:	"A project must be opened to use this command."
Active project must be project type not macro type	Error	Message box:	"You can only execute this command on a project."
The project is used by other users	Error	Summary log	"You cannot start this process because this project is opened by other user."
Try to lock project before generation	Error	Summary log	"You cannot start this process because this project is locked."
Other user try to open project during generation	Error	Summary log	
The user has specified an Excel file that does not exist	Error	Summary log	"The file "testfile.xls" does not exist."
The file is not an Excel file	Error	Summary log	"Error opening the Excel file Test.xls: "readHeader: file is corrupt ..."
Look for key row (header) in Excel file	Error	Summary log	"The Excel file has an invalid structure. It has not header."
The column name should start by # or by %	Warning	Summary log	The header of the column 3, "MyColumnName" is not valid. The first character is not '#' or '%'. "
The project destination has only one language and you have specified some fields with I2 or I3.	Warning	Summary log	"The language specified on column 5: #bun.tra_0.I3, is not valid.\n\nThe project does not contain text using this language."
The column name does not correspond to any table on database	Warning	Summary log	"The header of the column 12, "#com_tag" is not valid. Table incorrect."
The column name does not correspond to any valid field on database	Warning	Summary log	"The header of the column 5, "#myfield" is not valid. Field does not exist."

Excel Automation

Then, the rows of the Excel file are processed to generate drawings, and at the end the project is unlocked. If any error happens during process, a summary dialog is displayed to list them.

Case	Type	Output	Message
Look for macro name on a row	Error	Summary log	"There is no column with the name of the macro: %s."
Look for macro position X on a row	Error	Summary log	"There is no column with the position in X of the macro: %s."
Look for macro position Y on a row	Error	Summary log	"There is no column with the position in Y of the macro: %s."
Look for mark of the drawing on a row	Error	Summary log	"There is no column with mark of the drawing: %s."
Look for book mark field because there are some book fields	Error	Summary log	"There are some columns for books, but there is no column with the mark: %s."
Look for location mark field because there are some location fields	Error	Summary log	"There are some columns for locations, but there is no column with the mark: %s."
Look for function mark field because there are some function fields	Error	Summary log	"There are some columns for functions, but there is no column with the mark: %s."
Check if all listed macros exist	Error	Summary log	"Error: the macro %s does not exist on row %d."
Check for variable existence	Warning	Summary log	"The variable ""%s"" found in macro, %s not found in Excel row."
A row with a drawing type of cover page has a macro name.	Warning	Summary log	"The drawing type of row %d is %s. The macro ""%s"" will not be inserted."
Check compatibility between macro type and drawing type	Error	Summary log	"The type of the macro %s, %s, is not compatible with the type of drawing %s."
The process has been cancelled	Warning	Summary log	"The process has been canceled by the user."
Process finish correctly	Information	Summary log	The process has finished correctly.

Excel Automation

4.7.2 Processing main columns

Once the minimum checks are done, for each row describing a macro to insert, the values are processed the following way:

Columns for mark of book, location and function:

- If there is a mark defined, if object does not exist, it is created.
- If there is no mark defined (column missing or empty), it gets the default object of the project.

Columns for drawing mark / type:

- If the drawing does not exist, creates the drawing and if they are defined, associates it to book, location and function.
- If the drawing exists, drawing is not changed, not deleted or cleared (associated book, location and function are not affected).
- If there is the column #fil_filetype, then the drawing will be created of this type. Otherwise will be created of type mixed scheme.

Columns for macro:

- Macro name. If the macro does not exist, gives a warning, skips the current row, but continues with the process.
- Check the type of macro is compatible with type of drawing. Otherwise warning is raised and current row is skipped, but continue with the process. Project macros are not allowed in any case, as they can have multiple pages.
- Insert the macro in the current drawing, in the position indicated by the X, Y or the row and column mark.
- In the insertion process, changes all the variables by their corresponding values.

4.7.3 Processing variables

- If the object has a mark that is a variable that exists in the Excel file, then the variable will be replaced by its value. Besides that, it will look if the mark already exist on the destination project and, in this case, it will associate it with this object. For instance, if a component of the macro has the mark name "%VAR1%" and his value is M5, then the final name of this component will be M5. If the component M5 already exist on the project, this component will be associated with it.
- If the object is a component that has a manufacturer part, and the name of the manufacturer is a variable and/or the reference is a variable, the command has to change the variables by their respective values and assign the component to the correct manufacturer part.
- If the object is a component, assigned to a location that has a mark that is a variable, the command has to change the variable by his value and do the appropriate action (creates if doesn't exist, associates with the location on destiny if exist).

Excel Automation

5. User right / special edition

If user rights is enabled, this feature cannot be started by any user.
This feature is not either available for any edition of the application.
Refer to feature grid to see where applicable.