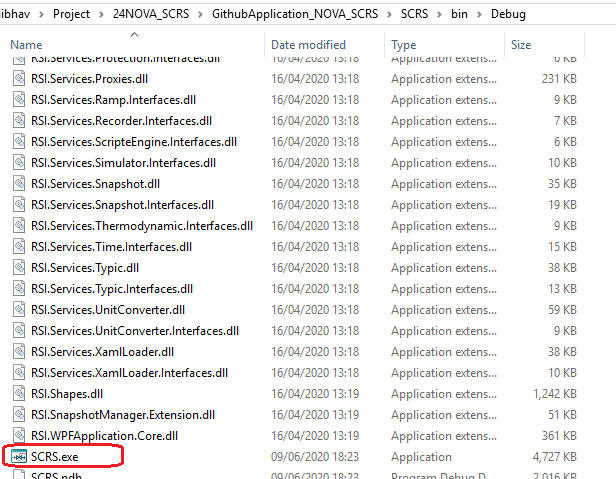
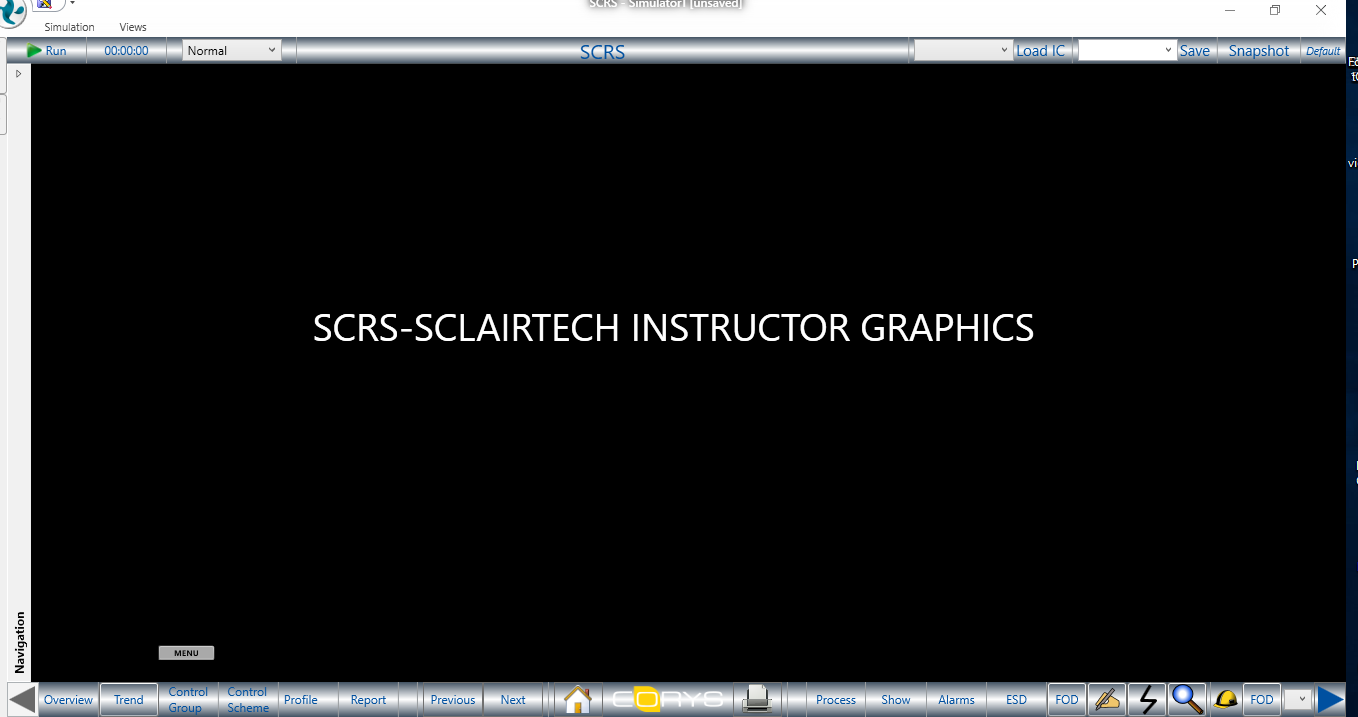
* **Steps for Translation the graphics from xml (DCS files from client)**

1. Make sure that blend is closed during translation of graphics
2. Make sure that model is running to run the application
3. Run the application .exe from bin/debug



1. After running application you will get starting window of application



* **Steps for Translator Centum VP**

## CentumVP Translator

CentumVP translator is used to convert the native files from customers into translated graphics in order to:

* Implement the instructor layers for instructor functionalities
* Create a project HMI object library and recover the yokogawa animations
* Put in place database.xml to communicate with IndissPlus model by configuration of the tagname.

### Use of Translator

Client Graphics can be directly translated from CentumVP framework.

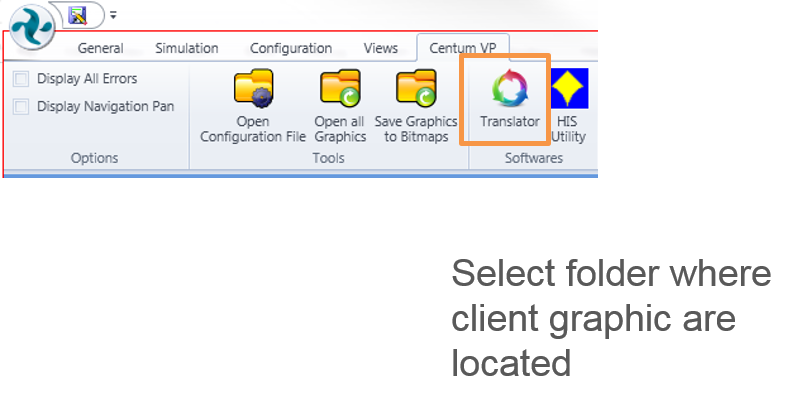


Figure 12 CentumVP Configuration Ribbon

* Click on Translator button
* Select files folder where original graphics are located
* Select options of translation detailed below:
  + Destination panel
    - Set the project folder and project.csproj file. By default configuration shall not be modified. It is already well defined.
  + Configuration Panel
    - Crossreference file: N/A
    - Namespace name: N/A
    - ProjectBase library: N/A
    - Generate Xaml layers: Translated xaml graphic files are generated with 4 layers; SYNOPTIC\_OVERLAY, FOD, BL, FAILURE used for instructor graphic object implementation see documentation part 1.1.2.
    - Generate Table of Content: Add TOC in Database.xml
    - Beautify Xamls: Make Xamls more readable for human being
* Click on Import to start the translation



Figure 13: CentumVP translator

### Result of Translator

***From CentumVP framework:***

When translation is done, close the graphic application and see results in Blend.

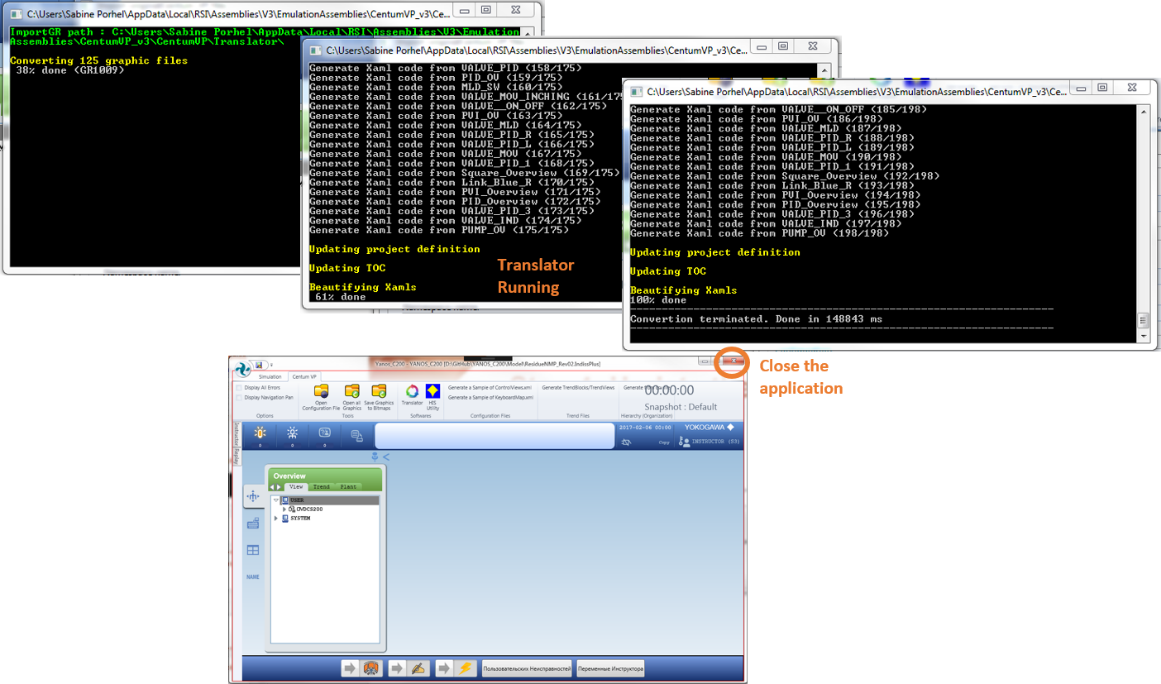


Figure 14: Graphic translator

***From Blend:***

When translation is done, the .sln shall be reloaded in order to take into account the result of the translation.

As results, two folders have been created:

* GraphicViews: in this folder developer will find the translated graphics in subfolder Translations and all graphics generated with 4 instructor layers.
  + From translated graphics, all graphic objects, background objects can be edited.
  + From instructor layers, graphic objects can be inserted to put in place the instructor functionalities.
* HMI: in this folder developer will find the graphic objects defined as Controls. Actually translator generates a graphic objects library based on original files which can be used in any graphic views. Also these graphic objects are defined as controls to manage as fast as possible the modification. When one correction is applied on one object of the HMI, the others objects instanced will be updated.

### Quality of Translator:

The quality of the translation is mainly based on two criteria:

* The quality of the native files provided by the customers

Quality of the native files can impact the spending time to develop graphic application.

Actually the nature of native files will have an impact on translated graphics which will lead to extra work for project team. For instance sometimes graphic objects are ungrouped leading to a configuration object by object on each graphic instead of the configuration of a control HMI. Other example: some CentumVP expressions (for animation purpose) are not written correctly leading to a full implementation of animation by project team for CentumVP objects.

* The quality of the translator

The quality of the translator is improved from project to project. But some new things can appear during a new project. It is advised to create request in order to log the missing things and improve continuously the translator.

* **Steps for Translator ABB**

1. Create ABB Project named XXX
2. Under XXX/XXX directory:
3. Add « Colors.csv » under ConfigurationFiles directory
4. Create new folder « TranslationFiles »
5. Add « CrossRefGenericElements.csv » under TranslationFiles directory
6. Launch translation
7. Before compiling, modify following xaml files manually (files contain two controls with same name which is not possible in Indiss+ )
8. XXX/XXX/HMI/\_3DThreewayValve\_3DThreewayValve.xaml : replace “Test1” by “Test100” (line 49) and “Test2” by “Test200” (line 66)
9. XXX/XXX/HMI/ChokeValve\_ChokeValve.xaml: replace “Test1” by “Test100” (line 47) and “Test2” by “Test200” (line 65)
10. Compile Project