Procedure

CREATION OF A CODESYS UNICOS-CPC6 APPLICATION FOR SCHNEIDER SOMACHINE

Abstract

This procedure explains how to create a CoDeSys UNICOS CPC6 application with Schneider Somachine from the specifications using the UAB (UNICOS Application Builder) generation tool.
## HISTORY OF CHANGES

<table>
<thead>
<tr>
<th>REV. NO.</th>
<th>DATE</th>
<th>PAGES</th>
<th>DESCRIPTIONS OF THE CHANGES</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1.0</td>
<td>19.03.2013</td>
<td>All</td>
<td>Document Creation (M.Koutli) for UAB(1.4.x)</td>
</tr>
<tr>
<td>V1.1</td>
<td>11.12.2013</td>
<td>All</td>
<td>Update document for UAB(1.5.0-1.6.0)</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS

1. INTRODUCTION .................................................................................................................. 5
   1.1.1 SUMMARY ............................................................................................................. 5
   1.1.2 ARCHITECTURE OVERVIEW ............................................................................. 5
2. REQUIREMENTS .................................................................................................................... 6
3. PROCEDURE .......................................................................................................................... 6
   3.1 FILL IN THE SPECIFICATIONS ................................................................................. 7
   3.2 UAB GENERATION TOOL ....................................................................................... 7
   3.2.1 INSTALLATION OF UAB GENERATOR TOOL .................................................... 7
   3.2.2 GENERATION PRINCIPLES .............................................................................. 7
   3.2.3 GENERATION PROCEDURE .............................................................................. 7
   3.2.4 LOG ...................................................................................................................... 16
   3.2.5 TEMPLATES ......................................................................................................... 16
   3.3 CREATING THE PLC PROJECT ............................................................................. 17
   3.3.1 BASELINE .......................................................................................................... 17
   3.3.2 HARDWARE CONFIGURATION ............................................................................ 17
   3.3.3 ETHERNET CONFIGURATION ............................................................................ 18
   3.3.4 CONNECT TO THE PLC TARGET ...................................................................... 19
   3.3.5 PREPARATION FOR THE FIRST TIME ............................................................... 20
   3.4 IMPORTATION OF THE GENERATED FILES INTO SOMACHINE PROJECT ......... 21
   3.5 PLC CODE COMPILATION ....................................................................................... 24
   3.6 CREATION OF WINCC OA PROJECT ..................................................................... 25
   3.7 USAGE OF TOUCH PANELS ................................................................................... 26
4. SUPPORT .............................................................................................................................. 26
5. REFERENCES .......................................................................................................................... 26
# TABLE OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>System Architecture</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>CPC-Wizard</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>New UAB application</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>Application Data</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>PLC Specifications</td>
<td>10</td>
</tr>
<tr>
<td>6</td>
<td>Plugins</td>
<td>11</td>
</tr>
<tr>
<td>7</td>
<td>Instance Generator</td>
<td>12</td>
</tr>
<tr>
<td>8</td>
<td>Logic Generator</td>
<td>13</td>
</tr>
<tr>
<td>9</td>
<td>WinCC OA generator</td>
<td>14</td>
</tr>
<tr>
<td>10</td>
<td>Touch Panel Generator</td>
<td>15</td>
</tr>
<tr>
<td>11</td>
<td>Report Window</td>
<td>16</td>
</tr>
<tr>
<td>12</td>
<td>Hardware Configuration</td>
<td>18</td>
</tr>
<tr>
<td>13</td>
<td>Edit Ethernet Parameters</td>
<td>19</td>
</tr>
<tr>
<td>14</td>
<td>Connect to the PLC</td>
<td>20</td>
</tr>
<tr>
<td>15</td>
<td>Customize SoMachine menu to have ImportPLCOpenXML order</td>
<td>21</td>
</tr>
<tr>
<td>16</td>
<td>Somachine importation</td>
<td>21</td>
</tr>
<tr>
<td>17</td>
<td>Browse the importation file</td>
<td>22</td>
</tr>
<tr>
<td>18</td>
<td>Choose the imported objects</td>
<td>23</td>
</tr>
<tr>
<td>19</td>
<td>Messages</td>
<td>23</td>
</tr>
<tr>
<td>20</td>
<td>Object overwrite</td>
<td>23</td>
</tr>
<tr>
<td>21</td>
<td>Add Topology file in Task Configuration</td>
<td>24</td>
</tr>
<tr>
<td>22</td>
<td>Build</td>
<td>25</td>
</tr>
</tbody>
</table>
1. INTRODUCTION

The goal of this document is to provide a procedure of creating a UNICOS-CPC application using the Schneider PLC M258, Somachine software and the UAB tool.

1.1.1 SUMMARY

Here follow the general steps for the creation of a Somachine UNICOS-CPC project:

1. Fill in the specification file according to the application (Spec.xml).
2. Use UAB tool to generate:
   a. The Instances and Communication for the PLC program from the CoDeSys Instance Generator.
   b. The Logic and Topology for the PLC program from the CoDeSys Logic Generator.
   c. The WinCC OA Instances database from the WinCC OA Instance Generator.
   d. The Touch Panel instances for the Magelis touch panel from the TouchPanelGenerator. (Optional)
3. Open the Baseline PLC project.
4. Configure the hardware.
5. Import the generated files into the PLC project: Instance, Communication, Logic and Topology files
6. Complete the Logic in the PLC project (it can also be done by using User Templates before step 2)
7. Build and Download the project to the PLC.
8. Create the WinCC OA project and import the instances(UAB file).
9. Create the Magelis application and import the instances(UAB file).

1.1.2 ARCHITECTURE OVERVIEW

Codesys is a new platform being introduced in UNICOS-CPC. Currently two Codesys-extended developing systems are supported by the UAB generation tool, the Schneider’s Somachine and the Beckhoff’s Twincat. In this document we will refer only to Somachine but a similar concept is followed for a UCPC6 application with Twincat. To know more about Codesys in UNICOS-CPC take a look at the ICALEPCS 2013 paper CoDeSys in UNICOS: Opening the floor to PLCs and IPCs [1]. To follow the procedure
for creating a CPC application with Twincat see the Procedure Twincat-UCPC Application document [2]

In Figure 1 you can see a typical configuration of a UNICOS-CPC6 project.

The SCADA (WinCC OA) and the PLC are linked by the Ethernet Network. For the communication is used a CERN defined extension of Modbus/TCP protocol, the TSPP (Time Stamp Push Protocol). The PLC is linked with the sensors and the actuators via the I/O cards or with a fieldbus.

![System Architecture Diagram]

*Figure 1 - System Architecture*

## 2. REQUIREMENTS

Developing a Somachine UNICOS application implies the usage of several software packages:

- Specification and Generation tools: **MS Office Excel** (v2007/v2010), **Java SE Runtime Environment** (v7) and **UAB** (v1.5.x or higher)
- PLC: **Somachine V3.0**
- SCADA: **PVSS/WinCC OA**(V3.8 or V3.11)

## 3. PROCEDURE
3.1 FILL IN THE SPECIFICATIONS

The specification file is an xml file where all the UNICOS objects instances must be defined and parameterized according to the functional analysis. To know about how to fill the specification file check the document *UCPC Spec Documentation* [3].

3.2 UAB GENERATION TOOL

3.2.1 INSTALLATION OF UAB GENERATOR TOOL

For the installation of UAB tool follow the instructions described in the following [Link].

3.2.2 GENERATION PRINCIPLES

The generators used for a Codesys application are four:

- Codesys Instance Generator: It generates the instances code for the PLC and the code for the communication with the SCADA.
- Codesys Logic Generator: It generates the logic code for the PLC and the execution order of the code parts.
- WinCC OA Instance Generator: it generates the importation file for the supervision.
- Touch Panel Generator: It generates the instances file used for touch panel applications. In the case of CoDeSys, only the Magelis touch panel was used.

3.2.3 GENERATION PROCEDURE

The launching of the UAB tool is done by launching the UAB Bootstrap. In the Bootstrap there is the button for selecting the CPC Wizard to run and when a new resource package is available you can see and install it by clicking on *Check Updates*, Figure 2.
The first panel that will appear on the Wizard is the one shown in the Figure 3. In this panel by clicking on the first radio button a new application will be created and here the user has to choose a platform and a resource package. In this case, the selection is CoDeSys from the target platform list and the resource package is the latest one, though the users can choose the one that is fitting their application the most. Finally, at the bottom of this panel there is the browse button for choosing a location in the system for saving the new application.

In case you have created the application in a previous time you can open or upgrade this existing application by selecting the appropriate option and specifying the application folder.

Figure 2 -CPC-Wizard

Figure 3 -New UAB application
By clicking the Run button, a new window will appear where the user can fill the Application General Data. In this panel is where the Project Name and the Application Name are given. Additionally, the directory of the specifications file is defined in this panel. The fields in the upper part of this panel are obligatory to fill in, though the lower part is optional. So, optionally, the user can introduce extra information to the application, like a description, a comment or contact person. (Figure 4)

![Application Data Panel](image)

**Figure 4 - Application Data**

The next panel, Figure 5 after the Application General Data is the panel dedicated to the PLC specifications. There can be defined a name for the PLC the PLC Environment which in this case is SoMachine and the type of the PLC which in this case is M258(Schneider).

In the middle of the PLC Specifications panel, there are 2 groups concerning the Ethernet and MODBUS parameters. For the Ethernet there are inputs for the PLC IP address, the gateway and the network mask and also the IP of the SCADA Data Server (and also for a second DS if there is). The user can also define the MODBUS parameters. For an Ethernet communication the Netlink parameter should always be 3. The number of tables per cycle depend on the performance of the PLC used. The recommended value is 2 respectively for Analog and Binary tables. The Modbus Unit can be freely selected but should be unique for its SCADA project or an error will occur during the importation of the database in WinCCOA.
Also the recipes mechanism can be enabled or disabled and the PLC memory address (predefined 10000) can be set for the recipe buffers as well as their length (predefined 1000).

At last the user can set the PLC memory starting address of the variables for the data which are transmitted from the PLC to SCADA(Binary and Analog tables) and the starting address of the variables for the received values of the PLC from SCADA. According to these values variables will be created and mapped in the PLC memory. You can find more information for the memory mapping in the document *Codesys Middleware* [4].

![PLC Specifications](image)

*Figure 5 - PLC Specifications*

The Next button on panel PLC Specifications will guide the user to choose one of the CoDeSys plugins (Generators) presented in the UNICOS Generators for SoMachine Schneider (Figure 6). As mentioned above the plugins are the CoDeSys Instance Generator, the CoDeSys Logic Generator, the WinCC OA Instance Generator for the SCADA code generation and optionally the Touch Panel Instance Generator. The
Expert User Generator provides a way to execute user scripts to generate specific additional files for purposes that are not covered by the previous plug-ins.

![Image of Expert User Generator interface]

**Figure 6 - Plugins**

For the Instance files generation, the option **CoDeSys Instance Generator** has to be chosen and the corresponding panel will appear, Figure 7. The location of the templates as well as the location of the output files is specified by default when the user is choosing where to save the new application (Figure 3). This location is presented here and the user can access the folders directly through the panel by the Open button next to each of the directories. Under those directories, there is a checkbox, giving the option for processing or not the semantic rules during the generation. A step lower in the panel, are located two checkboxes that provide the option to the user to generate or not the Communication and/or the I/O Commissioning file. The communication file is used for the communication between the PLC and the SCADA and should **always** be generated since in the topology file there is reference to this program.

In the “Post Process User Template” option, the user can execute an additional action for the Instance generator by executing the template declared in the field “Post Process User Template”.
Then follows the list of all the CPC objects existing in the current application (information coming from the specification.xml file). The user can either press the button to select them all or manually select the ones needed to be generated. It is important to notice here, that in the output file is also included the global variables definition. So a selection of only some of the objects for the generation, will give to the output only the global variable of those objects.

It is recommended to select all the objects for generation for the first time. After selecting the objects and the files to be generated the user should press the **Generate** button.

The generated files are located under the folder “...\UAB_Project\Output\CodesysCodeGenerator\”. The files in this folder will be the **plc_instances_file.xml**, the **communication_DS.xml** and the **IOCommissioning.xml**, if its generation is chosen to be performed.

In the generators panels there are buttons at the bottom of the window for navigating between generators.

![Instance Generator](image)

*Figure 7 - Instance Generator*

For the generation of the logic code, the **CoDeSys Logic Generator** should be selected next. The Figure 8 shows the panel dedicated to the logic generation. The
generated logic can be found in the folder “...\UAB_Project\Output\CodesysLogicGenerator” and includes the files `plc_logic_file.xml` and the `topology_file.xml`. It is mandatory for the first generation to generate all the logic (object types, masters and the logic sections), since inside the topology file there are references to the logic code. Not full generation of the logic will lead to errors in the PLC program. Later, you can partly generate some objects of the logic which you may have modified (e.g. with the User Templates).

The `Templates Development API` button will open a pdf file that gives guidelines for the creation of the User Specific Templates.

The generated files import into Somachine a number of separate small programs (POUs). The topology file indicates the execution order of these programs at its cycle.

Also here after selecting the objects for generation the user should press the `Generate` button.

![Image of Logic Generator interface]

*Figure 8 - Logic Generator*
The third generator, the **WinCC OA Instance Generator**, is displayed below in the Figure 9. Here you should select all device types by clicking the *Select All* button and then the *Generate* button. The generated file for the WinCCOA importation, `wincc_oa_db_file.txt`, is located in the folder “...\UAB_Project\Output\WinCCOAInstanceGenerator\”.

As you can see in the figure below when there is a change in the specifications file there is a warning triangle appearing in the panel. The user can modify the specifications file by *Edit specs.* button and reload it with *Reload Specs.* in order to take into account the new changes. The above apply for all the generators.

![WinCC OA generator](image)

*Figure 9 - WinCC OA generator*

The forth generator is optional and should be used only if a touch panel will locally be used for the application. The target platform should be **Magelis** and all objects should
be selected to generate all the instances for the touch panel by pressing the Generate button. (Figure 10)

![Figure 10 - Touch Panel Generator](image)

UAB provides a way to report to the user the process of the generation and its status. The UAB user report window shown in Figure 11 is a very helpful tool to detect any error that may happen and the reason why it occurred. The various messages can be hidden or displayed according to the user’s choice for a specific category of messages. The categories are: Severe, Warning, Information, Configuration, Fine and Debug. To select or not a message category the buttons are displayed in the top of the Log window. Additionally there is a button for clearing the report data and one for copying to clipboard.
When a generation runs and finishes with no severe errors then in the report will appear the following messages according to the generation that was performed:

“The exit status of the CodesysCodeGenerator plug-in is SUCCESS”

“The exit status of the CodesysLogicGenerator plug-in is SUCCESS”

“The exit status of the WinCCOAInstanceGenerator plug-in is SUCCESS”

### 3.2.4 LOG

For every generation performed there is a log file created that contains information of the generation process and any error messages. This file is by default saved in the “...\UAB_Project\Log\” folder. In the same folder there is UABLogging.html file with all the information displayed in the user report window during the generation. For every new generation the content of this file will be overwitten with new information.

### 3.2.5 TEMPLATES

The templates used by the generators are inside the folder “<user project folder>\Resources”. They are divided in several folders:

The following folders are the ones containing the templates used for the generation of code of the importation files:

- **CodesysInstanceGenerator**: The templates are used to generate instances for the PLC program (Global, Instance, Post Process).

- **CodesysLogicGenerator**: The templates are used to generate logic for the PLC program (Common, ST, User Specific, Topology, Post Process).

- **WinCCOAInstanceGenerator**: The templates are used to generate the WinCC OA importation file (Global, Type, Post Process).
- TouchPanelGenerator: The templates are used to generate the importation file for the touch panels (Magelis, TiaPortal, WinCCFlex, Post Process)

Furthermore used in the generation:

- Device types: The templates used for device types contain general rules and description of the objects.
- SemanticCheckRules: The templates used for the semantic rules analyse the specifications file against the existing constraints (if applicable) to find out if there is any semantic error. These rules take into account the existing rules into PLC and WinCC OA.
- Upgrade: The templates are used to upgrade projects to the latest version of CPC6

In order to make your CPC application you will only need to make your User Specific Templates and then place them inside the CodesysLogicGenerator\Rules\UserSpecific folder. Also you should add a reference to them in the specification file under Custom Logic Section.

To know more about the usage of User Templates you can check the specific document [5].

### 3.3 CREATING THE PLC PROJECT

#### 3.3.1 BASELINE

First, the user should open the **Baseline** project. This is a Somachine project which contains all the **objects of UNICOS CPC library**. The project is placed under UAB Project folder and more specifically under UAB_Project\Baseline\ucpc-softplc-codesys\SCHNEIDER_SoMachine.

#### 3.3.2 HARDWARE CONFIGURATION

After opening the project the user can start configuring the hardware according to the application. The project is predefined for the Controller TM258LF42DT but this can be changed in the **Configuration** tab as well as more I/O modules can be added to the four existing ones (Figure 12). (e.g. SAI2PH, SAO2L, SAI2L).
3.3.3 ETHERNET CONFIGURATION

In the Configuration tab there is an image of the hardware/controller selected. By right-clicking on the image a pop up list is giving the Edit Device Parameters option (double-clicking on the image will give the same result). The next panel appearing is for the configuration of the communication parameters. For the Ethernet parameters configuration, the user has to click on the option Communication-> Ethernet-> Physical Settings in the MyController list appearing in the left side of the editor. Then on the right part of the editor is going to appear the interface, where all the Ethernet parameters should be put (Figure 13).

Here, it should be taken into account that the parameters given in the Ethernet configuration in the SoMachine have to match with the Ethernet parameters in the PLC Specifications panel, in the Figure 5.
In order to be able to connect with the PLC the developing PC used for the application should be in the same sub-network with the PLC. SoMachine is using a protocol feature (Broadcast IP) in order to detect and connect to a PLC, that is not allowed to pass though CERN backbone routers. After the physical connection is established the PLC can be detected through the Somachine software.

In Configuration tab double-click on the PLC image and the panel of figure 14 will appear. If there is no gateway choose to Add gateway.. Note that CODESYS Gateway SysTray should be started which is usually automatically done when the gateway is added. If it is not, you can manually start it. The icon should be on the right side of your desktop toolbar. If the gateway is started you will see this icon in your toolbar and in Somachine a green circle will be near the gateway.

Then you can scan the network to find your device. A list of the connected devices found will appear under the gateway. Double-click on the device and then press Alt+F to activate the communication.
3.3.5 PREPARATION FOR THE FIRST TIME

Before importing the generated files the user will probably need to customize the Somachine menu in order to be able to see the Import PLCOpen XML order. This is a one-time step.

In the Program tab select Tools->Customize... you can customize the menu. In this new pop-up window you can add the Import PLCOpen XML order which is in the category Objects and you can preferably put it under the Project tab of the main menu as shown in figure 15.
3.4 IMPORTATION OF THE GENERATED FILES INTO SOMACHINE PROJECT

Now the user is ready to import the generated files from UAB. Preferably the user can create two folders (or as many as needed), one for the instances and one for the logic in order to have a more easy to navigate structure. Just right-click on the Application and select Add Folder..

Then the user can select the appropriate folder where the files will be imported and from the Project menu can select the ImportPLCOpenXML order as shown in Figure 16.

Figure 15 - Customize SoMachine menu to have ImportPLCOpenXML order

Figure 16 - Somachine importation
A pop up window is appearing for the user to browse to the location of the generators’ output files and import them. (Figure 17)

![Figure 17 - Browse the importation file](image)

After choosing the importation file a new window will pop-up showing all the POU (Programmable Organization Units) which will be imported (Programs, GVL etc.). There the user can select all or some of them to be imported. All the files should be imported the first time and only in case of changes the user can choose to import some of them. (Figure 18) So the user should select all and press OK. In the additional information tab can be seen the information coming from the File and Content header of the file.

Unfortunately the SFC language in the form of PLCOpen XML is not currently supported for Somachine V3.0. The user should uncheck the files with an _SL ending so that they are not imported and then create the stepper logic manually or paste it from another Somachine project if it exists. If the _SL files are imported a warning will appear “Implementation language is currently not supported for <body> of <pou> element. The implementation part will be empty ST.” in the messages window (Figure 19) and an empty program in ST language will be imported.
Once the importation has finished, a list of the imported objects is created in the Messages window inside the editor, or in case of any errors occurring during the importation, the respective messages will appear there as well (Figure 19).

In case of re-importation of some of the programs in the project, it is appearing a dialog window, for the user to decide if he is going to rename, skip or overwrite the object(s). See Figure 20.
This procedure should be followed for all the four generated files that come from the Instance and the Logic Codesys generator, at least for the first time you create the application. The instances and communication file can be imported from “...\UAB_Project\Output\CodesysCodeGenerator\” and the logic and topology file from “...\UAB_Project\Output\CodesysLogicGenerator\”.

The topology file imports a program called Main which sets the execution order of the programs that constitute the PLC code. In order to be taken into account this execution order, this file needs to be added in the Task Configuration as shown in figure 21.

![Figure 21 - Add Topology file in Task Configuration](image)

### 3.5 PLC CODE COMPILATION

After all the files have been imported, there are present in the project tree all the different programs for all the objects instances that were specified in the specifications file and also the logic for all the field objects and PCOs. The sections IL, CL, GL, TL, SL and DL of the PCO and the DL of the field objects can be modified by the user. These modifications can then be used to create the respective User Templates. So the user can fill the specific to the application logic into the above program units and then continue with the compilation. In order to have this specific logic automatically generated so that the user can get all the application’s code directly from UAB, the extra code and variables have to be included in the User Specific Templates and then point from the spec file to these Templates.

The meanings of the functions filled automatically with the generator are:

- **BL**: Basic Logic, contains the basic logic of the PCO. It is not necessary to fill it.
- **CDOL**: Common Dependant Object Logic, calculates the auto requests of all dependant objects and the alarm acknowledgment. It is not necessary to fill it.

It is necessary to fill some of the functions with the specific logic of the project:

- **GL**: Global Logic, allows defining the global logic of the PCO.
- **IL**: Interlock Logic, alarm instantiation and calculation of the Interlocks of the PCO (Start, Temporal, Stop Interlock).
- **SL**: Sequencer Logic, sequential behaviour of the PCO (generally in SFC).
- **TL**: Transition Logic, contains all the calculations of the transitions between the steps in the SL.
- **CL**: Configuration Logic, calculation of the On status and Off status conditions of the PCO used mainly for animation purposes. In addition it computes the controlled stop finished conditions.
- **DL**: Dependent Logic of every object linked of the PCO. This function contains the behaviour of these objects according to the status (e.g. stepper states). These objects can be: PCO, Controller, On/Off, Analog, AnaDig, and AnaDO. All these objects have a Dependent Logic function associated to their PCO master.

*Note*: PCO can be its own master.

Once all the necessary files have been imported the source code should be compiled by the **Build** button, Figure 22. If there are no errors after the compilation, the user can Login to the PLC, download and start the program.

![Build Login](image)

*Figure 22 – Build*

### 3.6 CREATION OF WINCC OA PROJECT

To create a new WinCC OA project for the supervision, you can check the *Procedure WinCC OA-CPC Application* [6]. You will need to import the generated file from WinCC OA generator (database).

You can also design the panels for the supervision in the project and link them to the objects of the PLC/SCADA project.
3.7 USAGE OF TOUCH PANELS

If local touch panels will be used for the application then you will need the generated file from the UAB TouchPanelGenerator plug-in to import all the object instances. To know more about how to create touch panel applications take a look at the procedure Procedure Touch Panel Magelis [7].

4. SUPPORT

Please address your questions to mailto:IceControls.Support@cern.ch

5. REFERENCES

[1] Opening the floor to PLCs and IPCs: CoDeSys in UNICOS
https://webgrn0001.llnl.gov/prepress/MOPPC033.PDF
[2] Procedure Twincat-UCPC Application
https://edms.cern.ch/file/1228441/1.6.0/Procedure_Twincat-UCPC_Application.pdf
https://edms.cern.ch/file/1228441/1.5.0/Specs_doc.xml
https://j2eeps.cern.ch/wikis/download/attachments/47972428/CoDeSysMiddleWare.pdf
?version=1&modificationDate=1386694157630#page=21
[5] UCPC6 User Template
https://edms.cern.ch/file/1228441/1.5.0/UCPC6_UserTemplate.pdf
[6] Procedure WinCC OA-CPC Application
https://edms.cern.ch/file/1228441/1.4.0/Procedure_PVSS-UCPC_Application.pdf
[7] Procedure Touch Panel Magelis