Procedure

CREATING AND DESIGNING A LOCAL PANEL FOR A UNICOS APPLICATION

Abstract

This document explains how to create an application for a local panel in an UNICOS project.

DOCUMENT PREPARED BY: IZQUIERDO ROSAS, Silvia

[EN-ICE]
<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>2011-11-14</td>
<td>All</td>
<td>First Version (Silvia Izquierdo Rosas)</td>
</tr>
<tr>
<td>v1.1</td>
<td>2012-03-26</td>
<td></td>
<td>Changes about generator</td>
</tr>
<tr>
<td>v1.2</td>
<td>2012-05-30</td>
<td>9</td>
<td>Constraints added</td>
</tr>
<tr>
<td>v1.3</td>
<td>2012-06-27</td>
<td>23</td>
<td>Word Status widget information added</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS

1. HARDWARE NEEDED ............................................................................................. 7
2. SOFTWARE NEEDED ............................................................................................. 8
3. CONSTRAINTS ..................................................................................................... 9
4. ADDING HMI STATION TO THE S7 PROJECT ......................................................... 10
5. CONNECTIONS ................................................................................................... 12
6. CREATE BACKGROUND ...................................................................................... 13
7. GENERATION OF TAGS/TEXT LISTS/ALARMS/SCRIPTS ........................................ 14
   7.1 TAGS ............................................................................................................ 15
   7.2 TEXT LISTS ................................................................................................... 16
   7.3 ALARMS ........................................................................................................ 17
   7.4 SCRIPTS ....................................................................................................... 17
8. UNICOS LIBRARY ............................................................................................... 18
9. PANELS DESIGN ................................................................................................. 19
   9.1 Analog Input/Output widget ............................................................................ 20
   9.2 PCO widget/faceplate ...................................................................................... 21
   9.3 Word Status widget ........................................................................................ 23
   9.4 Simulation ..................................................................................................... 24
   9.5 Layers .......................................................................................................... 25
10. DOWNLOADING THE APPLICATION ..................................................................... 26
   10.1 Transfer via MPI cable ................................................................................. 26
   10.2 Transfer via Ethernet ..................................................................................... 27
11. MISCELLANEOUS .............................................................................................. 29
   11.1 Screen navigation .......................................................................................... 29
   11.2 Template ....................................................................................................... 31
   11.3 Security settings ............................................................................................ 33
   11.4 Alarm list ...................................................................................................... 38
   11.5 Updating the library ...................................................................................... 39
   11.6 Modifying widgets/faceplates locally ............................................................. 39
   11.7 Alarms screen ................................................................................................ 41
   11.8 General advice ............................................................................................... 42
12. REQUIRED TAGS .............................................................................................. 43
   12.1 Analog alarm ................................................................................................ 43
   12.2 Analog Digital ................................................................................................ 43
   12.3 Analog Input .................................................................................................. 43
   12.4 Analog Input Real ........................................................................................... 43
   12.5 Analog .......................................................................................................... 44
   12.6 AnaDO .......................................................................................................... 44
   12.7 Analog Output ................................................................................................ 44
   12.8 Analog Output Real ........................................................................................ 45
   12.9 Analog Parameter ........................................................................................... 45
12.10 Analog Status ................................................................. 45
12.11 Digital Alarm ................................................................. 45
12.12 Digital Input ................................................................. 46
12.13 Digital Output ............................................................... 46
12.14 Digital Parameter .......................................................... 46
12.15 Local ............................................................................. 46
12.16 On Off .......................................................................... 46
12.17 Process Control Object .................................................. 46
12.18 Controller .................................................................... 47
12.19 Word Parameter ........................................................... 47
12.20 Word Status ................................................................. 48
13. EXAMPLE............................................................................. 49
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PC adapter</td>
</tr>
<tr>
<td>2</td>
<td>Power supply</td>
</tr>
<tr>
<td>3</td>
<td>Insert SIMATIC HMI Station</td>
</tr>
<tr>
<td>4</td>
<td>Integrate in STEP 7</td>
</tr>
<tr>
<td>5</td>
<td>Touch panel model</td>
</tr>
<tr>
<td>6</td>
<td>Changing touch panel type</td>
</tr>
<tr>
<td>7</td>
<td>MPI connection configuration</td>
</tr>
<tr>
<td>8</td>
<td>Ethernet connection configuration</td>
</tr>
<tr>
<td>9</td>
<td>Proflibus connection configuration for Siemens</td>
</tr>
<tr>
<td>10</td>
<td>New folder link</td>
</tr>
<tr>
<td>11</td>
<td>Opening UAB application project</td>
</tr>
<tr>
<td>12</td>
<td>UNICOS Generators window</td>
</tr>
<tr>
<td>13</td>
<td>WinCC Flexible Instance Generator</td>
</tr>
<tr>
<td>14</td>
<td>Importation of tags</td>
</tr>
<tr>
<td>15</td>
<td>Importation options</td>
</tr>
<tr>
<td>16</td>
<td>Tags folders</td>
</tr>
<tr>
<td>17</td>
<td>Script call</td>
</tr>
<tr>
<td>18</td>
<td>Importation</td>
</tr>
<tr>
<td>19</td>
<td>Opening UNICOS-CPC6 library</td>
</tr>
<tr>
<td>20</td>
<td>Drag and drop AI widget and choose the corresponding tags</td>
</tr>
<tr>
<td>21</td>
<td>Drag and drop AI faceplate and choose the corresponding tags</td>
</tr>
<tr>
<td>22</td>
<td>AI widget modification</td>
</tr>
<tr>
<td>23</td>
<td>PCO widget</td>
</tr>
<tr>
<td>24</td>
<td>Text lists imported</td>
</tr>
<tr>
<td>25</td>
<td>Configuring text list for PCO option modes</td>
</tr>
<tr>
<td>26</td>
<td>PCO widget</td>
</tr>
<tr>
<td>27</td>
<td>PCO faceplate</td>
</tr>
<tr>
<td>28</td>
<td>New text list on WS widget</td>
</tr>
<tr>
<td>29</td>
<td>Configuring text list for WS widget</td>
</tr>
<tr>
<td>30</td>
<td>Runtime system with simulator</td>
</tr>
<tr>
<td>31</td>
<td>Panel simulation</td>
</tr>
<tr>
<td>32</td>
<td>Layers</td>
</tr>
<tr>
<td>33</td>
<td>Transfer from WinCC flexible</td>
</tr>
<tr>
<td>34</td>
<td>Transfer the application to the touch panel</td>
</tr>
<tr>
<td>35</td>
<td>Set PG/PC Interface as “PC Adapter(MPI)”</td>
</tr>
<tr>
<td>36</td>
<td>PC Adapter (MPI)</td>
</tr>
<tr>
<td>37</td>
<td>Transfer of the application via Ethernet</td>
</tr>
<tr>
<td>38</td>
<td>Set PG/PC Interface as “TCP/IP -&gt; Broadcom NetXtreme Gig…”</td>
</tr>
<tr>
<td>39</td>
<td>Drag and drop screens</td>
</tr>
<tr>
<td>40</td>
<td>Screens hierarchy</td>
</tr>
<tr>
<td>41</td>
<td>Defining navigation</td>
</tr>
<tr>
<td>42</td>
<td>Navigation buttons</td>
</tr>
<tr>
<td>43</td>
<td>Template screen</td>
</tr>
<tr>
<td>44</td>
<td>Panel</td>
</tr>
<tr>
<td>45</td>
<td>Buttons in the template</td>
</tr>
<tr>
<td>46</td>
<td>ActivateScreen function</td>
</tr>
<tr>
<td>47</td>
<td>Users</td>
</tr>
<tr>
<td>48</td>
<td>Logoff time</td>
</tr>
<tr>
<td>49</td>
<td>Tags needed for login/logoff</td>
</tr>
<tr>
<td>50</td>
<td>Placing login/logoff buttons</td>
</tr>
</tbody>
</table>
Figure 51 - Writing text for buttons ................................................................. 34
Figure 52 - Selecting ShowLogonDialog function ........................................... 34
Figure 53 - Writing text on a text field ............................................................. 35
Figure 54 - Configuring I/O Field .................................................................... 35
Figure 55 - View of the login/logoff buttons and user display .......................... 35
Figure 56 - Scheduler ..................................................................................... 36
Figure 57 - Invisible button to disable objects .................................................. 36
Figure 58 - Enable invisible button ................................................................. 37
Figure 59 - Visibility of invisible button .......................................................... 37
Figure 60 - Runtime Security Settings ............................................................ 37
Figure 61 - Alarm list....................................................................................... 38
Figure 62 - Alarms importation ....................................................................... 38
Figure 63 - Updating objects .......................................................................... 39
Figure 64 - Changing the format type ............................................................. 39
Figure 65 - Widget after changing the format type ............................................ 39
Figure 66 - Widget after rearranging fields .................................................... 40
Figure 67 - Al 3decimals faceplate ................................................................. 40
Figure 68 - Device Settings ............................................................................ 41
Figure 69 - Example 1 of Alarms screen ......................................................... 42
Figure 70 - Example 2 of Alarms screen ........................................................ 42
Figure 71 - Main screen ................................................................................ 49
Figure 72 - Process ......................................................................................... 49
Figure 73 - Alarms screen 1 .......................................................................... 50
Figure 74 - Alarms screen 2 .......................................................................... 50
Figure 75 - Alarms screen 3 .......................................................................... 51
1. **HARDWARE NEEDED**

To create an application in a touch panel you need:

- 2 PROFIBUS cables (with their PROFIBUS connectors)
- 1 PC adapter USB (if you are transferring the application with MPI)
- 1 Power supply for the touch panel (if you are not connecting it directly to the lab rack)

One of the PROFIBUS cables is used to connect the touch panel with the PC (for transferring the application with MPI), and the other one is used to connect the touch panel with the PLC. The PC adapter USB is used with one of the PROFIBUS cables to connect the touch panel with the PC.

![Figure 1 - PC adapter](image1.png)

![Figure 2 - Power supply](image2.png)
2. SOFTWARE NEEDED

Apart from the Simatic S7, it is necessary to install WinCC flexible. The version used in this tutorial was WinCC flexible 2008 SP2.

Before getting started, make sure you have the following software installed:

1) Check you have access to:
   \cern.ch\dfs\Applications\Siemens\LicensedSoftware
   a. If you don't have access, send email to Icecontrols.support@cern.ch

2) Simatic Step7Professional
   a. Install latest version of Step7Professional software
   b. Go to:
      \cern.ch\dfs\Applications\Siemens\LicensedSoftware\Step7Professional
      c. For the license, go to:
         \cern.ch\dfs\Applications\Siemens\LicensedSoftware\Step7Professional\CERN_license
            i. pick correct version which you installed, and run setup.exe

3) WinCCFlexible
   a. Install latest version of WinCCFlexible
   b. Go:
      \cern.ch\dfs\Applications\Siemens\LicensedSoftware\WinCCFlexible\2008_SP2\Compact_Standard_Advanced and run setup.exe

4) UAB Bootstrap
   a. Install latest version of UAB Bootstrap
   b. Go:
      https://j2eeps.cern.ch/wikis/display/ENICECOL/UAB+Bootstrap
3. CONSTRAINTS

To be sure a touch panel is the best solution for your HMI, it’s better to check Siemens touch panel documentation.

These are some of the constraints for MP277 and MP377 panels.

Table 1 - Touch panel constraints

<table>
<thead>
<tr>
<th></th>
<th>MP277</th>
<th>MP377</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discrete alarms</td>
<td>4.000</td>
<td>4.000</td>
</tr>
<tr>
<td>Analog alarms</td>
<td>200</td>
<td>200</td>
</tr>
<tr>
<td>Tags</td>
<td>2.048</td>
<td>2.048</td>
</tr>
<tr>
<td>Text lists</td>
<td>500</td>
<td>500</td>
</tr>
<tr>
<td>Screens</td>
<td>500</td>
<td>500</td>
</tr>
<tr>
<td>Tags per screen</td>
<td>200</td>
<td>400</td>
</tr>
<tr>
<td>Complex objects per screen</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>VBScripts</td>
<td>50</td>
<td>100</td>
</tr>
</tbody>
</table>
4. ADDING HMI STATION TO THE S7 PROJECT

If you have the S7 project, to add an HMI station right click over the project and click the option “Insert New Object > SIMATIC HMI Station”.

![Figure 3 - Insert SIMATIC HMI Station](image)

Choose the touch panel that is going to be used. In this tutorial we will use a MP 277 10” Touch v1.1.3.0. Click OK.

You can also create the HMI project starting WinCC flexible and creating a new project. It is possible to integrate the hmi project in an existing S7 project by clicking on ‘Project > Integrate in STEP 7…’ (see next figure).

![Figure 4 - Integrate in STEP 7](image)
Open the HMI station using WinCC flexible Advanced. For the tutorial it was used WinCC flexible Advanced 2008 SP2.

It is also possible (and **recommended**) to use a project template. There are two templates available, one of them for a MP277 15” panel and the other one for a MP277 10” panel. These templates have already implemented the longin/logoff buttons. Open the template and save it with a different name in the desired location, then you can add it to your Simatic project like it is explained in this section. If these panels don’t match, it is possible to change the touch panel type by right clicking on it on the Project tree. See Figure 6.

![Figure 5 - Touch panel model](image)

You can find them in the following link:

https://j2eeps.cern.ch/wikis/display/EN/UNICOS-CPC+Documentation

For **Schneider** it is not necessary to change anything in the PLC project.
5. CONNECTIONS

To configure the connection with the PLC double click on 'Communication > Connections' in the project tree.

Add a new connection (double click on first line) and configure it as follows: choose the name "Connection_1"\(^1\), select the corresponding communication driver (usually SIMATIC S7 300/400 for Siemens and Modicon MODBUS TCP/IP for Schneider) and select your station if you are linked to the Simatic project (from drop-down menu, select ‘unconnected’ and you will see your PLC). Then configure the communication depending on the type (MPI/DP, Ethernet...). See Figure 7.

---

\(^1\) It is important you call it "Connection_1", if not it won’t fit with the generated file for tags.
To know more about the communication possibilities check the official Siemens panel documentation on the web:


If you want to use a Profibus DP connection do not forget to set the connection on the PLC side. MPI connection is usually configured by default.

You can take a look at this video for more information:

http://www.youtube.com/watch?v=9Hgy7S-HoCI

6. CREATE BACKGROUND

To add a background taken from pvss, you need to make the folder accessible as follows: in ‘Tools’ menu, right click and click on ‘New folder link...’ (see next figure), pick the desired folder where the images are stored. Now you can drag and drop the images inside the folder.

You can take a screenshot from the panel in PVSS or ask the project responsible to provide it.
The background color must be changed from “unSynopticBackground” to “{52,52,52}”. Widgets must be removed to leave space to place Touch Panel widgets.

7. **GENERATION OF TAGS/TEXT LISTS/ALARMS/SCRIPTS**

As mentioned in section 2, **UAB Bootstrap** is needed. Run UAB Bootstrap, choose **cpc-wizard** and open UAB project of the panel application if it is not already opened.

![Figure 11 - Opening UAB application project](image)

In the UNICOS Generators window choose “WinCC Flexible Instance Generator” and click on “Next”. See Figure 12.

![Figure 12 - UNICOS Generators window](image)

To generate the importation files (alarms, tags, scripts and text lists) of all the objects click on “Select All” and then click on “Generate”.

---

**Run cpc-wizard**

**Select option “Open Existing Application”**

**Select the path where the project is located**
You can find the generated files in “\UAB_Project\Output\WinCCFlexInstanceGenerator”. The files are:

- **winccFlex_alarm.csv**: this file is used to create alarm variables, useful if you are using an alarm list in the panel.
- **winccFlex_script.txt**: this text file contains the script used to assign names/units to the tags.
- **winccFlex_securitySettingsScript.txt**: this text file contains the script used for the security settings.
- **winccFlex_tags.csv**: this file contains all the tags definitions.
- **winccFlex_textLists.csv**: this file is used to import the text lists used for the PCO option modes.

### 7.1 TAGS

To import the tags click on ‘Project > Import/Export... > CSV import...’. In the window showed in Figure 14 select Tags and choose the directory where the tags file has been generated.

![Figure 14 - Importation of tags](image)

Make sure the option “Use folder delimiter” is marked in the Importation options, if not folders won’t be created. See the option window in Figure 15.
Like this, folders will be created dividing the tags by object type as you can see in Figure 21.

You can take a look at the required tags at the end of this document.

7.2 TEXT LISTS

To import them click on ‘Project > Import/Export... > CSV import...’. Select Text lists and choose the directory where the text lists file has been generated or saved.
7.3 **ALARMS**

To import them click on ‘Project > Import/Export... > CSV import...’. Select *Analog/discrete alarms* and choose the directory where the Alarms file has been generated or saved. This is just useful if you are using an ‘Alarms list’ in the panel (Enhanced objects > Alarm view).

7.4 **SCRIPTS**

Two scripts are generated, one of them is used for security issues, and the other one is used to assign names and units as text to the tags of type *string*.

To add a new script in the project go to the Project tree and double click on ‘Scripts > Add Script’. The new script will be opened automatically, open the text generated *(winccFlex_script.txt)* and copy paste the text in the script. To check the spelling right click anywhere on the script and click on ‘Check syntax’, if it is correct you will read in the Output “Syntax check successfully completed”. Do the same with the script called *winccFlex_securitySettingsScript.txt*.

It is possible to modify names, units and description in the script, following the limitations: 40 digits for names, 10 digits for units and 100 digits for descriptions.

It is necessary to make a call to the script from a panel. Go to the main panel and go to ‘Events > Loaded’. Select the scripts from *User Scripts* and add it.

![Figure 17 - Script call](image)

It is recommendable to use meaningful names for the scripts, such as “Naming_Units” and “Security_Settings”.

You can also import everything at the same time by choosing all the importation files at the same time.

![Figure 18 - Importation](image)
8. **UNICOS LIBRARY**

To import UNICOS-CPC6 library open *Tools (View > Tools)*. Go to *Library* and click on *Open*.

![Figure 19 - Opening UNICOS-CPC6 library](image)

You can find the library in your UAB project in the following generic path “...\UAB_Project \Baseline”, select it and click on Open (the library is called “ucpc-wincc-flexible-vx.x”).

Now you can start designing the panels.
9. PANELS DESIGN

It is possible to drag and drop widgets and/or faceplates, let’s add an Analog Input widget and faceplate to the panel “Screen_1”.

Drag and drop “AI widget – cpc6” from the library in the panel and choose the corresponding tags of the desired object.

![Figure 20 - Drag and drop AI widget and choose the corresponding tags](image)

Now drag and drop “AI faceplate – cpc6” and choose the corresponding tags.

Suggestions:

2. Under Properties > Misc, change name to “instanceName_widget” for widget and “instanceName_faceplate” for faceplate. Then you can easily find the widget and faceplate using ‘Find’ (Ctrl+F)

3. To make it quicker you can copy/paste the name of the object (“Name.”) in each field of the faceplate/widget, and just choose the desired tag

Note: every widget must have its corresponding faceplate, except for the ones with no faceplate (Analog Status, Word Status)
Figure 21 - Drag and drop AI faceplate and choose the corresponding tags

9.1 Analog Input/Output widget

To change the format of the Analog input or an Analog output you can modify the widget by right click and clicking on ‘Edit faceplate’. Now click on the IO Field used to display PosSt and change Format pattern as you want (the ‘s’ at the beginning of the format means signed)\(^2\).

Figure 22 - AI widget modification

\(^2\) Advice: copy/paste the widget and rename it as ‘AI x.digits widget’
9.2 PCO widget/faceplate

PCO objects are unique, because it is necessary to configure the text lists for the option modes. It is necessary to copy the PCO widget from the library and copy it in the project library, and then change its name to our PCO name.

![Figure 23 - PCO widget](image)

Text lists will be generated with **WinCC Flexible Instance Generator**. To import text lists check section 7.2.

Text lists will be accessible in the Project tree in ‘Text and Graphics Lists > Text Lists’. Right click in the PCO widget in the project library and click in *Edit faceplate*. Copy the text list from the ones imported (to select it click on the square at left, see Figure 24) and paste it in the widget *Text list editor* (Figure 25).

![Figure 24 - Text lists imported](image)
Now click on the *IO Field* in the widget (see next figure) and select the text list just defined in the field *Text list*.

![Figure 25 - Configuring text list for PCO option modes](image)

Do the same with PCO faceplate for the *IO Fields* marked in Figure 27.

![Figure 26 - PCO widget](image)

![Figure 27 - PCO faceplate](image)

3 **NOTE:** If there is not a text list for the PCO, there will be a "0" instead of the PCO mode name in the screen, so the best option is to remove the IO Field from the widget. The solution for the faceplate is to place a square over the IO Field.
9.3 Word Status widget

If a Word Status with a pattern is used in the application, the widget used for this object should be “WS_text widget – cpc6”. The pattern must be created as a text list in the widget, so as happens with PCOs, one widget per WS is needed in this case.

Copy/paste the widget in the library project with a different name, and then edit the widget with right over the widget and clicking on “Edit faceplate”. To configure the text list click on “Text list” on the properties of the Symbolic IO field, and then click on “New”, see Figure 28.

![Figure 28 - New text list on WS widget](image)

Now give a name to the text list and configure it on the “Text list editor” on the “Faceplate configuration”. You can see an example in Figure 29.

![Figure 29 - Configuring text list for WS widget](image)
9.4 Simulation

We can simulate the panel by clicking in the button that can be seen in the next figure.

Figure 30 - Runtime system with simulator

It is possible to simulate the variable values with the Runtime Simulator table (see next figure).

Figure 31 - Panel simulation
9.5 Layers

It is very useful to use layers to hide the faceplates, it is possible to put the faceplates in different layers and then activate/deactivate them.

Figure 32 Layers
10. **DOWNLOADING THE APPLICATION**

10.1 Transfer via MPI cable

To download the application to the touch panel first you need to have the touch panel connected to the PC with the PROFIBUS cable and the PC adapter.

The configuration of the transfer has to be the same in the touch panel and in the software. To configure it in the touch panel go to ‘Control panel’ and double click on ‘Transfer’. Choose “MPI/Profibus/S7-Ethernet” and click on ‘Advanced’. Choose MPI and click on ‘Properties…’. Now choose the number of the station address, we will choose number 1 for the tutorial. The rest of the parameters are the ones set by default Time-out: 10 s, Transmission Rate: 187,5 kbits/s, Highest Station: 31, Panel is the only master on the bus. Click OK and close all the windows.

Now in WinCC flexible click on the button showed in next figure to start the transfer.

![Figure 33 - Transfer from WinCC flexible](image)

The station address must match with the one chosen in the panel configuration. Click on ‘Transfer’.

![Figure 34 - Transfer the application to the touch panel](image)

On the panel click on the button ‘Transfer’ from the desktop.

Make sure the MPI Adapter has been selected in SIMATIC Manager, to change it go to SIMATIC Manager and click on ‘Options > Set PG/PC Interface…’ and select “PC Adapter (MPI)”. 
You can also check it on the bottom of the SIMATIC Manager window, see next figure.

![Figure 36 - PC Adapter (MPI)](image)

10.2 Transfer via Ethernet

If the Touch panel is connected to the network you can download the application via Ethernet.

The configuration of the transfer has to be the same in the touch panel and in the software. To configure it in the touch panel go to ‘Control panel’ and double click on ‘Transfer’. Choose “Ethernet”.

Now in WinCC flexible click on “Transfer”, choose the mode “Ethernet” and write the IP address of the touch panel. Then click on the button “Transfer”.

![Figure 35 - Set PG/PC Interface as “PC Adapter(MPI)”](image)
Make sure the TCP/IP mode has been selected in SIMATIC Manager, to change it go to SIMATIC Manager and click on ‘Options > Set PG/PC Interface...’ and select “TCP/IP -> Broadcom NetXtreme Gig...”.

Figure 38 – Set PG/PC Interface as “TCP/IP -> Broadcom NetXtreme Gig...”
11. MISCELLANEOUS

11.1 Screen navigation

Useful if you have several screens.

To configure the screen navigation go to ‘Device Settings > Screen Navigation’ in the Project tree.

Drag and drop your screens to the Screen Navigation window and build the hierarchy of the project.

![Drag and drop screens](image)

Figure 39 - Drag and drop screens

![Screens hierarchy](image)

Figure 40. Screens hierarchy

To define navigation link the screens by clicking and stretching the arrow from the number on the top left corner of a screen to another screen.

![Defining navigation](image)

Figure 41 - Defining navigation

Doing this you will also get the configuration of the buttons in the panels.
Figure 42 - Navigation buttons
11.2 Template

It is possible to modify the template to have for example the project name in all the panels.

Figure 43 - Template screen

Figure 44 - Panel

You can also add buttons to the template instead of using the navigation buttons, see next figure. Take the buttons from the Simple objects in the Tools.
Then in the *Events* of the button add the function “ActivateScreen” and choose the desired screen you want to activate with that button, see next figure.

![Figure 45 - Buttons in the template](image)

Figure 45 - Buttons in the template

![Figure 46 - ActivateScreen function](image)

Figure 46 - ActivateScreen function
11.3 Security settings

To configure security settings first you have to set users. In the Project tree open ‘Runtime User Administration > Users’ with double click. Add the desired user name and its password with double click in the first line. Choose the rights of the user, normally it should be in the group “Users”. See next figure.

![Figure 47 - Users](image)

It is also possible to configure the logoff time as you can see in next figure.

![Figure 48 - Logoff time](image)

We need to create two tags for the login/logoff, in the Project tree double click on Tags, and double click on first line to declare them. The tags are: userName (internal tag, String) and userLogged (internal tag, Int). See next figure.

![Figure 49 - Tags needed for login/logoff](image)

We will add a script to check if the user has logged in. The name of the script is “userLoggedCheck”, and the code is the following:

```vbnet
If userName = "" Then
    userLogged=0
Else
    userLogged=1
End If
```

Now, to have the Login/logoff buttons accessible from all the screens we are going to place them in the template. Open the template and drag and drop two buttons from ‘Tools > Simple Objects’ and place them in the desired place, we will place them in the right top corner, leaving some space to display the user name, as you can see in next figure.
Configuration of login button: in ‘Properties > general’, write ‘Login’ in the field ‘Text OFF’.

Configuration of logoff button: in ‘Properties > general’, write ‘Log off in the field ‘Text OFF’. On ‘Properties > Events > Press’, choose the function ‘Logoff’, which will log off the user pressing the button. See previous figures.

We will display the name of the user logged in all screens by adding a TextField and a IO Field to the template. In the TextField (Tools > Simple Objects) we write ‘User:’ in ‘Properties > General’. It is also possible to change the font of the text in ‘Properties > Text’ and the color in ‘Properties > Appearance’.
You can find the *IO Field* in ‘Tools > Simple Objects’. Drag and drop it and choose ‘Output’ as ‘Mode’, ‘String’ as ‘Format type’ and the created tag “userName” as ‘Process tag’.

To run the script and get the user name we will use the Scheduler. Open it in the Project tree ‘Device Settings > Scheduler’. Double click on the first line to add a job. Choose a name for it (“Login” for example). In the ‘Event’ field choose ‘Change user’. In the Function list add function `GetUserName` and choose tag “userName” as Tag(Out). Now add function `userLoggedCheck`. See next figure.
To disable objects in the screens you can place an invisible button over them. Take the button from ‘Tools > Simple Objects’. In the ‘Button mode’ choose ‘Invisible’, place it in layer 31 so it is over everything else, and make as big as you need to cover all the objects except for the login/logoff buttons.

In the Project tree ‘Runtime User Administration > Runtime Security Settings’ you can set the configuration of the password. See next figure.
11.4 Alarm list

It is possible to add an alarm list as the one showed in next figure.

![Alarm list](image)

Figure 61 - Alarm list

You can find it in ‘Tools > Enhanced Objects’, its name is ‘Alarm view’.

Alarms will be generated with **Touch panel instance generator** (this tool is still under development, for more information contact Adrien Chiron [Adrien.Chiron@cern.ch](mailto:Adrien.Chiron@cern.ch) or William Booth [William.booth@cern.ch](mailto:William.booth@cern.ch)). To import alarms as Discrete Alarms click on ‘Project > Import/Export... > CSV Import...’ and choose the file ‘Alarms.csv’.

![Alarms importation](image)

Figure 62. Alarms importation

After the importation you can find them in the Project tree ‘Alarm Management > Discrete Alarms’.

It is possible to configure this Alarm list by navigating through its properties.
11.5 Updating the library

If a new version of the library is released while you are creating your application it is possible to update the objects. You can copy the desired widgets/faceplates from the library and paste them in the project library. You will get a window like the one in next figure, choose the option 'Update existing type' and click Ok.

![Figure 63 - Updating objects](image)

11.6 Modifying widgets/faceplates locally

This can be very useful for objects with Analog input/outputs.

As an example we will explain how to change the number of digits in an Analog input widget. Go to the project library, copy/paste AI widget and rename it.

For the example we will rename the widget as “AI 3decimals”. After changing the name open the faceplate (right click > edit faceplate).

Now click on the IO field used to show the PosSt information and change the *Format pattern* to the desired one, in this case “s99.999”, so it is signed with two digits for the integer part and three digits for the decimal part.

![Figure 64 - Changing the format type](image)

This changes the size of the IO field and the widget looks like next figure.

![Figure 65 - Widget after changing the format type](image)
So it is necessary to rearrange the fields to make it look like in next figure. For this just move the IO field used for units so it is not behind the PosSt field, and also move the letters F and h so they get placed in the bottom right corner of the PosSt field.

![Figure 66 - Widget after rearranging fields](image)

Do not forget to make the same with the AI faceplate so they are consistent.

![Figure 67 - AI 3decimals faceplate](image)
11.7 Alarms screen

The usual structure of a touch panel application is to place an Alarm list in the main screen, or choose as main screen the process view. It is possible to configure the main screen in the Project tree in ‘Device settings > Device settings’ (see next figure).

![Device Settings](image)

Figure 68 - Device Settings

Then there is usually a screen (or several) with the alarms in the UNICOS project. For this alarm screens it might be useful to use the special widgets called ‘AAtext widget – cpc6’ and ‘DAtext widget – cpc6’ (this widgets are available in the CV library). This widget has the name and a description (taken from the spec) of the object included. The name and the description are assigned in the script generated with the Touch panel instance generator (this tool is still under development, for more information contact Adrien Chiron Adrien.Chiron@cern.ch or William Booth William.booth@cern.ch), and they can be changed with a maximum of 40 digits for the name and 100 digits for the description. The size and appearance of the strings can be changed in the widget if needed (for more information about modifying a widget check section 11.6).
11.8 General advice

- Not placing things out of the screen, they won’t be displayed
- Save often!!! The WinCC software is prone to crashing and you can lose all your work
- Use the buttons showed in next figure, they are very useful
12. **REQUIRED TAGS**

12.1 **Analog alarm**
- AlarmName.visible (internal tag, bool)
- AlarmName.name (internal tag, string)
- AlarmName.stsReg01
- AlarmName.stsReg02
- AlarmName.manReg01

12.2 **Analog Digital**
- AnaDigName.visible (internal tag, bool)
- AnaDigName.setValue (internal tag, bool)
- AnaDigName.name (internal tag, string)
- AnaDigName.unit (internal tag, string)
- AnaDigName.stsReg01
- AnaDigName.stsReg02
- AnaDigName.manReg01
- AnaDigName.PosSt
- AnaDigName.MPosR

12.3 **Analog Input**
- AIname.visible (internal tag, bool)
- AIname.name (internal tag, string)
- AIname.unit (internal tag, string)
- AIname.setValue (internal tag, bool)
- AIname.stsReg01
- AIname.manReg01
- AIname.PosSt
- AIname.MPosR

12.4 **Analog Input Real**
- AIRname.visible (internal tag, bool)
- AIRname.name (internal tag, string)
- AIRname.unit (internal tag, string)
• AIRname.setValue (internal tag, bool)
• AIRname.stsReg01
• AIRname.manReg01
• AIRname.PosSt
• AIRname.MPosR

12.5 Analog
• AnaName.visible (internal tag, bool)
• AnaName.name (internal tag, string)
• AnaName.unit (internal tag, string)
• AnaName.setValue (internal tag, bool)
• AnaName.stsReg01
• AnaName.stsReg02
• AnaName.manReg01
• AnaName.PosSt
• AnaName.MPosR

12.6 AnaDO
• AnaDOname.visible (internal tag, bool)
• AnaDOname.name (internal tag, string)
• AnaDOname.setValue (internal tag, bool)
• AnaDOname.stsReg01
• AnaDOname.stsReg02
• AnaDOname.manReg01
• AnaDOname.PosSt
• AnaDOname.MPosR

12.7 Analog Output
• AOname.visible (internal tag, bool)
• AOname.name (internal tag, string)
• AOname.unit (internal tag, string)
• AOname.setValue (internal tag, bool)
• AOname.stsReg01
• AOname.manReg01
• AOname.PosSt
• AOName.MPosR

12.8 Analog Output Real
• AORname.visible (internal tag, bool)
• AORname.name (internal tag, string)
• AORname.unit (internal tag, string)
• AORname.setValue (internal tag, bool)
• AORname.stsReg01
• AORname.manReg01
• AORname.PosSt
• AORname.MPosR

12.9 Analog Parameter
• APARname.visible (internal tag, bool)
• APARname.name (internal tag, string)
• APARname.unit (internal tag, string)
• APARname.setValue (internal tag, bool)
• APARname.manReg01
• APARname.PosSt
• APARname.MPosR
• APARname.MPosRSt

12.10 Analog Status
• ASname.unit (internal tag, string)
• ASname.PosSt

12.11 Digital Alarm
• DAname.visible (internal tag, bool)
• DAname.name (internal tag, string)
• DAname.stsReg01
• DAname.manReg01
12.12 Digital Input
- DIname.visible (internal tag, bool)
- DIname.name (internal tag, string)
- DIname.stsReg01
- DIname.manReg01

12.13 Digital Output
- DOname.visible (internal tag, bool)
- DOname.name (internal tag, string)
- DOname.stsReg01
- DOname.manReg01

12.14 Digital Parameter
- DPARname.visible (internal tag, bool)
- DPARname.setValue (internal tag, bool)
- DPARname.name (internal tag, string)
- DPARname.stsReg01
- DPARname.manReg01

12.15 Local
- LocalName.visible (internal tag, bool)
- LocalName.name (internal tag, string)
- LocalName.stsReg01

12.16 On Off
- OnOffName.visible (internal tag, bool)
- OnOffName.name (internal tag, string)
- OnOffName.stsReg01
- OnOffName.stsReg02
- OnOffName.manReg01

12.17 Process Control Object
- PCOname.visible (internal tag, bool)
- PCOname.name (internal tag, string)
- PCOname.setOpMo (internal tag, bool)
12.18 Controller

- PIDname.visible (internal tag, bool)
- PIDname.name (internal tag, string)
- PIDname.setSetPoint (internal tag, bool)
- PIDname.stsReg01
- PIDname.stsReg02
- PIDname.manReg01
- PIDname.manReg02
- PIDname.MV
- PIDname.ActSP
- PIDname.ActTi
- PIDname.ActKc
- PIDname.ActTd
- PIDname.ActTds
- PIDname.MSP

12.19 Word Parameter

- WPARname.visible (internal tag, bool)
- WPARname.name (internal tag, string)
- WPARname.unit (internal tag, string)
- WPARname.setValue (internal tag, bool)
- WPARname.PosSt
- WPARname.MPosRSt
- WPARname.manReg01
- WPARname.MPosR
12.20 Word Status

- WSname.unit (internal tag, string)
- WSname.PosSt
13. EXAMPLE

This is an example of a touch panel application.

Figure 71 - Main screen

Figure 72 - Process
A widget with name and description is available in the CV library: see section 8.
Figure 75 - Alarms screen 3