Action$_0$NET – NewDemo_1.0 Manual

Version an-2016.2

NewDemo_1.0 Manual

Dec, 2017

Copyright 2017©

Spin Engenharia de Automação Ltda.

All rights reserved.

No part of this document may be reproduced, copied, photocopied, distributed or altered without the prior written consent of Spin Engenharia de Automação Ltda.

NOTE

Action₀NET © is a registered trademark of
Spin Engenharia de Automação Ltda.

All other brand and product names are trademarks of their respective owners and / or companies.

Because of the continuous development of our products, the information contained herein is subject to change and/or modifications without notice. Spin is not responsible for typographical errors or interpretation of information contained herein; and/or damages caused/generated to third parties. The contents of this publication may be changed at any time without there being any obligation to notify any party involved; this does not imply, under any circumstances, in changes, claims, or extended warranty.

Caution! It indicates that the user should proceed exactly as described in this manual, otherwise he/she damage or wrongly set the software.

Tip. Indicates helpful, timely information for small problems.

It indicates that the user should proceed exactly as described in this manual, at the risk of shock or electrical discharge.

Summary

1. INTRODUCTION 1

1.1 Action$_0$NET – The Next Generation SCADA 1

1.2 Objetives of this manual 1

2. ACTION$_0$NET – STARTING THE PROJECT 2

2.1 Installing the app 2

2.2 Starting the app 2

2.3 Running the Project 3

2.4 Layout 6

2.4.1 Header 6

2.4.2 Footer 10

2.4.3 Body 10

2.4.4 Screens Layouts 11

2.5 DefaultNewProject 11

3. NAVIGATING THE PROJECT 12

3.1 MAIN_SCREEN 12

3.1.1 Graph 12

3.1.2 Button 1 – Wind Plants 12

3.1.3 Button 2 – Hydropower 12

3.1.4 Button 3 – Hydro Generating Units 13

3.1.5 Button 4 – Performance Report 13

3.1.6 Button 5 – Performance Report 13

3.2 SPIN_WIND 13

3.3 WIND_GENERATORS 14

3.4 SL_SUBSTATION 14

3.5 3D Model 15

3.6 SPIN_HIDROLOGY 16

3.7 HYDRO_GENERATORS 17

3.8 Performance Report 18

3.9 Control Panel 19

3.10 Area Alarm 19

3.11 Run DNP 3.0 Server Program 20

4. INFORMATION ON THE APPLICATION PARAMETRIZATION 21

4.1 Project Info 21

4.2 Edit 21

4.2.1 Templates 21

4.2.2 Assets 22

4.2.3 Devices 23

4.2.4 Alarms 24

4.2.5 History Tables 24

4.2.6 Datasets 25

4.3 Drawing Screens 25

4.3.1 Construction of Screens 25

4.3.2 Scripts Associated with Screens 28
4.4 Compiling and Running Projects 28
4.4.1 Build 28
4.4.2 Start Up & Test 29
4.4.3 Dictionaries 31
4.4.4 Extension 31
4.5 Scripts Simulating the Process 33
4.5.1 General Information 33
4.5.2 Simulation of the substation 33
4.5.3 Simulation of Wind Turbine 34
4.5.4 Simulation of Hydraulic Power 34
4.6 Sophisticated Screen Example 34

5. ANNEX 1 – AXON TEST DNP 3.0 36

5.1 Configuration of the DNP 3.0 Slave Simulator 36
5.2 I/O Points 39
5.2.1 Connection Definition 39
5.2.2 Creating Binary Input 39
5.2.3 Creating Analog Input 40
5.3 Running DNP3 Slave Simulator 40

1. INTRODUCTION

1.1 Action₀NET – The Next Generation SCADA

Action₀NET is a new SCADA (Supervisory Control and Data Acquisition) software which incorporates in its development state of the art programming language (.NET Framework from Microsoft) and modern layout techniques from Windows Presentation Foundation (WPF). It uses Windows Communication Foundation (WCF) – another modern and high-performance technology supported by modern Microsoft operating systems – for communication between its various modules.

Its setup application allows shared engineering (multiple users can develop projects in parallel), in local or remote servers, including cloud computing.

Communication modules, with open and standardized protocols, are an evolution of the protocols developed in ActionView, the SCADA previously developed by Spin, which was implemented in hundreds of substation facilities, power plants and control centers in Brazil.

1.2 Objetives of this manual

The purpose of this manual is to allow new authorized users to perform, understand and test the functionality of the demonstration project called NewDemo_1.0 in Action₀NET.

The authorized NewDemo is an Express version type, limited in 75 I/O Points. In this NewDemo there are one "DNP3.0" channel and a DNP3.0 Server program used to test the protocol (see annex 1).

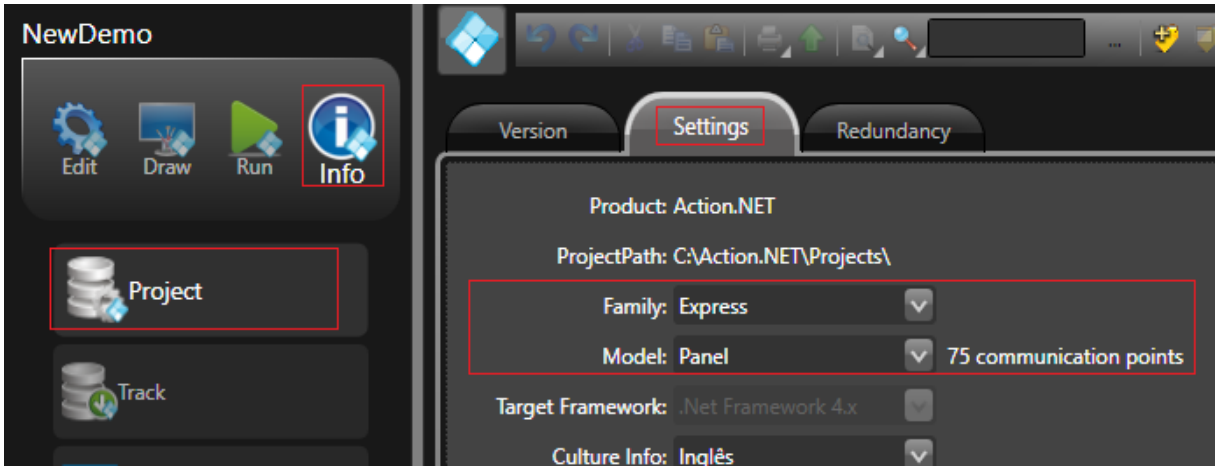


Figure 1 –NewDemo Version Family

For further details and other information about the Action₀NET, see the User Guide.

2. Action₀NET – STARTING THE PROJECT

2.1 Installing the app

Action₀NET 2016.2.58 installer is the file "an-2016.2.58.exe". Once executed, the entire package will be guided, a window will show the installation steps, the applications being installed and it will ask for confirmation of acceptance of the program's use of the license agreement. After installation, the program can be used immediately.

2.2 Starting the app

Once it has been installed on the computer Action₀NET a representative icon of the application will be created on the desktop. To start the app simply double-click on this icon:

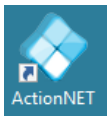


Figure 2 –Action₀NET setup icon

Opening the Project

After starting the application, you will see the system's Project Management window, shown below.

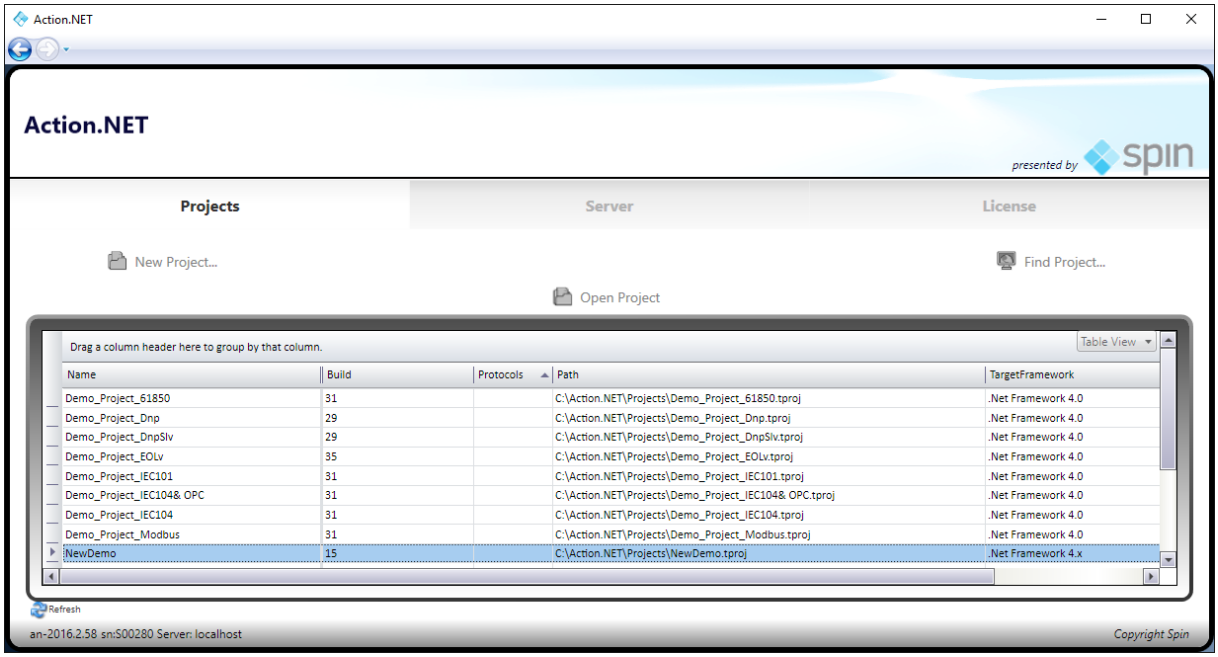


Figure 3 – Project Management Window

On this screen the projects ready to run are shown. In this NewDemo, only five projects demonstrations are configured in Action₀NET. This document will deal with **NewDemo** project that is not integrated with Action₀NET. In order to open the project, we must place the file "NewDemo.tproj" in the folder "C:\Action₀NET \Projects". Then we open the project through project management screen (Figure 3).

2.3 Running the Project

Once the double-click is performed on the selected Action₀NET project, the setup screen shown in Figure 4 opens. This info screen shows some information about the project, including the location where it is saved, the amount of communication points available in version, full of configured tags, etc.

Once you are using an express license NewDemo, on a non-licensed computer, the program presents some restrictions, allowing executing projects containing only 75 points of communication and ending the execution after one hour. One express application can be opened and run as many times as you want, but it will always run only 60 minutes.

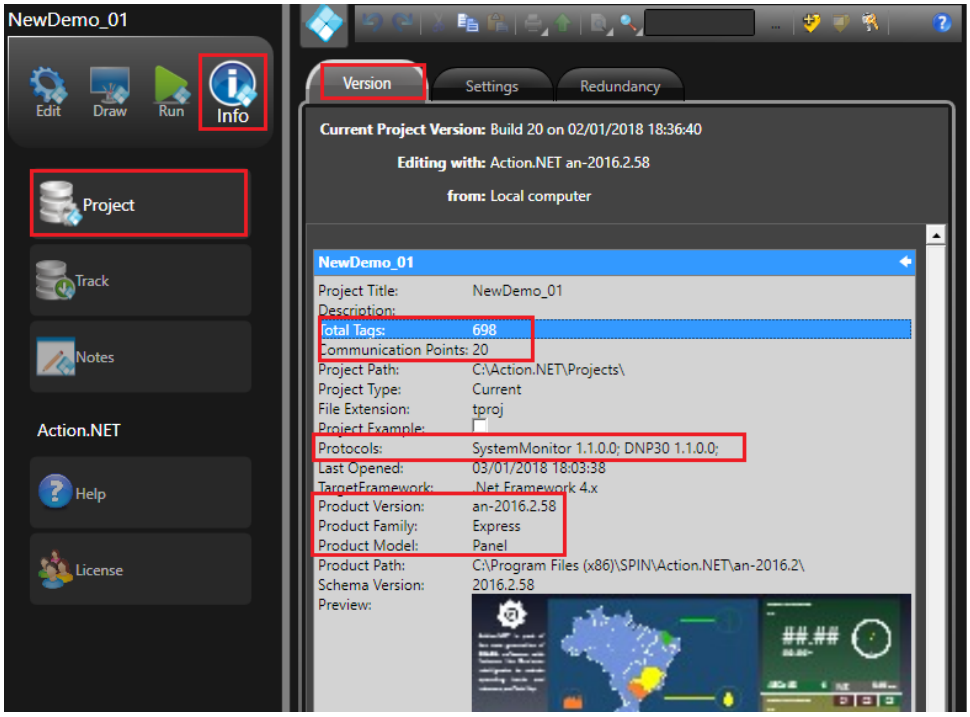


Figure 4 – Project configuration window

Attention: To distribute the demo free, without license, it is "express" type. This license, however, when used for development, has many limitations such as not support several template´s levels, bi - and tri-dimensional tables, etc.

The option adopted to show these features in an "express" version was to develop it using the "enterprise" license and after completing it, turn this license into "express".

To start project execution you need to click the Run menu, then on the Startup menu, and finally the Run Startup button, as shown in Figure 5. This area allows you to configure how the project will run in the final production mode. Means enabling the historian and Alarm recording functions for using actual databases defined in the project:

1) There are 4 main folders in the Action₀NET studio (Edit, Draw, Run and Info). To Run one application we are in the Run folder;
2) In the run folder, we are in the option Startup;
3) In the "Run Local Diagnostics Tool", we ticket the "Property Watch", so this tool will be executed automatically when the application Run (more detail of watch see item 4.4.2)

4) The "Run Modules" has 8 modules and only selected by ticket start running;

5) When pressed Run Startup, the application

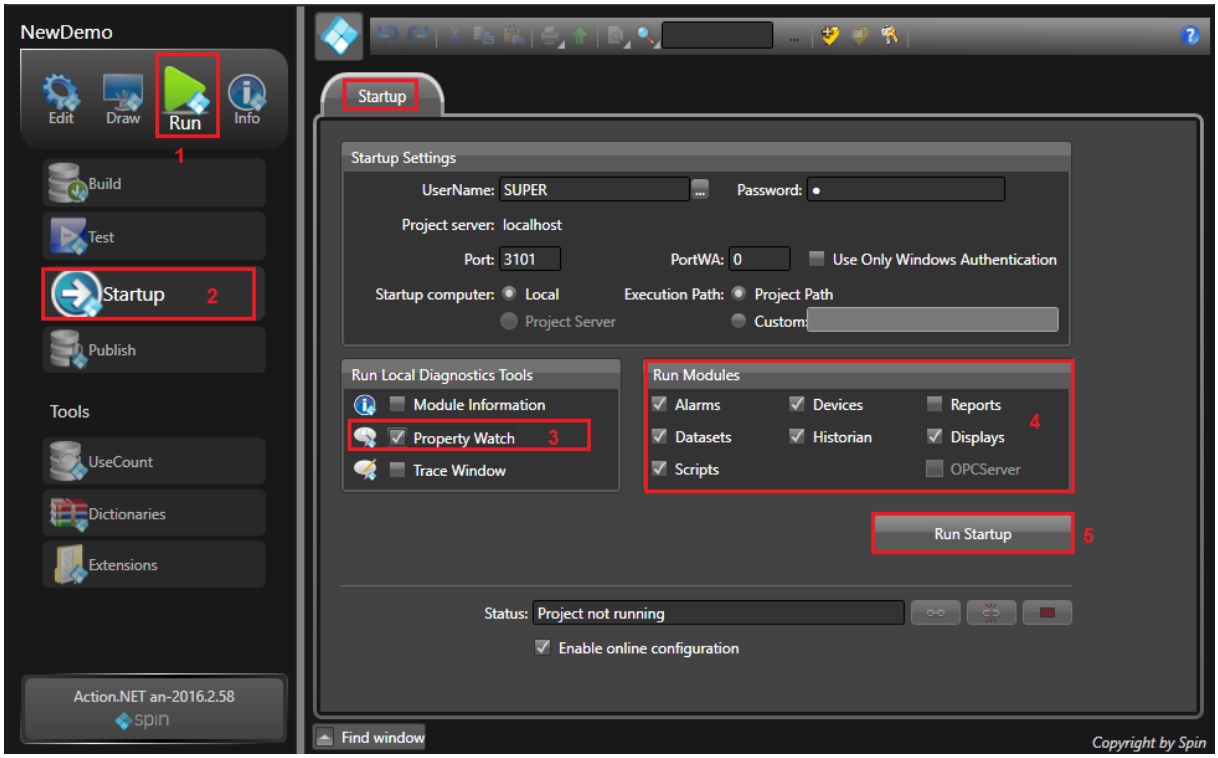


Figure 5 – Project configuration window

Once the Run Startup button is clicked, the project will be loaded and start running. Three windows are opened, the Action₀NET startup window, shown in Figure 6, the main screen shown in figure 8 and the property Watch shown in figure 49.

In Action₀NET Startup window are shown the modules that are running now. In the example, as you can see in the box Run Modules, we select the following modules to start running: Alarms, Datasets, Scripts, Devices, Historian and Display. We could, for example, start designing only the screens and the navigation between them, and to test these modules we should start only Displays and Scripts (all procedures associated to displays are defined on scripts associated to each display). In this window, you can also stop, pause and restart any module or the entire project clicking the Shutdown button.

To test functionalities or debug one or more modules you can select the tools Watch, Trace and Info. In NewDemo, we start the Watch tool discussed on item 4.4.2.

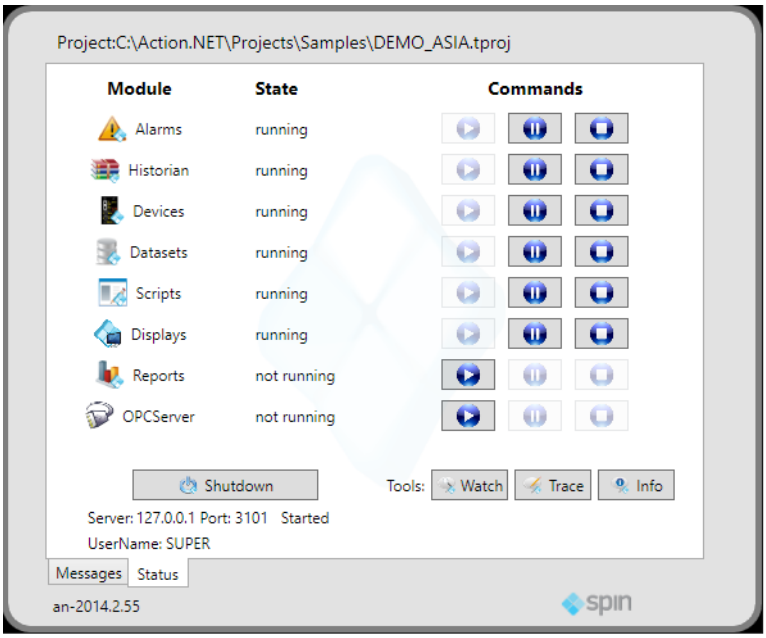


Figure 6 – Action₀NET startup window

The project execution window starts showing the main project screen and requesting the login (figure 9). For login, the user is "super" and the password is letter "s".

Once in the NewDemo there are scripts running and simulating tags values, you need to silence the horn sometimes, when one alarm is simulated. To silence the horn you must click mute button on the top of the screen (header).

Figure 7 – Mute button

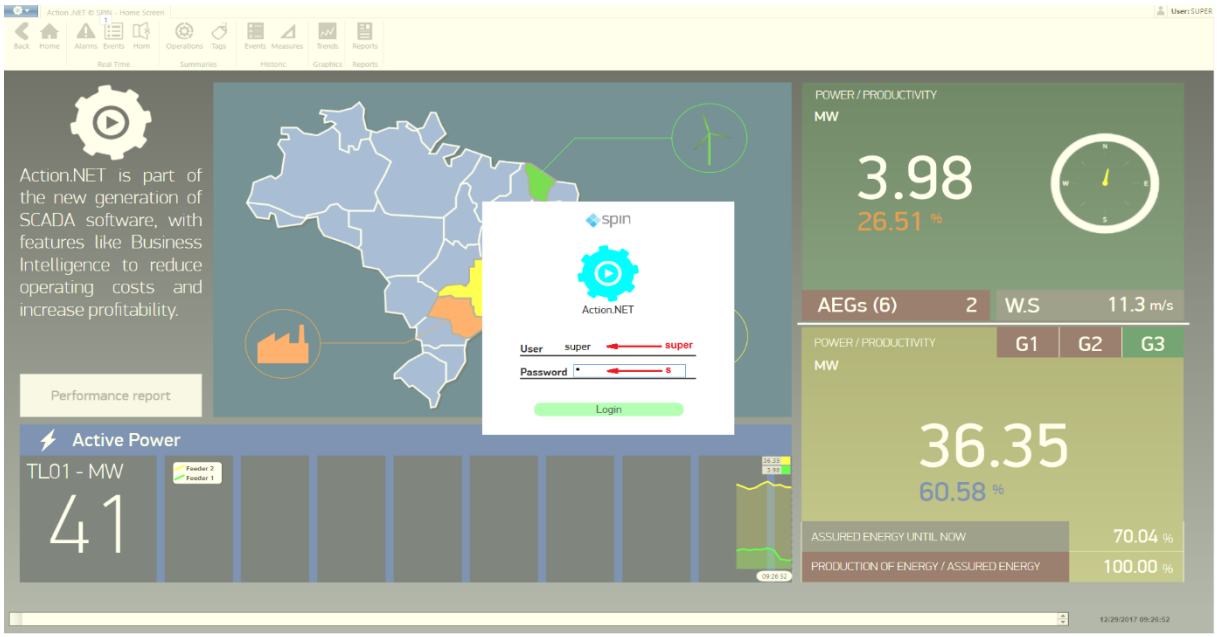


Figure 8 – Project execution window

2.4 Layout

At project NewDemo, the screens present tree distinct areas: Header, Footer and Body.

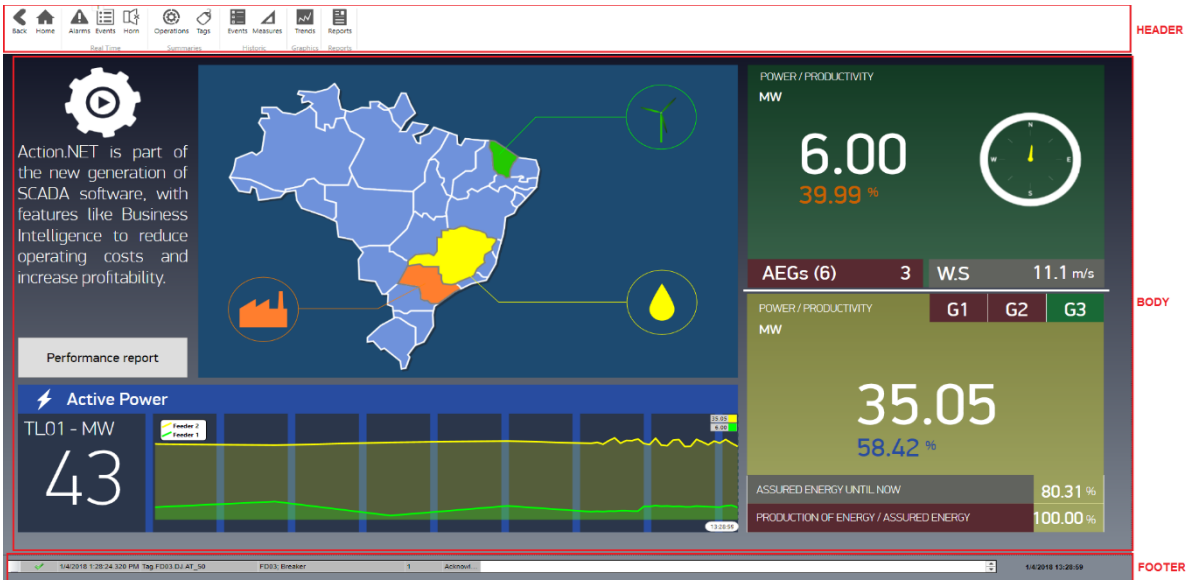
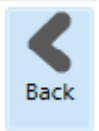


Figure 9 – MAIN_SCREEN

2.4.1 Header

Figure 10 – Header

The buttons located at the top of the screen enable navigating between the various reports and functions of the project, as described below. When you start a new project design, all header reports and functionalities are already ready on the default new project.



Back: takes you back to the screen that was previously opened.

Home: leads to the main screen of the project.

Alarm Summary: opens the screen **AN_AlarmsSummary** that is an active alarm report. Filters can be made by clicking the arrow that appears next to each column name, as shown in Figure 12. You can also sort the alarms by a chosen column, simply clicking on the name of the column.

If Alarms module is not running, the Alarm Summary icon change its color to red:

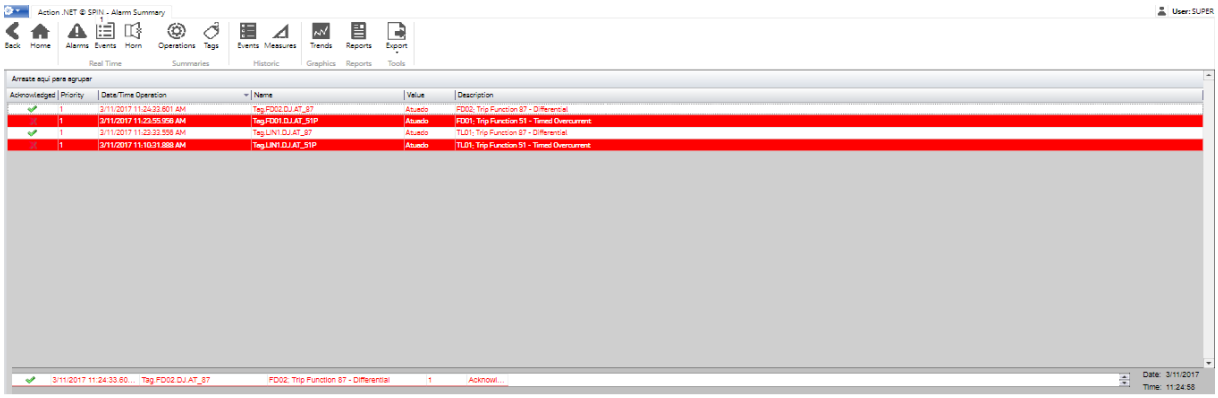


Figure 11 – Alarms Summary

On this application, all alarms must be acknowledge by double clicking the acknowledge column of each alarm line. When you do that, the software associate the acknowledge time stamp to the alarm and change the letter and background color of the alarm.

The columns presented on the form are that one set to visible on the project design, but if you click with the right button of the mouse over the line on the top column, you will see all column information available and the visible ones are with a √ symbol as presented on figure 13. All reports have this functionality.

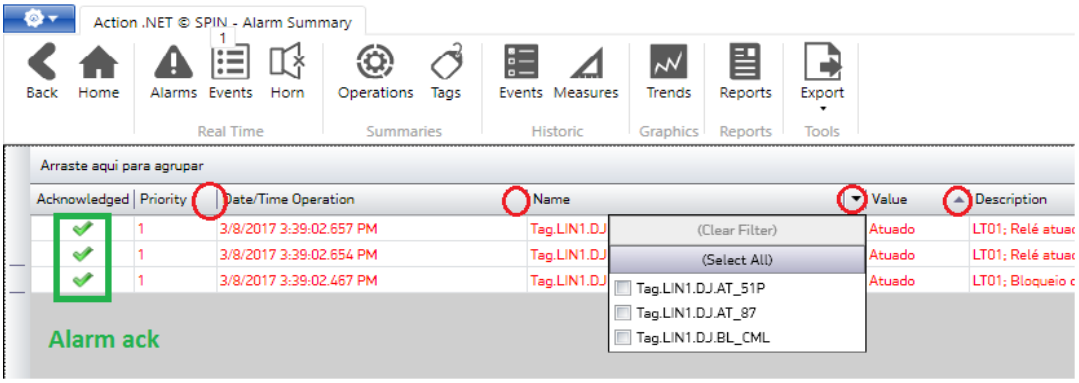


Figure 12 – Alarms Summary – filter option selected and acknowledge

Event Summary: opens the screen **AN_EventsSummary** that shows all the events of the day. As **Alarms Summary**, there are options of filters, sort and columns selection. If Alarms module is not running, the Event Summary icon change its color:

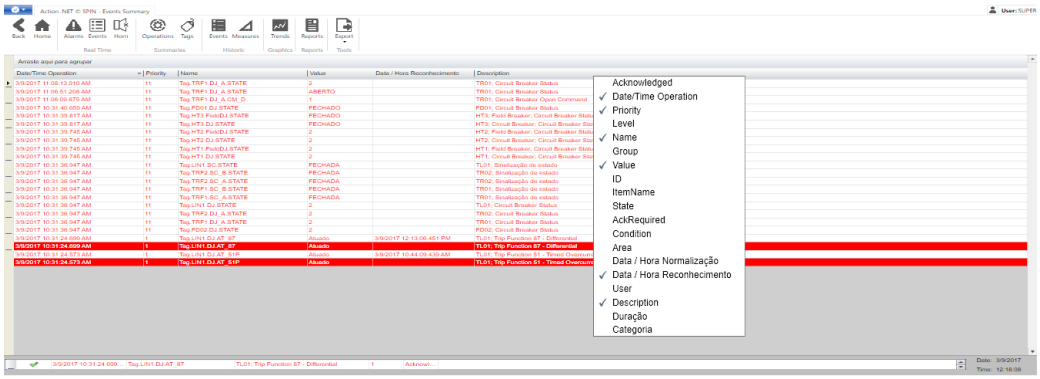


Figure 13 – Event Summary – Columns options

Mute Horn: Mute alarm horn.

Operation Summary: opens **AN_OperationSummary**, showing all operation actions like users logging, operator commands, etc. As **Alarms Summary**, there are options of filters, sort and columns selection.

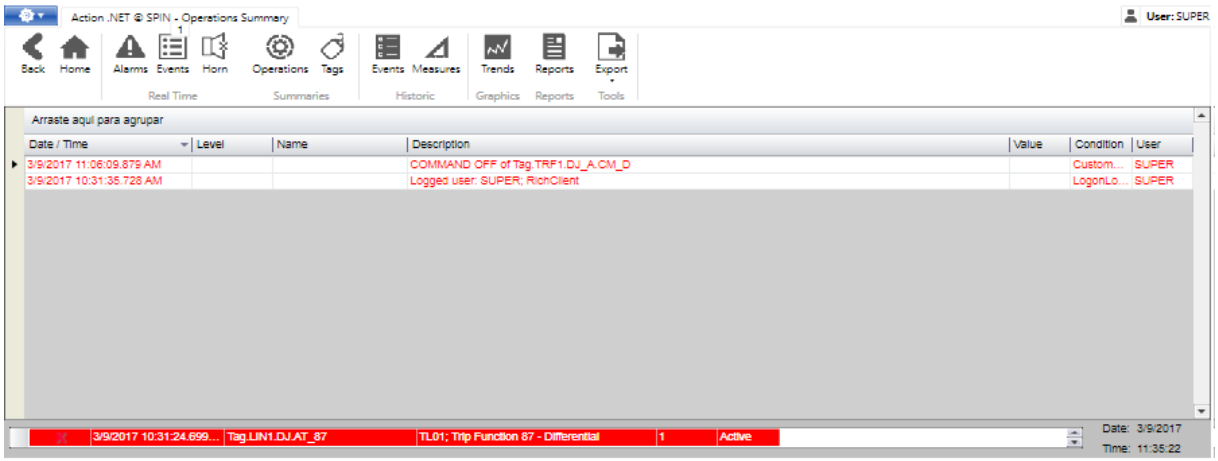


Figure 14 – Operation Summary

Tags Summary: opens the screen **AN_TagView** that is used to see the main attributes of a group of selected tags.

When you select tag summary report, as shown in figure 15, the screen is empty and is added two new columns in the report: Export and Filter.

Figure 15 – Tag Summary empty

You should select Filter, as shown in figure 16, and select the tags you want see and, in the sequence, press filter button to see the selected tags.

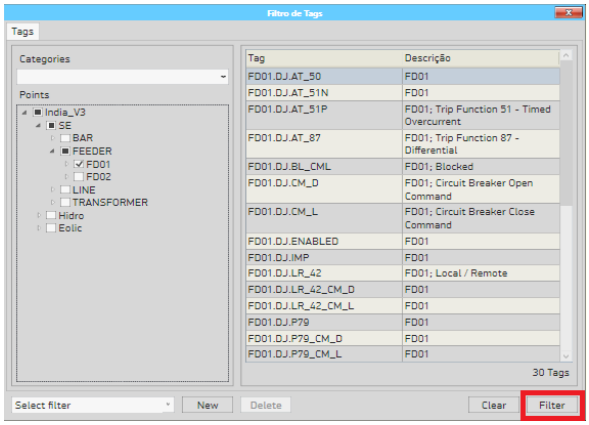


Figure 16 – Tag filter window

After press filter and the report will be filled out with all selected tags with all main information of the tags, as presented in figure 17. If you press export button, one .XML or .PDF file will be generated.

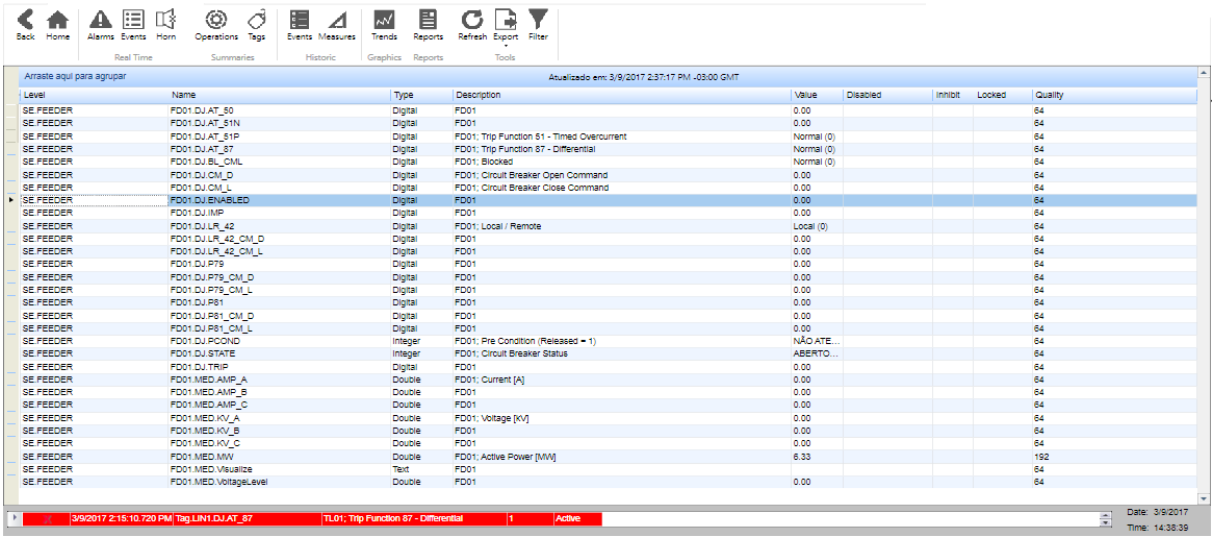


Figure 17 – Tag summary with all main attributes

Events **Event Search**: opens the screen **AN_HistoricEventsSummary** enabling the search for historical events. When it opens it came empty or with the result of last query. If you want to do a new query, select the filter, say the period and the data of the query, the tags that you want to analyze the history and press filter. As **Alarms Summary**, there are options of filters, sort and columns selection. . If Alarms module is not running, the Event Summary icon change its color:

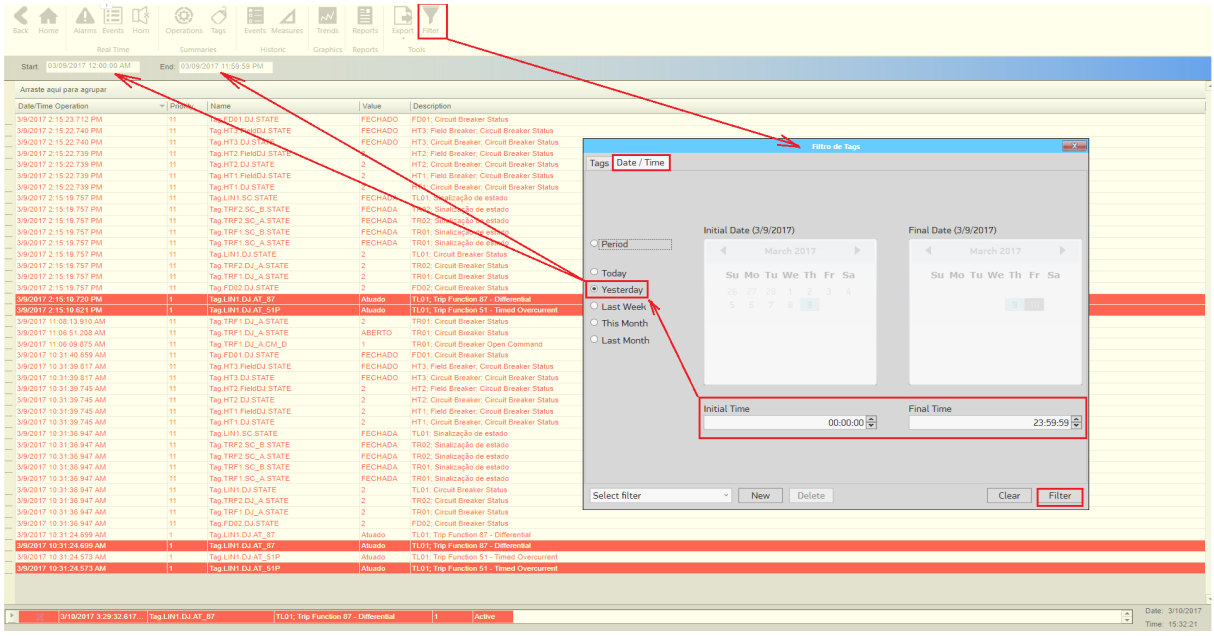


Figure 18 – History of events

Measures **Measures chart**: opens the screen **AN_HistoricMeasuresQuery** where, using again filter form, you select the variables and the period where do you want to see the measures. As **Alarms**

Summary, there are options of filters, sort and columns selection. If Historian module is not running, the measures icon change its color:

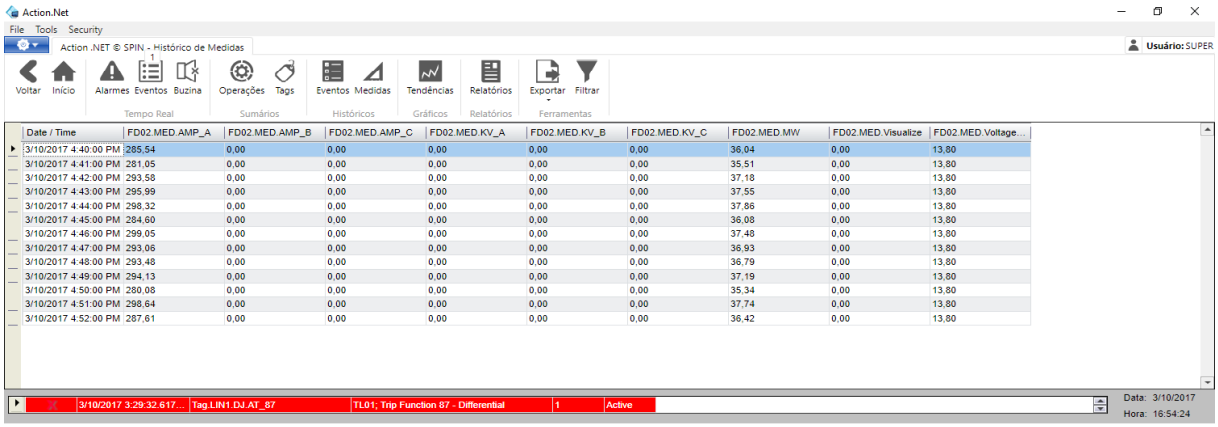


Figure 19 – Measures chart

Trend of variables (historical and real time): opens screen **AN_TrendsScreen** and, using the tag filter window, select the variables and the interval to trend. If Historian module is not running, the trend icon change its color:

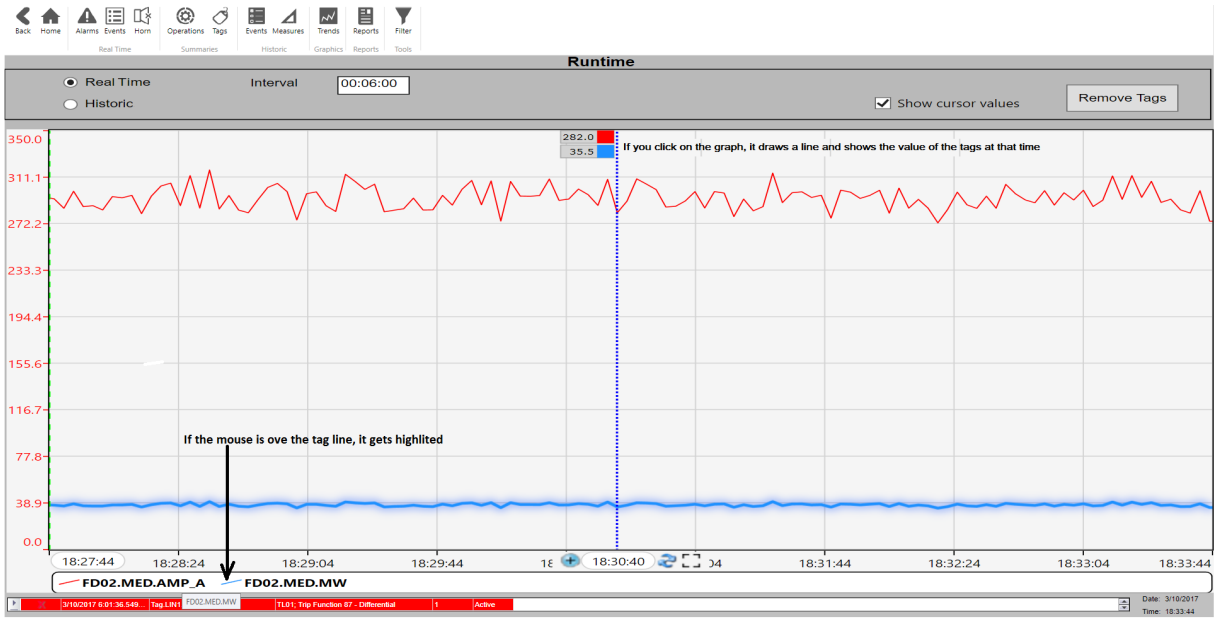


Figure 20 – Trend of tags

2.4.2 Footer

Figure 21 – Footer

The footer appears on most pages and shows the last active alarm of the system and date/hour.

2.4.3 Body

The body of a page is the area where information is displayed and where we have other navigation options.

2.4.4 Screens Layouts

A project can have different screen layouts. The screen below, as an example, shows the layout definition presented in NewDemo

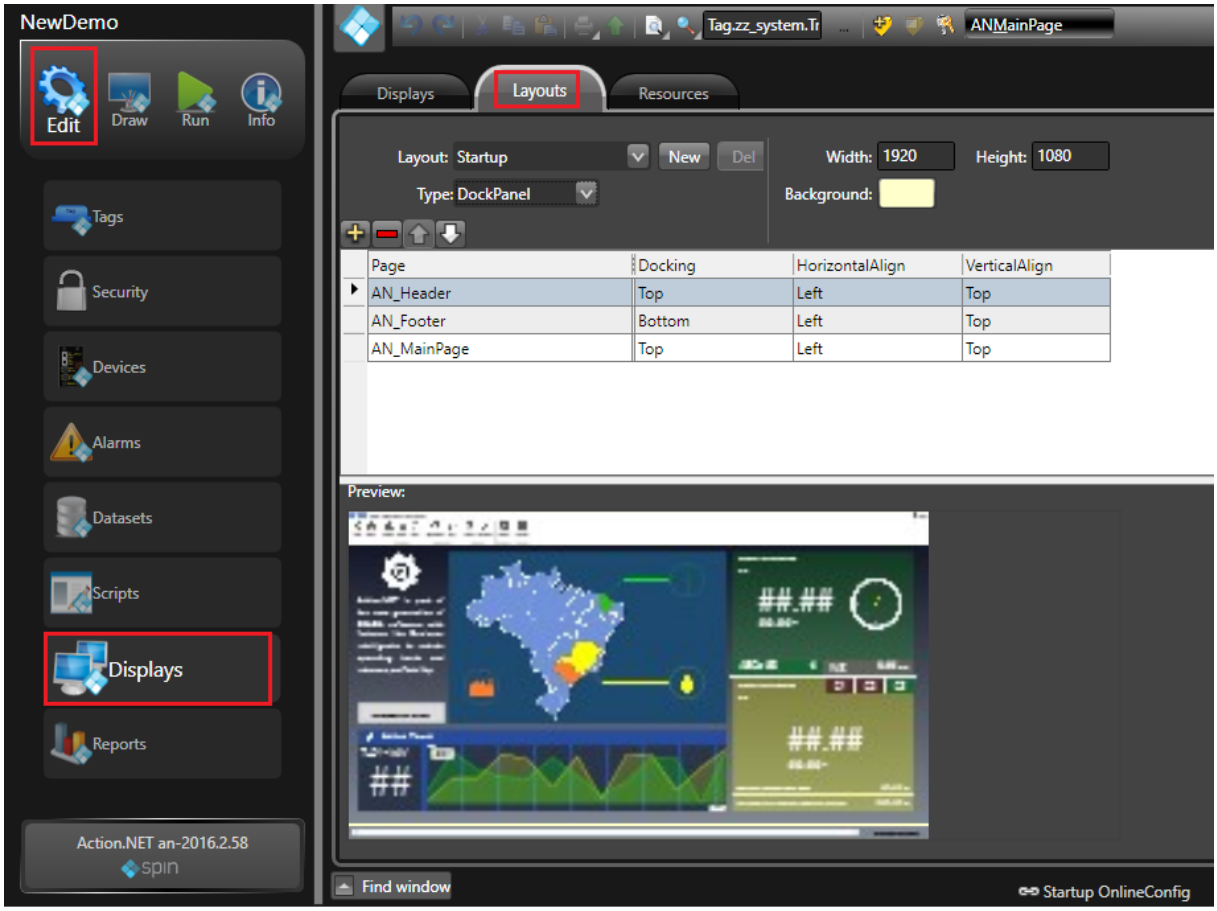


Figure 22 – Design of Layouts

2.5 DefaultNewProject

A great advantage of the program is that when creating a new application, the default project used already has all the reports mentioned, with their filters, as well as libraries of symbols suitable for use in electrical applications. This allows you to generate a project in much less time than normal.

If a customer already has a culture ready with their symbols, their reports, etc. Spin can tailor the Client DefaultNewProject to this culture and all new projects will be generated with the customer's cultural identity. A methodology called lean automation was developed to achieve this goal.

Lean Automation is a new methodology, developed by Spin, oriented to the parameterization and implementation of a SCADA application, for the supervisory control of a process. Spin, using Action₀NET, provides features that, when associated with the developed methodology, allows to generate better, cheaper, faster, error-free and high-quality solutions.

3. NAVIGATING THE PROJECT

This chapter will explain how to navigate through one project.

3.1 MAIN_SCREEN

In the body of the page **MAIN SCREEN,** we have a graph and four navigations buttons:

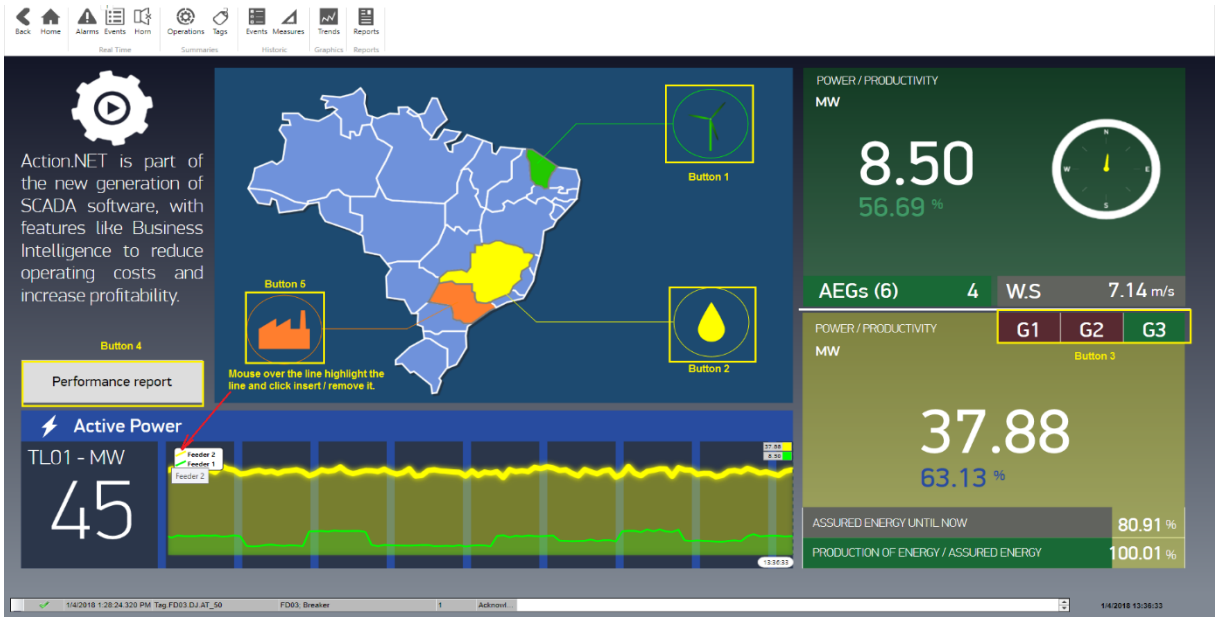


Figure 23 – MAIN_SCREEN: Body

3.1.1 Graph

The displayed graphic on this page shows real-time active power of the wind farm and power plant.

When we click mouse right button over the label (Active Power Hydro & Eolic) disposed in the upper left corner of the graph, the lines are highlighted to indicate the correspondence between the graph lines and subtitles.

Clicking at any point on the graph, it shows the control variables values at that point and the time associated.

To move the graph on the time axis just click and drag the mouse horizontally (below the graph). The mouse changes to " ", when you pass by the area change of the time axis.

Selecting a chart area by pressing the left mouse button on the chart selected portion displays the chosen field zoomed.

3.1.2 Button 1 – Wind Plants

Opens **SPIN_WIND** screen.

3.1.3 Button 2 – Hydropower

Opens SPIN_HIDROLOGY screen.

3.1.4 Button 3 – Hydro Generating Units

Opens HYDRO_GENERATORS screen.

3.1.5 Button 4 – Performance Report

Opens **AN_DIAGNOSTIC** screen.

3.1.6 Button 5 – Performance Report

Just as an example of screen possibilities is presented a sophisticated process control screen.

3.2 SPIN_WIND

In the body of this page are shown certain information relating to wind turbines, active power, capacity factor, wind direction and speed.

Figure 24 – SPIN_WIND

The color of each turbine is associated with its status and clicking on one turbine you are led to the **WIND GENERATORS**, which displays more details about the selected wind turbine. The substation icon is also a button and takes us to the "SL-SUBSTATION" that shows the single line of the substation.

On the bottom of the screen are shown, for each one of the six turbines, the active/reactive power, the wind speed and the status of turbine.

On the right side of the screen is shown the number of turbines in operation, the total active power, the capacity factor and the medium wind speed.

3.3 WIND_GENERATORS

Figure 25 – Detailing Wind Generators

On this page information about each wind turbine is displayed. The turbine being shown has its name up in the upper right side of the page and its status in the colored rectangle.

In the body of this page, we have no navigation options. Thus, in this page navigation is restricted to the header.

3.4 SL_SUBSTATION

This screen shows the substation synoptic where displayed measured values associated with the bays and states of switches and circuit breakers. Clicking over these elements, a command window opens and you have the option to open or close the selected equipment, as well as, change the status of it:

- Disable: Stop to read the point;
- Inhibited: Continue reading the point from field, but don´t cause event;
- Simulation: The state of the equipment is simulated by operator, disregarding the field value;
- Lock with flags: you lock the equipment command signalizing the motivation;

The buttons on the left side of the screen open a window with the relay front face associated to the feeders.

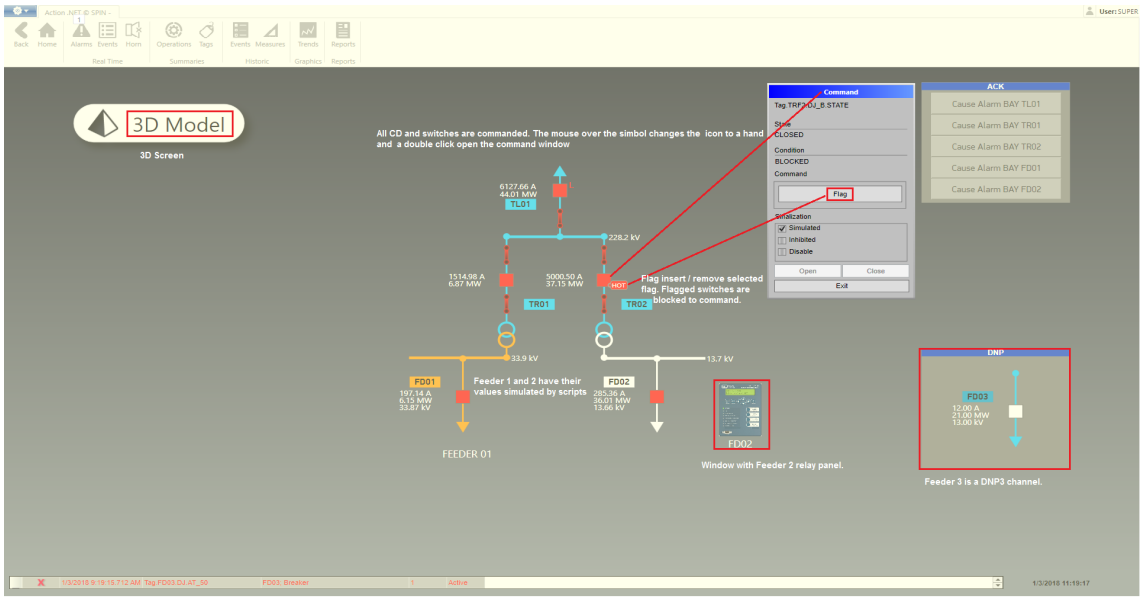
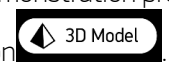


Figure 26 – Single-Line Substation

3.5 3D Model

Action NET has a feature that allows the development and implementation of 3D projects. In this demonstration project, a 3D screen can be accessed from the screen SL_SUBSTATION by clicking on the icon 

. The 3D configured screen corresponds to two lines of single-line diagram of the SE (substation) ACN. The SE of the 3D model screen is shown in Figure 27.

In this screen, there are three positioning controls, which are at the bottom left of the screen. The control leftmost moves the image up and down, right, and left. The second controller rotates the image around its central axis vertically and horizontally. The third control zooms in and zooms out. The position is shown in the lower right corner.

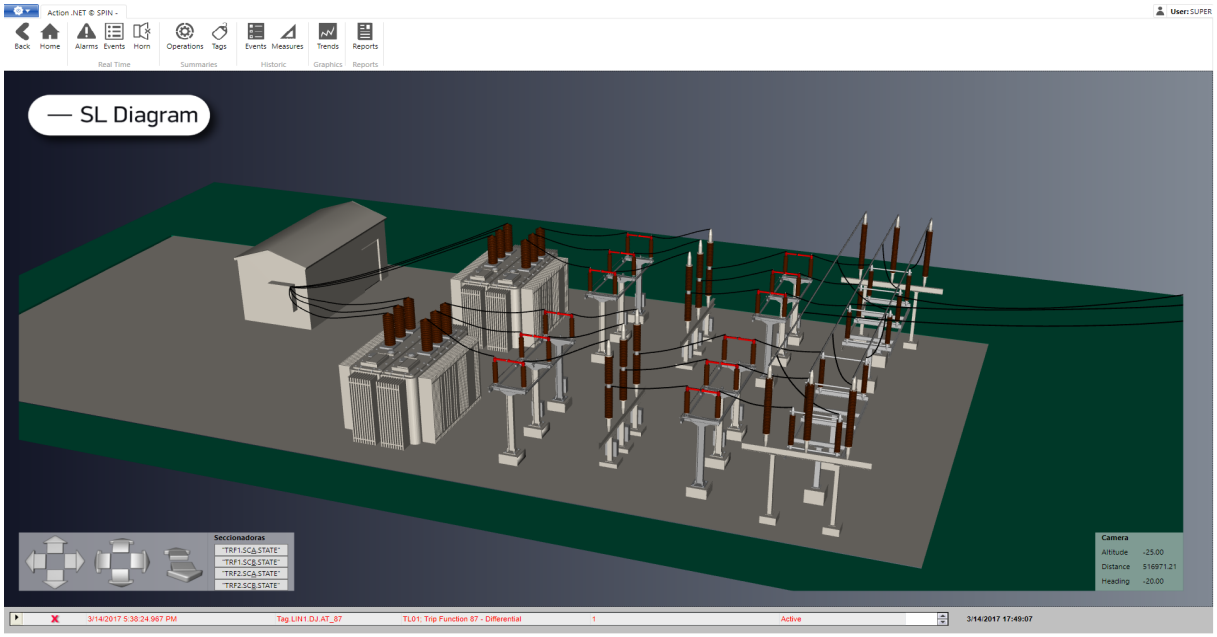


Figure 27 – Single-Line Substation 3D Model

It is possible to view and execute commands for the 3D screen. There are commands available for four rotary switches on the buttons next to the zoom control. To run a command on the disconnect the first line, for example, you must click on the **Secc1A** button. A command box will appear showing the **Open** and **Close** buttons, and just the opposite command button to the current state will be able. In the example, only the command to open appears enabled because the disconnect switch is closed. Then a command confirmation box will be opened (Figure 28). After confirming the command, it is executed and you can monitor the implementation through the animation.

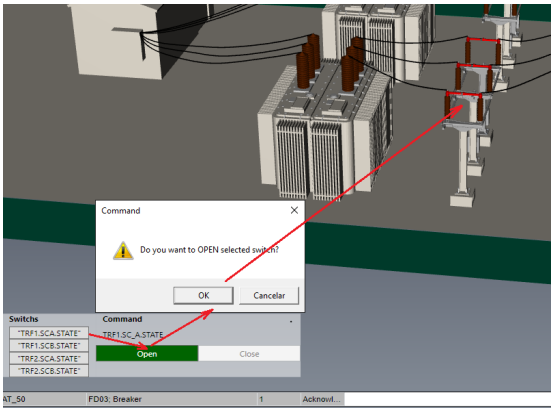


Figure 28- Control box on 3D display

To return to the single-line diagram simply click on the icon:

3.6 SPIN_HIDROLOGY

This screen shows hydrology information of a hydroelectric plant.

On the top of the screen there is information about the floodgates of water intake and bottom have information on the spillway gates. On the bottom right, there is information on the water discharge, the reservoir level and downstream level.

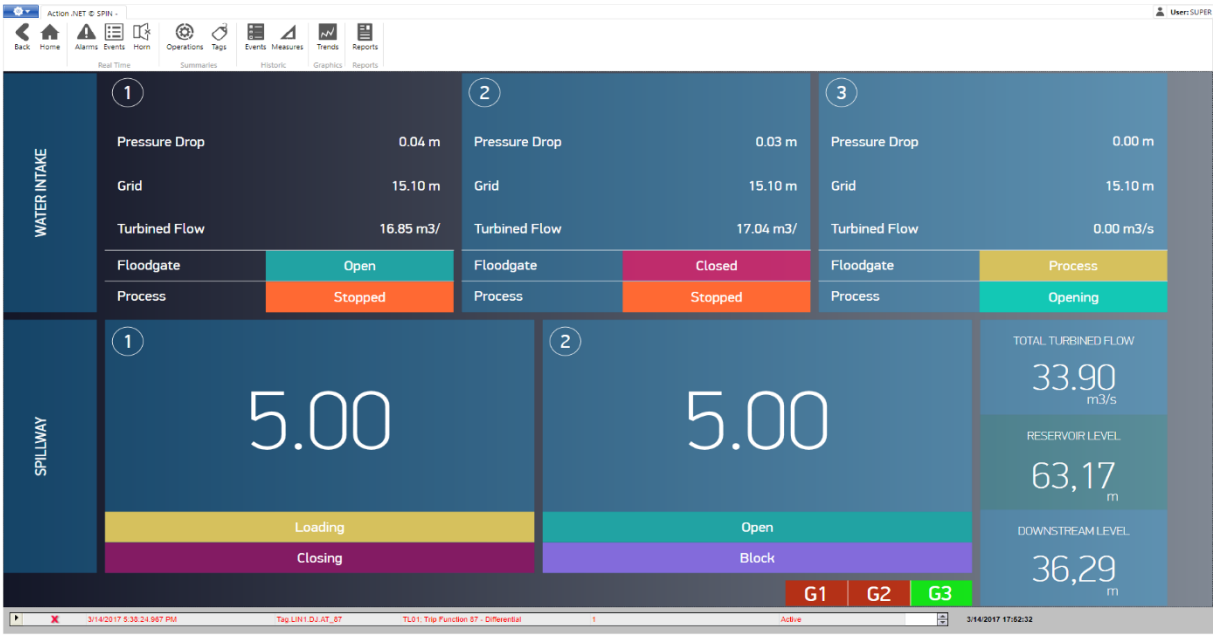


Figure 29 – Hydrology

Next to the bottom, there are three generators buttons (G1, G2 and G3). By clicking on it, you access the generator screen: HYDRO_GENERATORS.

3.7 HYDRO_GENERATORS

As in the single-line diagram screen, in this screen circuit breakers and disconnectors can be controlled by a click on the element. This screen displays the single line of the generating units, the units containing circuit breakers, disconnectors of ancillary services, breakers of the generator field circuit and the status of the generator as shown in Figure 30.

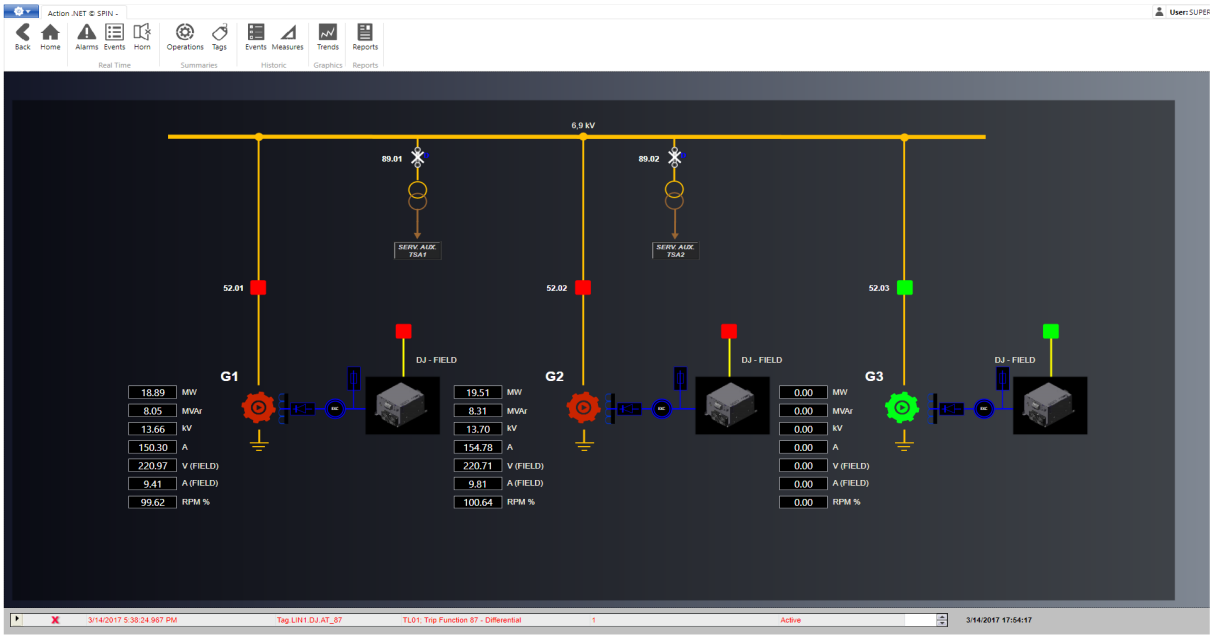


Figure 30 – Generating Units

3.8 Performance Report

Figure 31 – Performance Report Screen

Displays information about CPU usage, physical memory usage, disk and network usage, alarms and historian synchronization fail.

If the device application is no run, the memory values are not updated once it is used a monitoring channel to run this screen. If you are in a client workstation, the performance report always monitors the SCADA Server and to do this is used a protocol “SystemMonitor”.

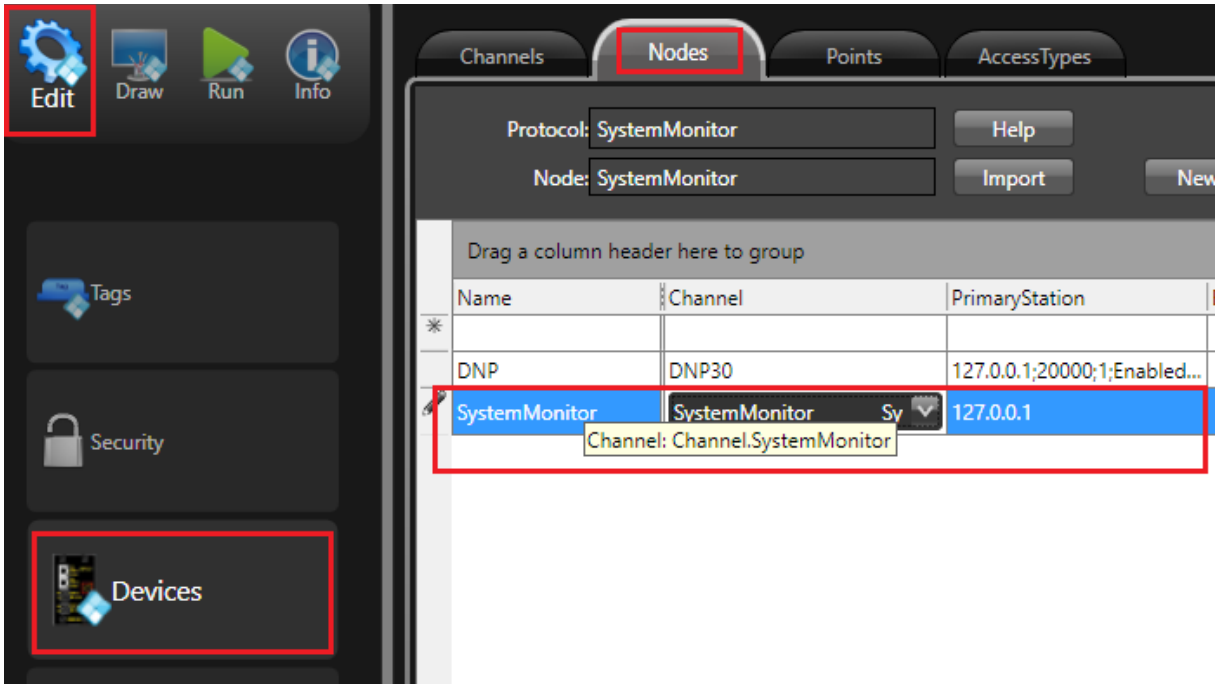


Figure 32 – SystemMonitor Channel

3.9 Control Panel

Figure 33 – Control Panel

The nº 1 and 2 buttons change the screen's variable. Button nº 3 changes de relay configuration Local/Remote. Button nº 4 changes Lock and Unlock 79. All the other buttons may be configured according of what is configured in the field.

The Led (5 to 10) has the following meaning:

a) Led 5 inform the 79 status (locked / Unlocked – tag.FD02.DJ.P79_Lockout);

b) Led 6 and 7 Enable and disable 79;

c) Led 8 CB Closed;

d) Led 9 CB Open;

e) 10 – has five LEDs:

- Tag.FD02.DJ.ENABLED: enabled / disabled;
- Tag.FD02.DJ.TRIP: If 1, tripped;
- Tag.FD02.DJ.AT_50 – Instantaneous;
- Tag.FD02.DJ.AT_51P – Phase overcurrent;
- Tag.FD02.Dj.AT_51N – GND / Neutral Overcurrent;

3.10 Area Alarm

There are five area alarms identifying if there are any alarm in 5 bays:

(1) Alarm in the Line bay (TL01);

(2) Alarm in the transformer TR01;

(3) Alarm in the transformer TR02;

(4) Alarm in Feeder 1;

(5) Alarm in Feeder 2.

The figure below has five buttons that generate / normalize the five area alarms.

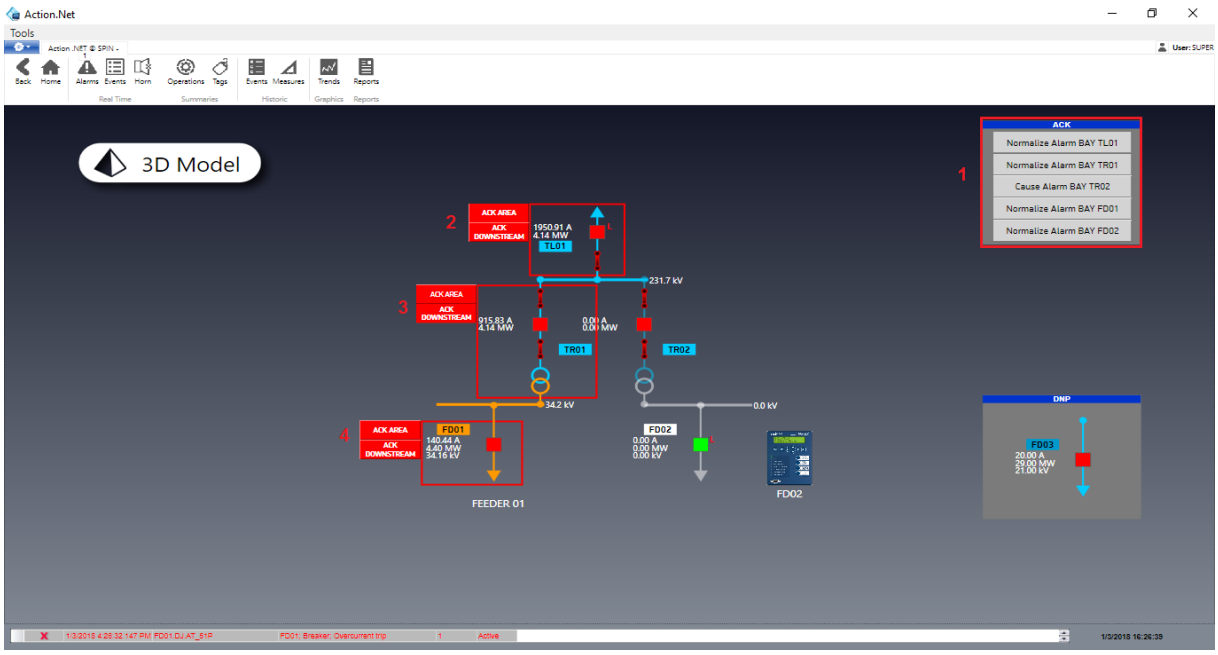


Figure 34 – Area Alarms

When you cause one area alarm, the respective bay is framed with a red line and two buttons appear: one that recognizes the area alarm and another that recognizes the area alarm and all area alarms below it. In the figure above three area alarms were caused, one on line bay TL01, one on transformer bay TR01 and one on bay feeder FD01. If we click on the "Ack Area" button of a given bay, we will only recognize this area alarm and, if we click the "ack downstream" button of a given bay, we will recognize this area alarm and those below it. Observe that one area alarm is caused if any alarm associated to that bay is caused.

3.11 Run DNP 3.0 Server Program

The Feeder 3 is supposed to be connected to a DNP3.0 device. In this manual, we get a 45 days free DNP 3.0 server simulator called Axon Test. See Annex 1 for more details.

4. INFORMATION ON THE APPLICATION PARAMETRIZATION

4.1 Project Info

As shown in figure 1 and figure 4, this is an Express version limited in 75 I/O points.

In the parameterization, 698 tags have been created. That means a release of 75 I/O points may have much more tags (alarms, variables calculated, display objects, etc.).

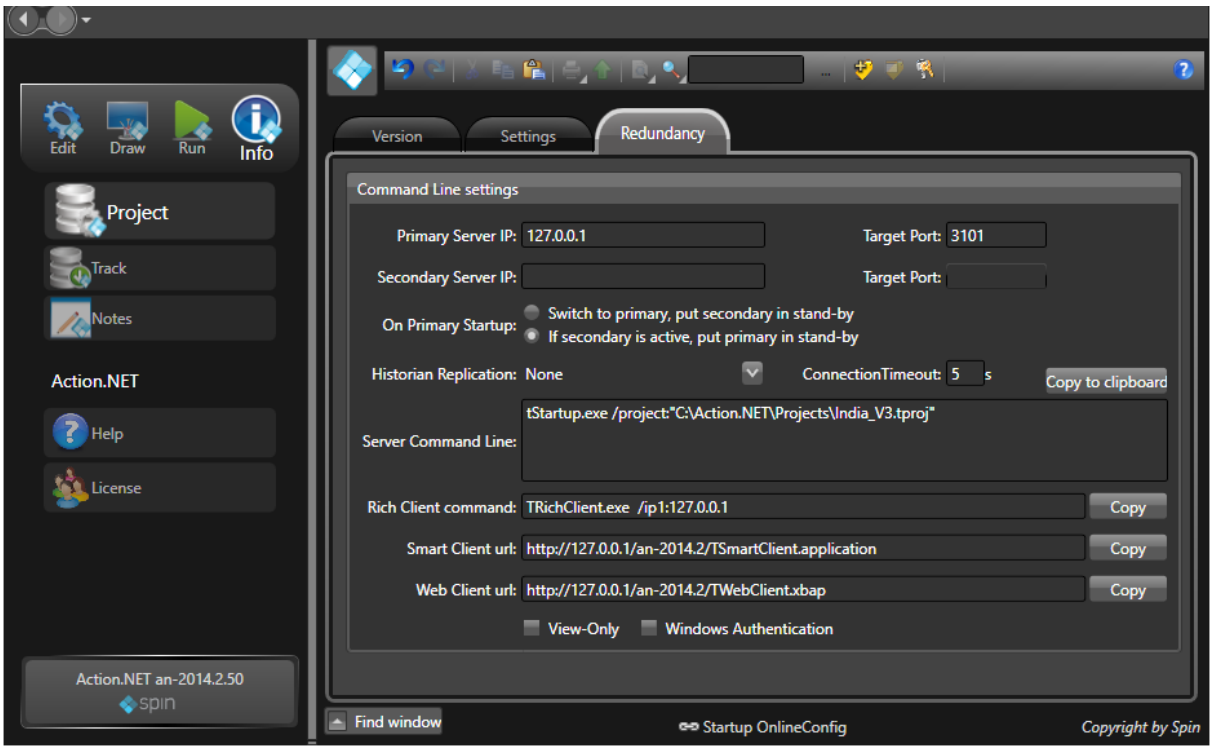


Figure 35 – Redundancy

This application has only one server, and its IP address is 127.0.0.1 (localhost). With this, the system automatically generates the partial path to the shortcuts:

- TStartup: enables the Runtime, allowing the system to automatically activate the application even if the machine boots;
- Other shortcuts can be copied to any machine on the network, if a valid IP, which should trigger HMIs type Rich, Smart and WEB.

The above screen shows that to declare a hot-standby application, you only have to declare a primary server and a secondary server. In the screen above you can see other options that can be defined, for example, three types of different clients connected to the servers.

4.2 Edit

4.2.1 Templates

Templates are used to define mirrors records of an element, such as Line, Transformer, Feeder, Generator, etc.

The Express version can only create two template levels, but in the normal version, you can create as many template levels as you want, that is, you can declare a template line, which contains a breaker template type, breaker protections type, etc.

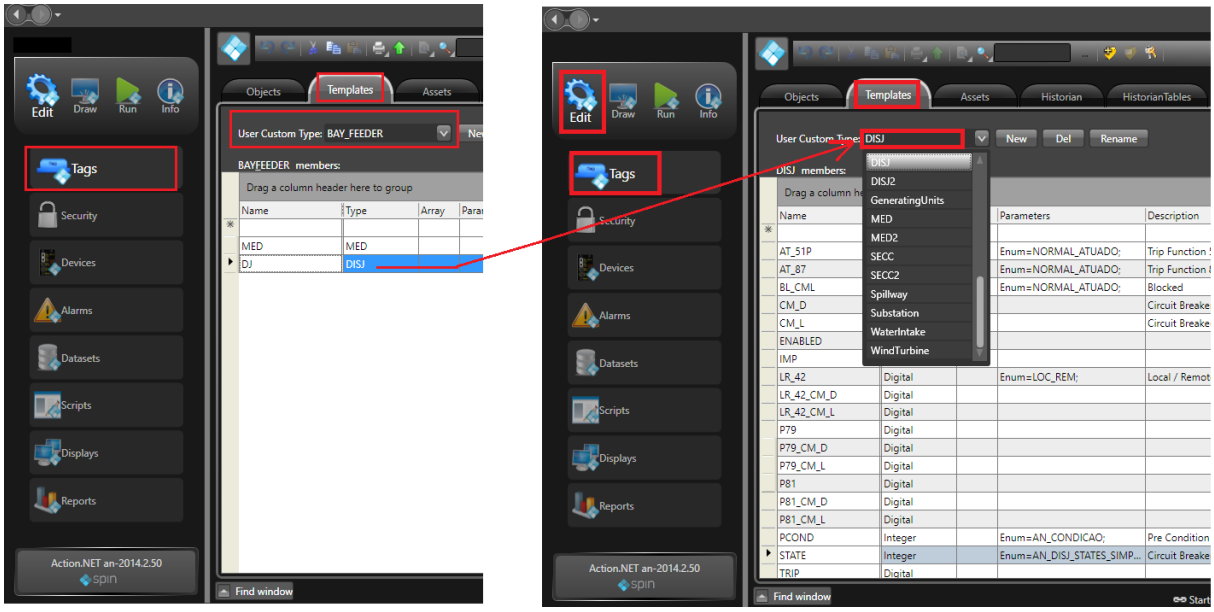


Figure 36 – Templates and templates inside templates

The templates that start with "AN" are internal to automatically generate alarms reports, events, query historical data, real-time trend and historical, etc.

Parameter type E-num is an automatic conversion table representing an integer value for an associated text. For example, in the case of a disconnector:

0 = undefined;

1 = opened;

2 = closed;

3 = failure.

4.2.2 Assets

Each system tags may be associated with a level that corresponds to an asset tree. This tree can be used in report filters, for example, a filter that shows only wind farm variables.

Thus, for example, in the figure below are all assets (levels) classified hierarchically, and in **Figure 38,** it is shown that each tag is associated with a level of **Figure 37**.

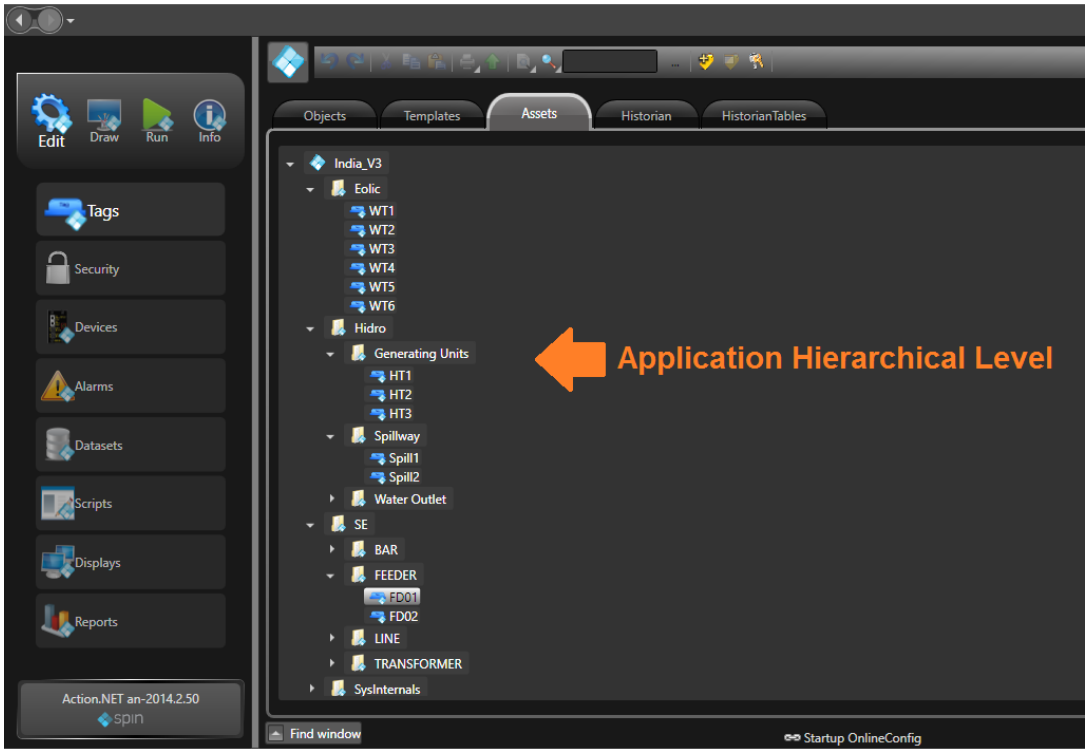


Figure 37 – Templates

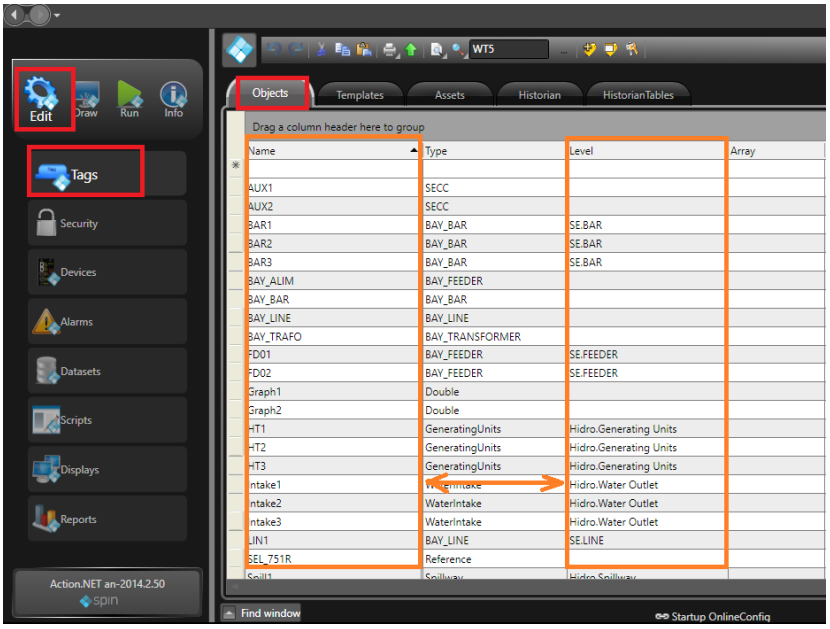


Figure 38 – Tags are associated with levels

4.2.3 Devices

In this edit area are defined the communication channels, the devices associated with a channel, and the points associated with all channels, organized by protocols.

For example, shown below is the only channel of the points of this application.

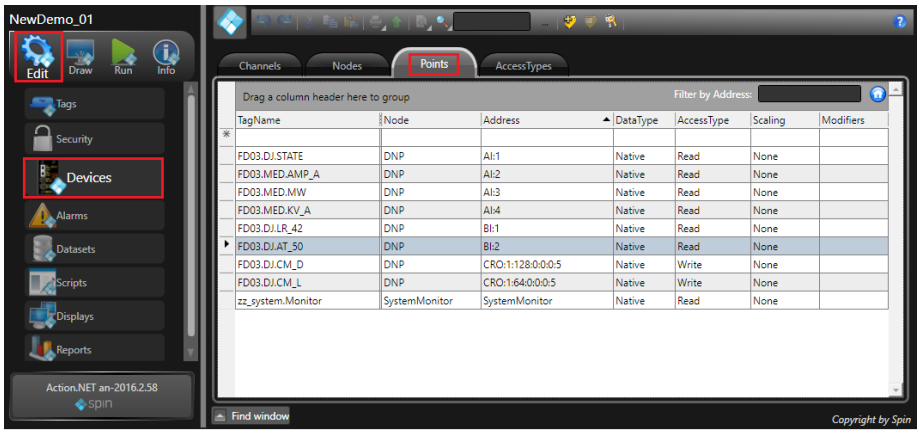


Figure 39 – I/O tags

4.2.4 Alarms

In this area are defined the alarm groups created and alarm items associated with each group. There is a third column called Area, which lets you create a group of alarm associated with multiple alarms. This allows, for example, creating an alarm group, which will be actuated if there is any alarm in a given area such as a generator, a substation, a wind, a circuit, a bay, etc.

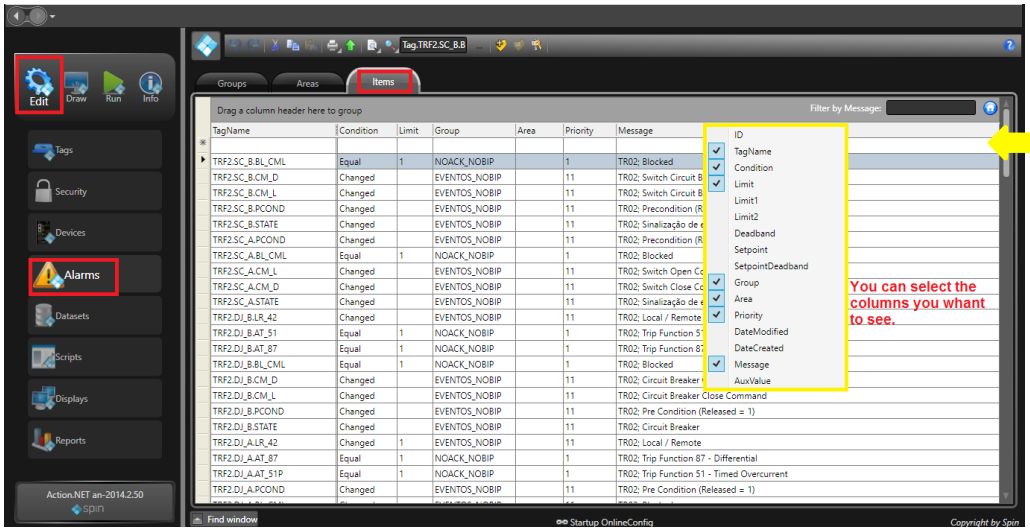


Figure 40 – I/O tags

In engineering module, the columns shown are the most used (the symbol "√" indicates the selection in the yellow highlighted list). If the user clicks the right mouse button on the line under the yellow arrow ( ) shows all existing columns options and clicking on it, it is displayed on the screen.

4.2.5 History Tables

At **Historian** and **Historian Tables** columns, you can define history tables that will be created, the recording frequency of the same, and the variables associated with these tables.

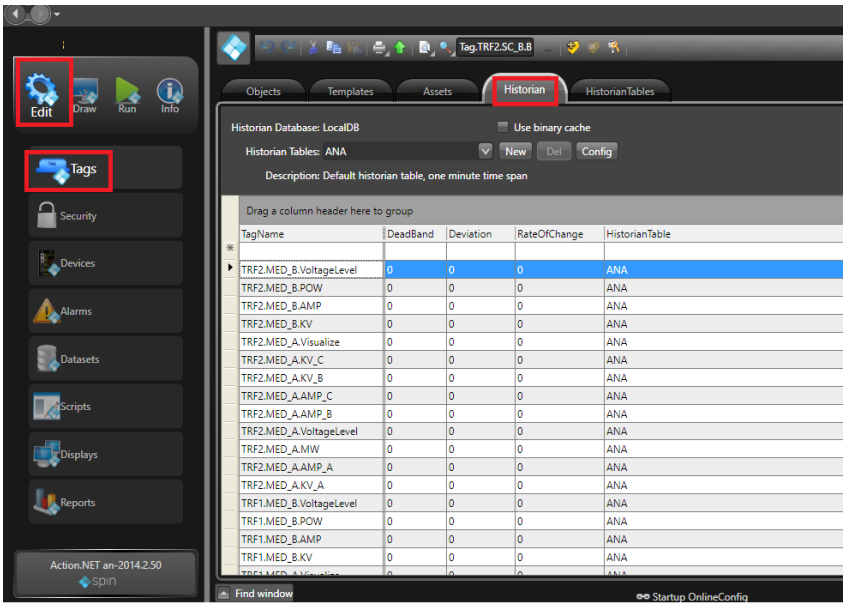


Figure 41 – History table

4.2.6 Datasets

In **Datasets,** you can associate history tables and alarms to a database. Action₀NET has a SQL Express default relational database built-in (SQLite). For larger applications, it is suggested a market relational database like SQL Server, Oracle, etc.

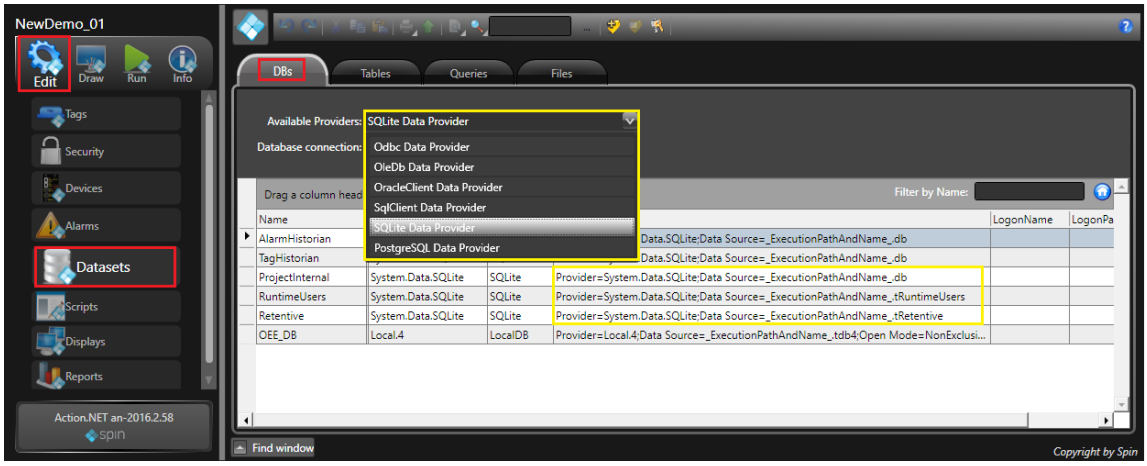


Figure 42 – Datasets

4.3 Drawing Screens

4.3.1 Construction of Screens

The Draw option lets you draw on Action₀NET screen. It is quite simple to create and/or modify the SCADA screens, but does not fit this NewDemo to teach how to do it. We suggest you to look on the manual.

For a quick look, we share a screen and some information on it.

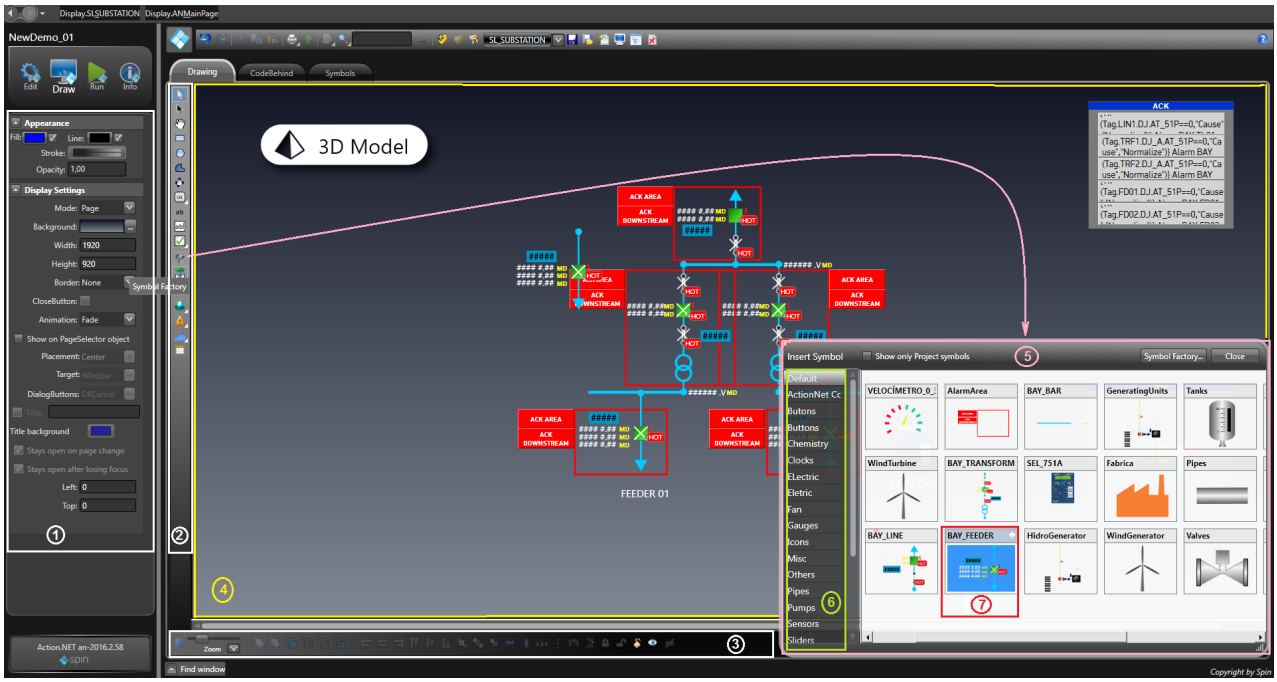


Figure 43 – Screen with a substation single-line

(1) This area contains attributes of the current screen (SL_Substation).

(2) Vertical toolbar allows you to create or select display objects. Among the tools: lines, circles, polygons, buttons, radio button, trends, alarms, etc. You can use an already built symbol or create a new one.

(3) The horizontal toolbar allows you to zoom, group/ungroup symbols, align objects vertically or horizontally, etc.

(4) This is the area dedicated to the design itself;

(5) This window is showing a symbol library used in this application. These symbols are vectors and already have all the features and associated animations. Once created the user symbol library with the culture of your company, the process of creating screens becomes a very simple activity. The system has dozens of built symbols cataloged in libraries referenced by type. On electric applications one symbol can be a bay (feeder, line, transformer, reactor, Generator, etc.) If one symbol has the name of a template, when you associated one bay to this template (for example FD01 ← Bay_Feeder) all bay animation, alarms, historian and even address points can be, automatically, created (see Lean Automation).

(6) The symbol library is what makes screenmaking a simple and yet sophisticated task. The software is already distributed with dozens of libraries associated with motors, tanks, vectors, pipes, etc. Using these libraries users create new libraries that are, for example, all symbols associated with the culture of a power utility. In this demo, for example, we have symbols that are the bays of line, feeder, transformer, etc. Each bay is composed of dozens of symbols, grouped, and with all the animation logic associated with the company culture.

(7) As stated above, each symbol library entry is the grouping of dozens of symbols with the entire culture of the power utility. For example, one feeder in this demo is presented below. When you associate the FD01 to a Bay_Feeder, all variables of the template are associated to that feeder (FD01.<tags of template>) and all tags are, automatically, associated to each symbol that comprises the symbol bay feeder.

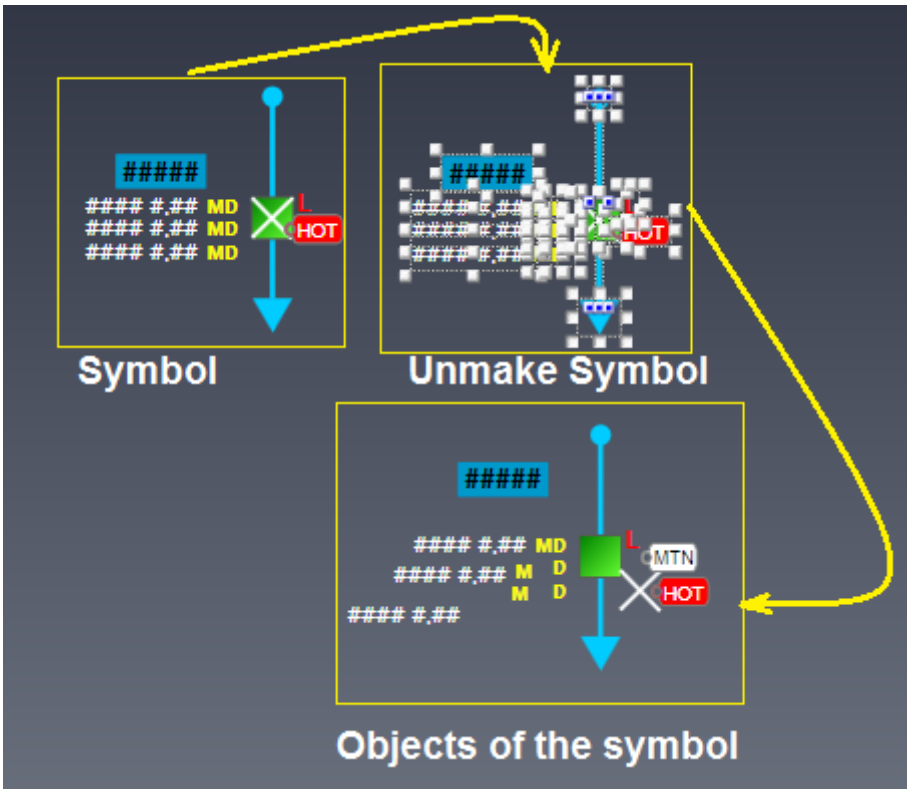


Figure 44 – Decomposing the symbol bay feeder

In addition to the symbols with the company culture, Action$_0$NET has the Symbol Factory table with over 3600 vector symbols to create new view objects.

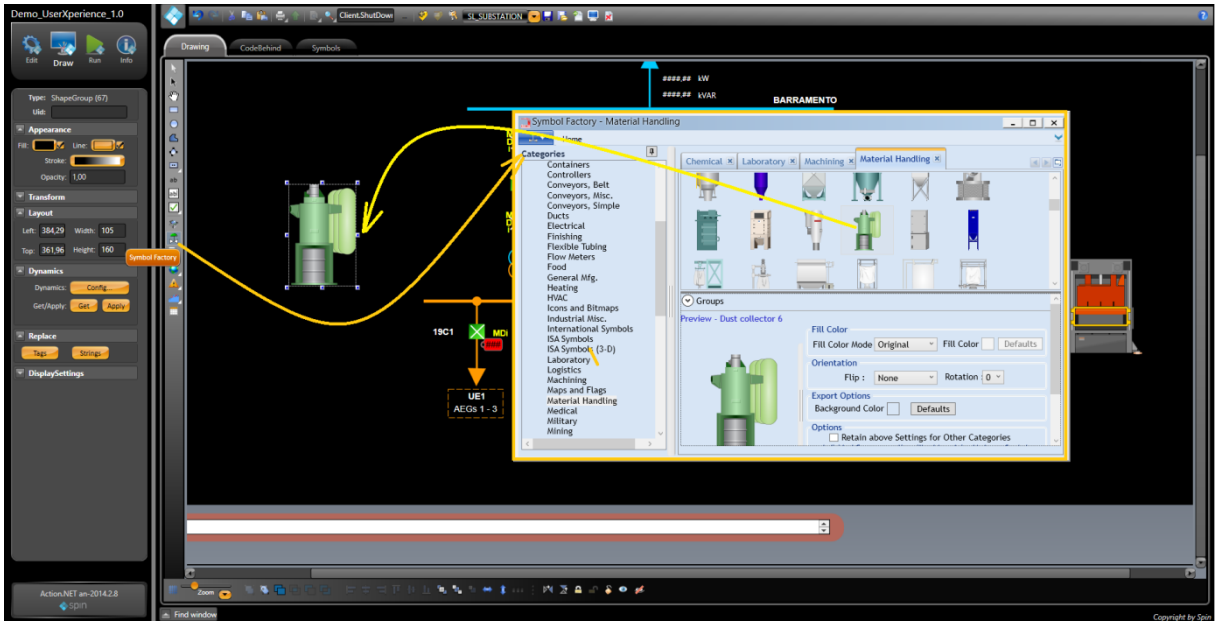


Figure 45 – Symbol Factory

4.3.2 Scripts Associated with Screens

The **Code Behind** window lets you associate compiled Scripts to events associated with this window, such as the opening screen, screen lock, etc.

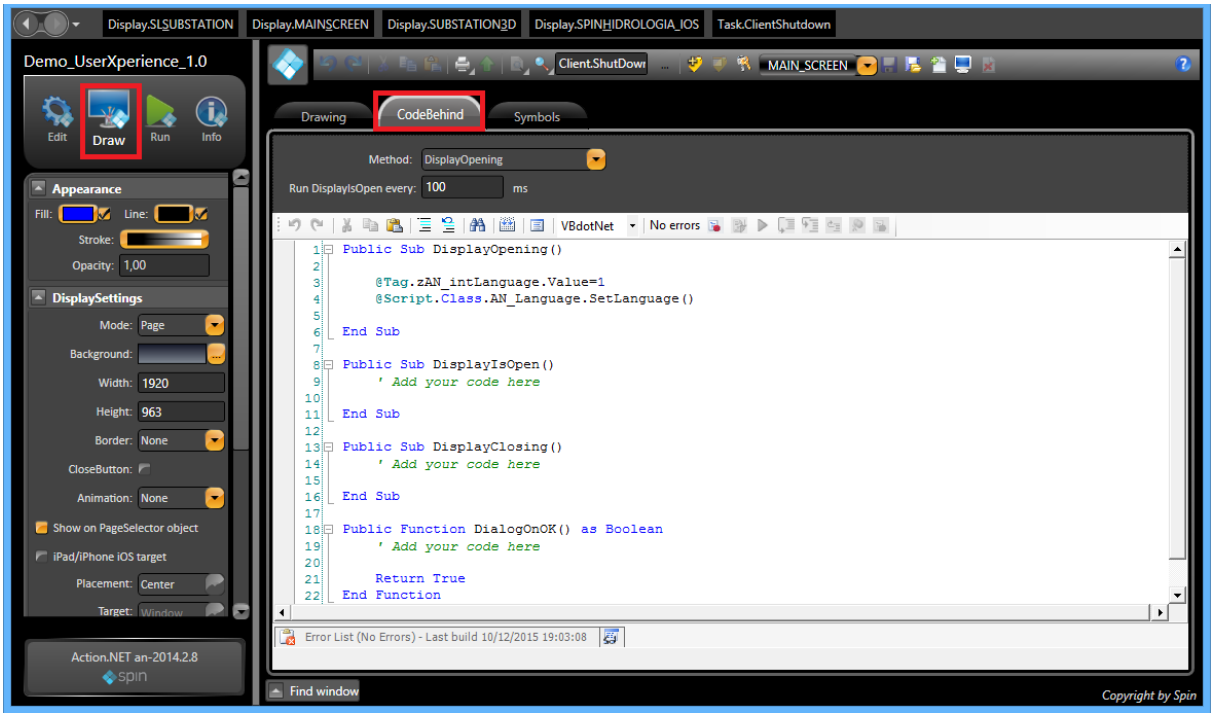


Figure 46 – Associated code to a screen

4.4 Compiling and Running Projects

4.4.1 Build

This option lets you compile the entire project, creating lists like:

- Number of times each object has been used;
- Cross Reference Table of Objects;
- List of objects created and not used;
- Etc.

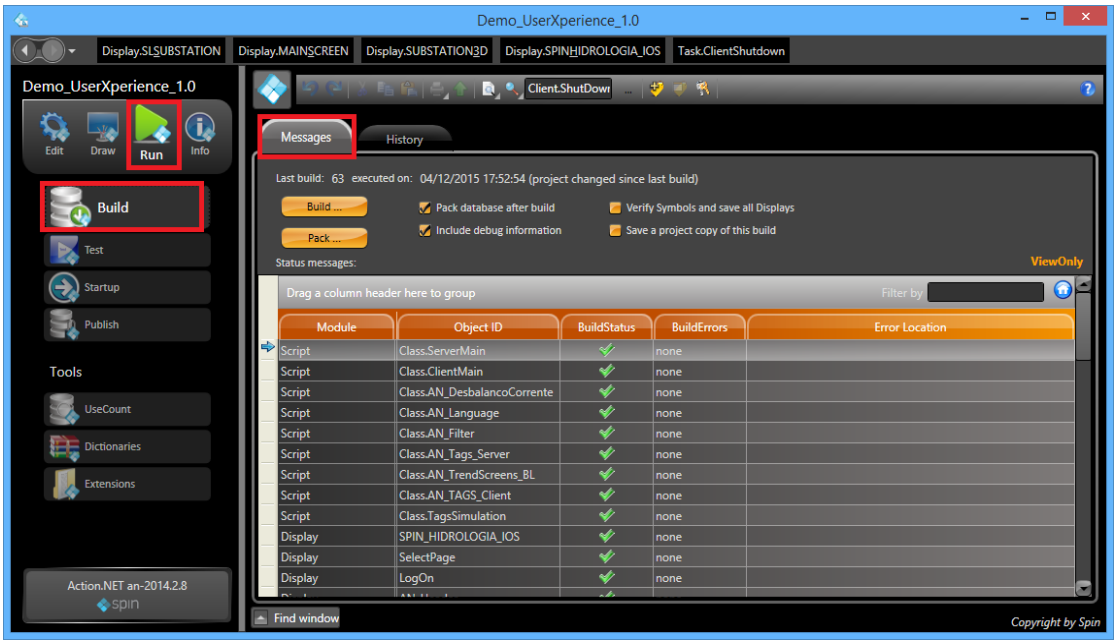


Figure 47 – Project Build

4.4.2 Start Up & Test

These two options allow you to run a project from the engineering module. In the case of Test, history and customer tables of alarms will not be disturbed.

In Start UP module, shown in the figure below, only the selected modules will be activated in **Run Module** (1). Thus, for example, if you did not create tags or point of I/O, but has done the design of screens and excitement associated with this, you can check your work by activating only the module **Display** and if there are any script associated with the screen, the Scripts module.

The **Run Local Diagnostic Tools** option (2) provides various debugging options of the project modules.

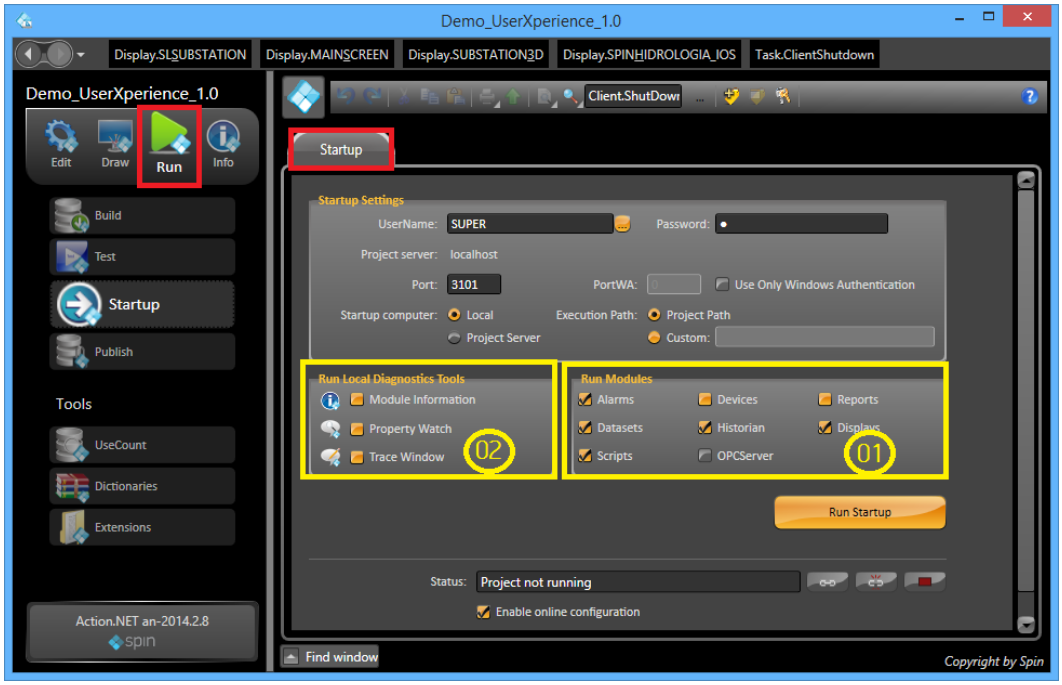


Figure 48 –Start UP project options

Among these options, this manual NewDemo shows the **Watch** window which, when selected, allows the user to change the value or property of any project object and check the behavior caused the application.

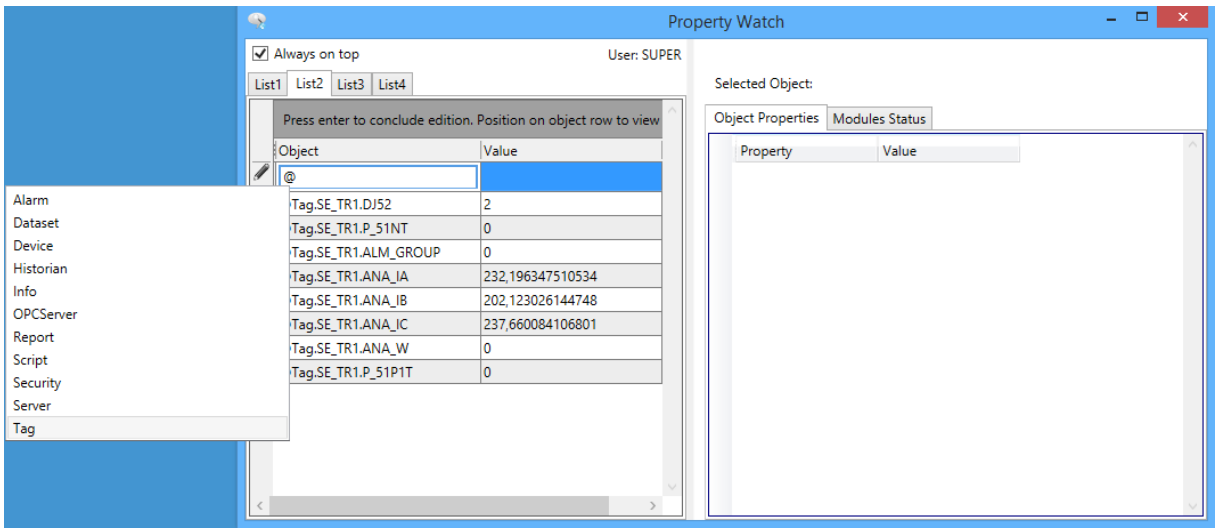


Figure 49 – Watch window (objects in the project)

You can create up to four lists of objects (List1 to List4), and in each list select objects you want to analyze. In **Figure 49** are listed all existing objects in a project, in which were selected **Transformer one** tags.

The window works with **intellisense** (as you type software selects the objects that cater to the text). So a "@" brings up all objects, selecting tag (light gray) and typing "." brings up all project level one tags and so on.

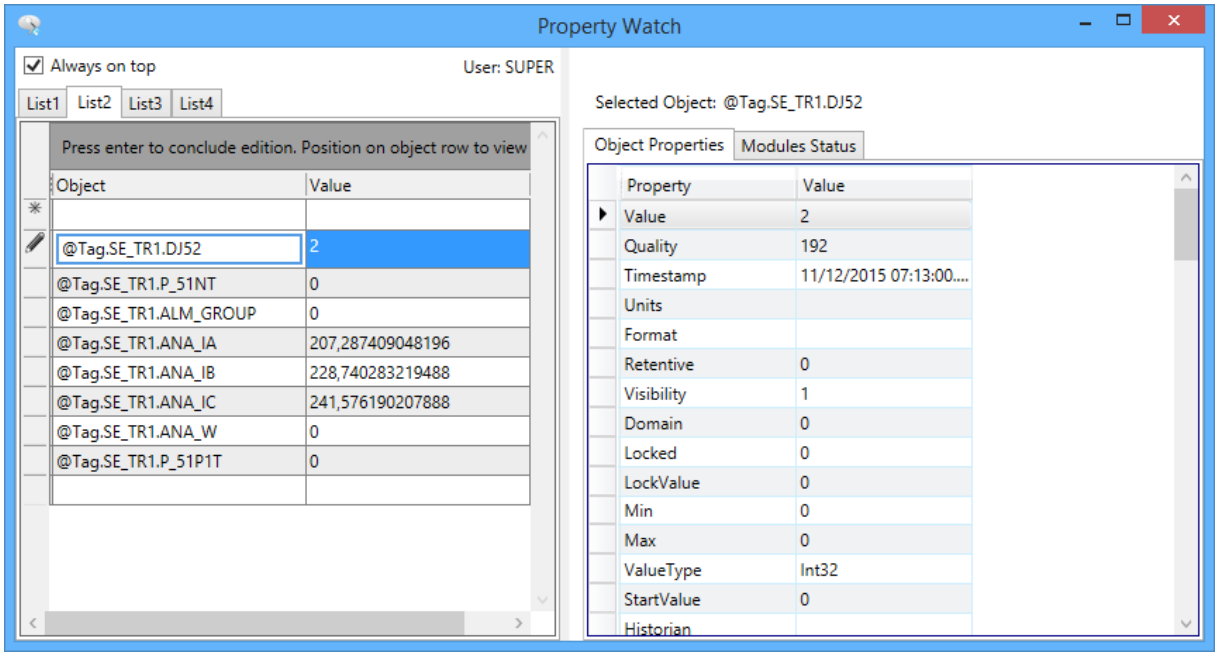


Figure 50 –Watch window (tag.SE_TR1.DJ52 selected)

Beside each object in the list is shown its current value (value property of the object). When you select a listed item, as in the case of the **Figure 50**, the **tag.SE_TR1.DJ52**, at the right side of the screen is shown all the attributes of this item (value, quality, etc.). If you change the value or attribute of the selected

object in the watch window, you will be changing it in real-time and scheduled animations will be caused. For example, in the above window:

- By changing the DJ52 value to 1, the circuit breaker will open, brightening up the screens where it is shown and causing up the events and associated alarms;
- By changing the quality of tags for any value other than <192>, the point will get error quality and, in the case of the circuit breaker, there will be shown an "X" on your image.

4.4.3 Dictionaries

There are three tabs in Dictionaries:

- Localization: it lets you create application dictionaries in several languages and, for example, creating an event that changes the language, all texts are translated into the new language. In this project NewDemo, the project is available in English and Portuguese.
- Enumeration: it creates tables with texts associated with numbers. These tables, on run time, allow a list of events, for example, to translate the status of a wind turbine for a callout.
- Categories: the system allows the creation of 64 categories, in which each category is a qualifier that can be associated with a tag. Thus, any tag can be associated with up to 64 categories and in design-time or real-time these categories can group tags into 64 groups. A typical categories application is shown in the film Deploy available on Spin website for automatically generating tables of alarms, events, points of I/O and history from templates.

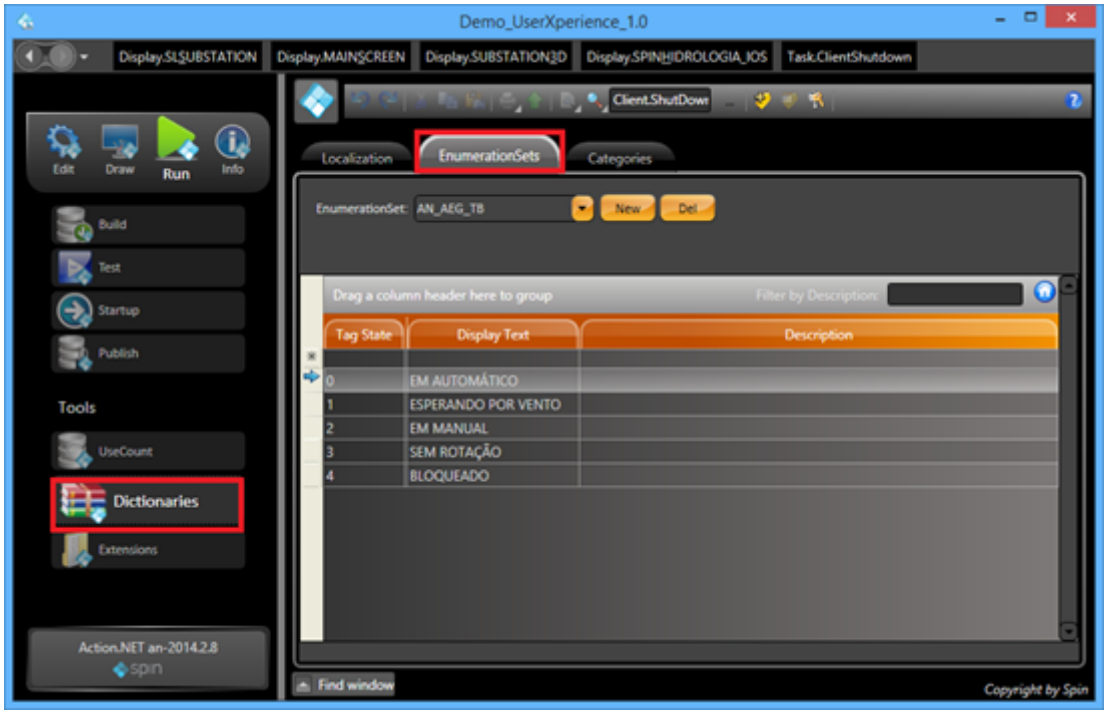


Figure 51 – Dictionaries (EnumerationSets)

4.4.4 Extension

Extensions are library routines used to import and export data and parts of projects.

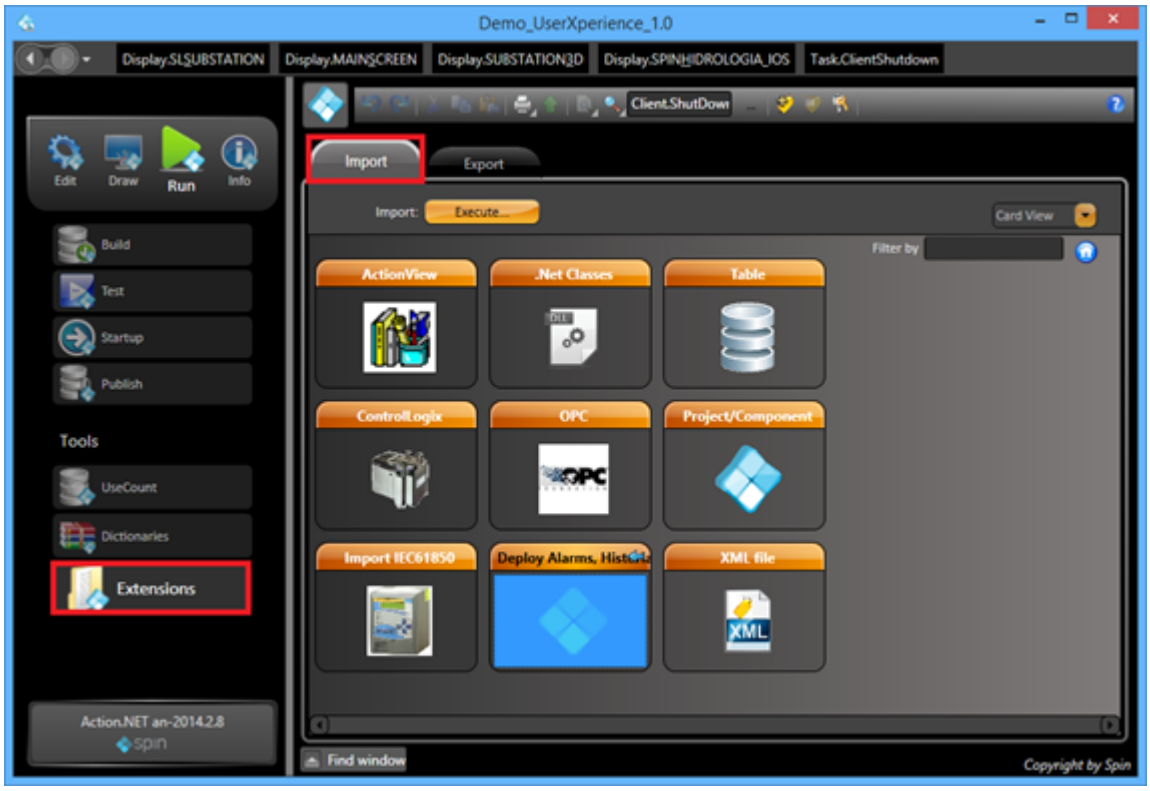


Figure 52 – Import extensions

Using VB .NET or C#, you can create routines for – at design time – import, export and explode data. Below are some routines already available:

- Import Action₀NET projects: allows the import of data from a project (all or part);
- Import Rockwell ControlLogix: it creates the Tag Names, Tag Templates and maps the PLC addresses, all in accordance with the information available at the PLC L5K file;
- Import configuration of an OPC Server: creates the Tag Names, templates tags and maps the addresses OPC client, all according to the information available on an OPC server.
- Import from OSIsoft PI System: by connecting with historian PI System servers, it automatically creates the link with the PI tags, elements and attributes of PI AF servers. When connecting with PI AF, automatically brings the definition of all models and active elements in the selected database, as well as the mapping to communicate with the server.

  NOTE - The IPI Import Wizard is only displayed in licensed premises with the IP connection module enabled.
- Importing Tag Templates in XML: creates the Tag Names, templates tags according to the XML file data structure, it also creates the Edit-Datasets-Files setting to read this file;
- Importing DotNET classes from a DLL: creates Templates Tag matching the selected .NET classes;
- Importing ActionView projects: the Import Wizard can be used to import tags, Alarm Items, Historian tables the points tables I/O of a ActionView project;
- Automated generation projects Action₀NET (Deploy): the Import Wizard can be used to, automatically, generate tags, Alarm Items, Historian tables and Points Tables. To this end, it is necessary to previously create templates, Channels, Nodes and the necessary Historian Tables. You should also create indicative categories to how templates and points should be used. The wizard causes a "burst," that is, creates the aforementioned tables when executed.

- **Tables import**: allows importing selective tables (with easy selection of columns in each table) from a data source to which exist in the environment, available data providers.
- **Export components**: this tool is available to provide the data selection options of this project and create a file with extension **".tcomponent"**. That means it makes possible to export selectively parts of an Action$_0$NET project, which can be imported into another project.

4.5 Scripts Simulating the Process

4.5.1 General Information

The NewDemo has a Script class called ***TagsSimulation*** (figure below) that run every 3 seconds and simulate the values of the three process presented in the demo: Eolic park with six wind towers, hydraulic power plant with three generators and a substation with five bays (one line, 2 transformer and 2 feeders).

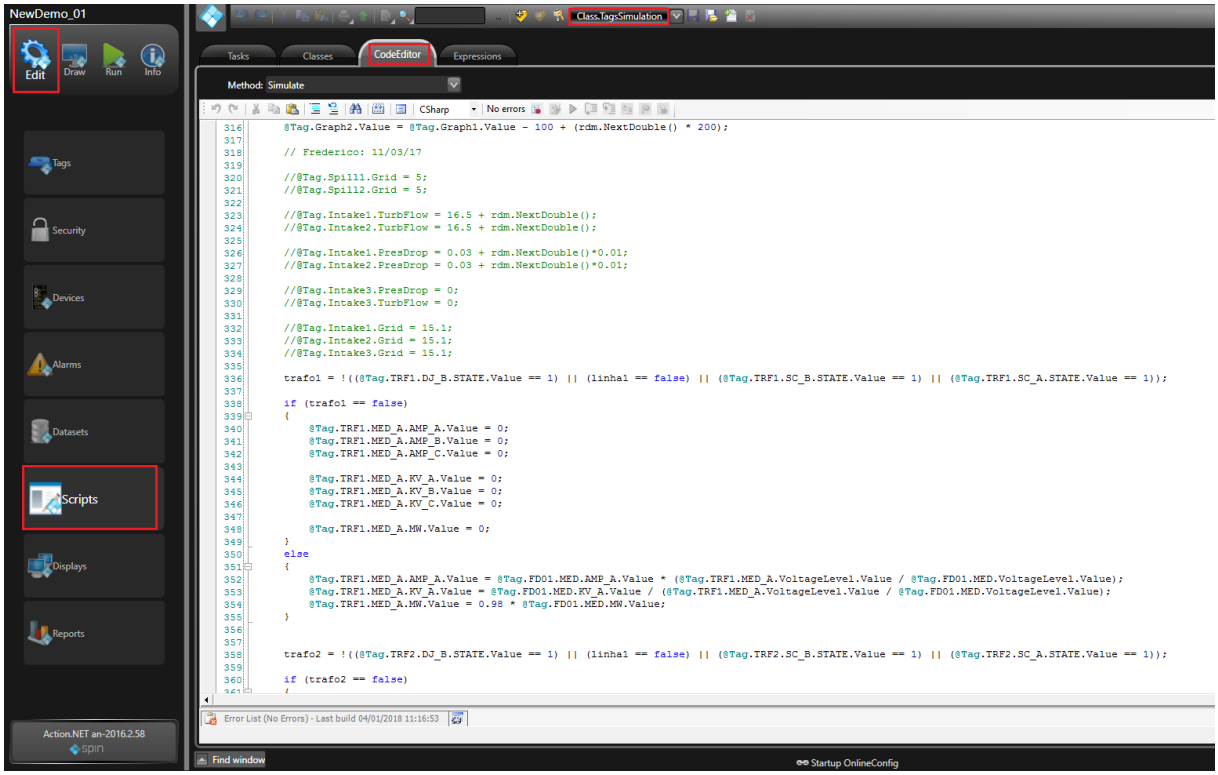


Figure 53 – Class TagSimulation

4.5.2 Simulation of the substation

As you can see in figure below, we have four environments in the substation screen:

(1) **The five bays substation** simulates the three variable of each feeder (current, voltage and power) and two variables of the line and transformers. If you command one circuit breaker, for example, the CB of TR01 bay, the analog variables of that bay and the FD01 bay become zero. Observe that the commanded variable must be simulated. **The five bays substation** has simulated the three variable of each feeder (current, voltage and power) and two variables of the line and transformers. If you command one circuit breaker, for example, the CB of TR01 bay, the analog variables of that bay and the FD01 bay become zero. Observe that the window command needs to have the simulated option select.

When the bay line has no current, it changes the color intensity.

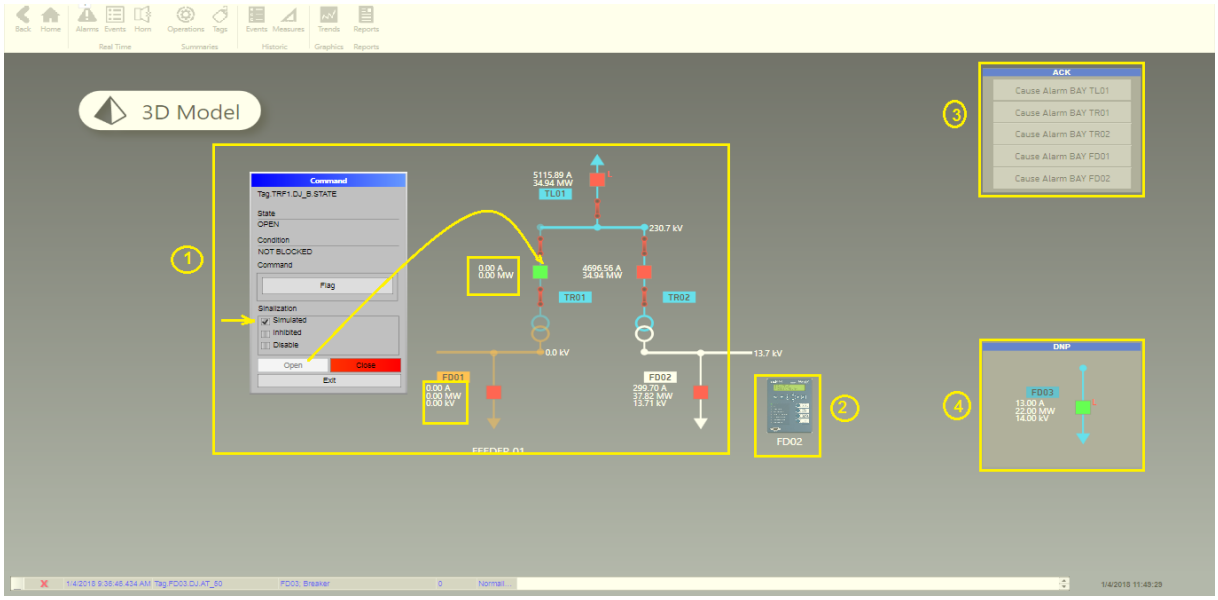


Figure 54 – Substation Simulation

(2) **The relay control panel** presents FD02 information. The led panel shown "Ia" and "Va" values simulated and the led variables are associated to Feeder 2 tags.

(3) Alarm Area Buttons has five buttons that cause and normalize the alarm area.

(4) The Feeder 3 is connected to a DNP3.0 server simulator. See annex 1 to more information.

4.5.3 Simulation of Wind Turbine

The wind turbine status is systematically changed and, depending on this status, several analog variables associated to the turbine are simulated.

4.5.4 Simulation of Hydraulic Power

Once this is an express version, with tags limitation, there are only simulated values of two turbines, varying without any logic.

4.6 Sophisticated Screen Example

The following screen, from a beverage industry, shows the screen building capability available in the Action$_0$NET.

In this screen, there are two buttons, one that turns on / off the process and second that allows you to change production parameters.

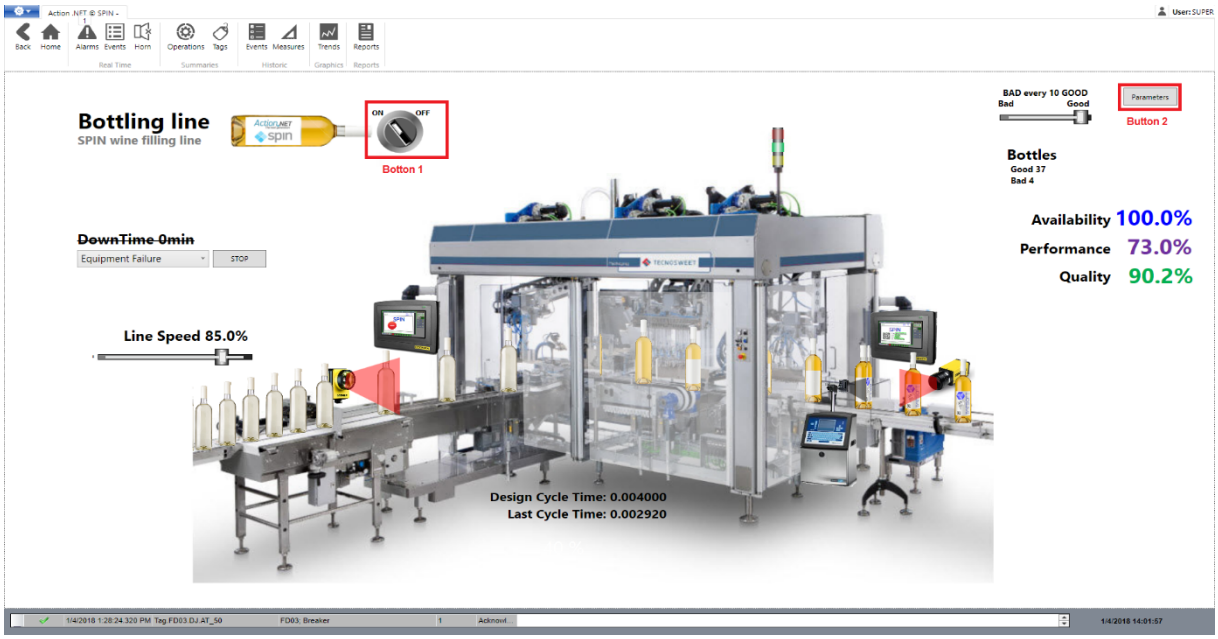


Figure 55 – Screen sophisticated example

5. ANNEX 1 – AXON TEST DNP 3.0

5.1 Configuration of the DNP 3.0 Slave Simulator

To use the Axon Test DNP 3.0 simulator, after install the program just click its shortcut

and select the option in red, showed below.

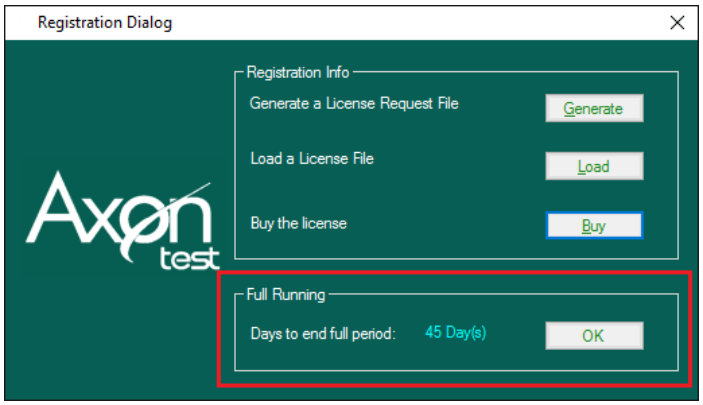


Figure A-1 - Registration Dialog

After select Full Running 45 Days, the software will open the window A-2 where we will select:

(1) Connection: TCP;
(2) Slave – DNP3;
(3) Add DNP3 (will add a new channel).

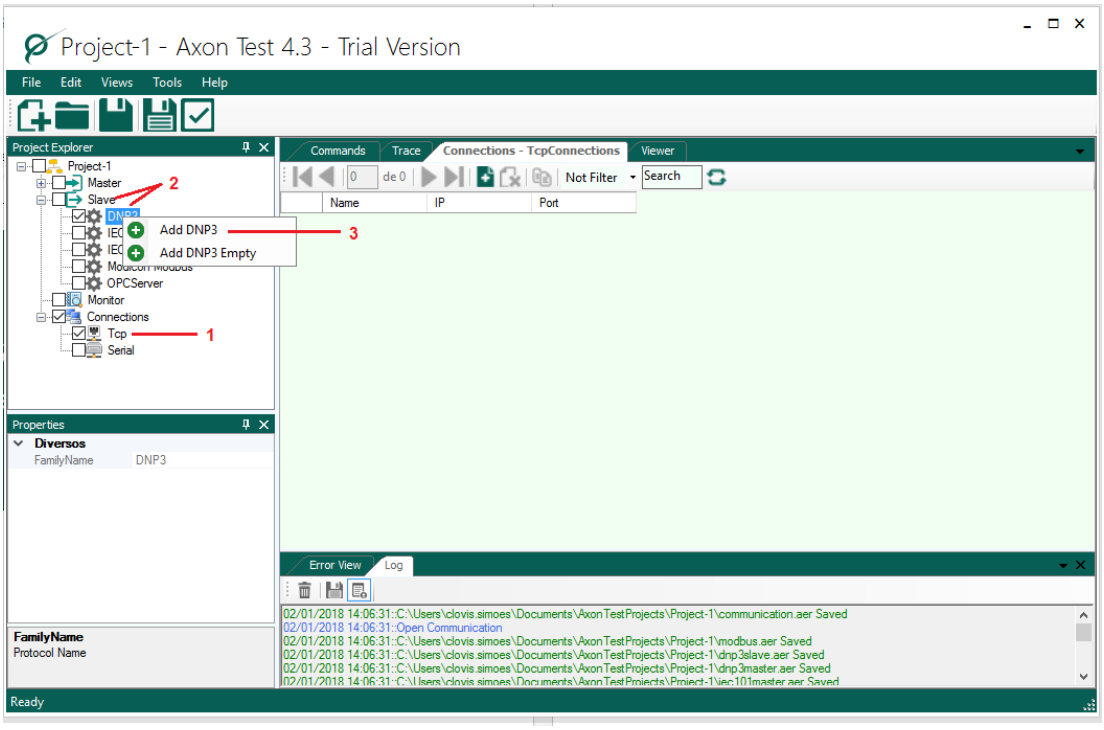


Figure A -2 Create a DNP3 Slave simulator

As a result, a DNP3 Slave channel will be created, as showed in the figure below:

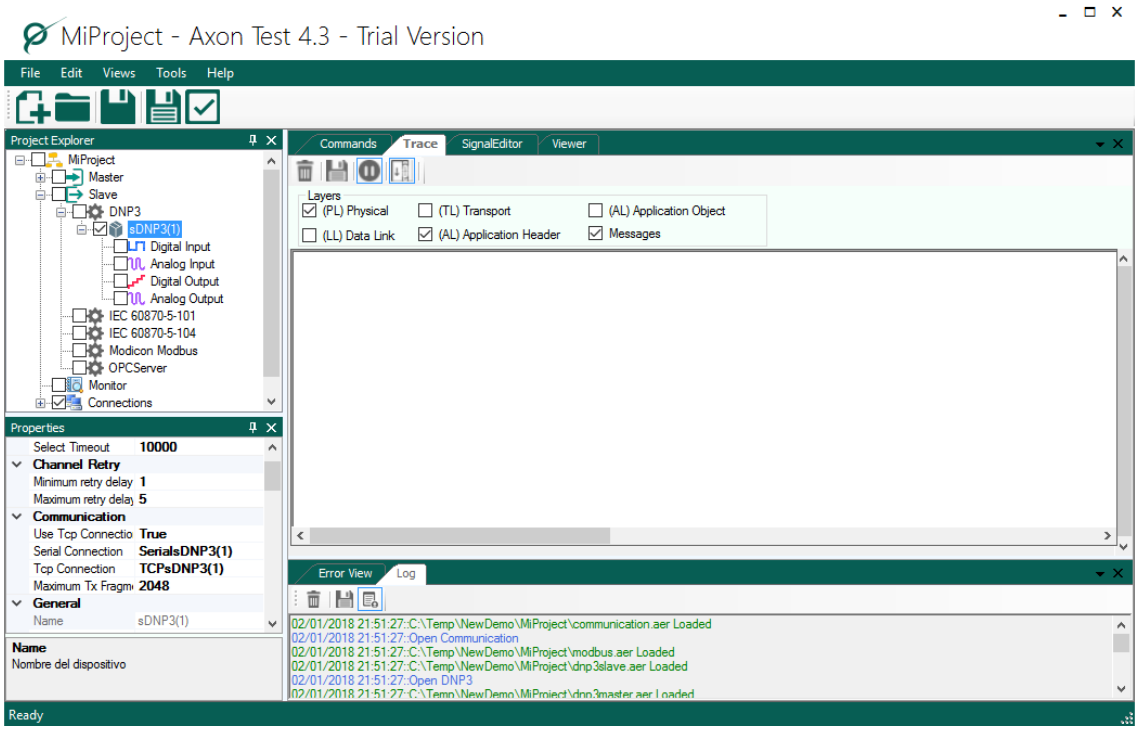


Figure A -3 Channel DNP3 (1) Slave just created

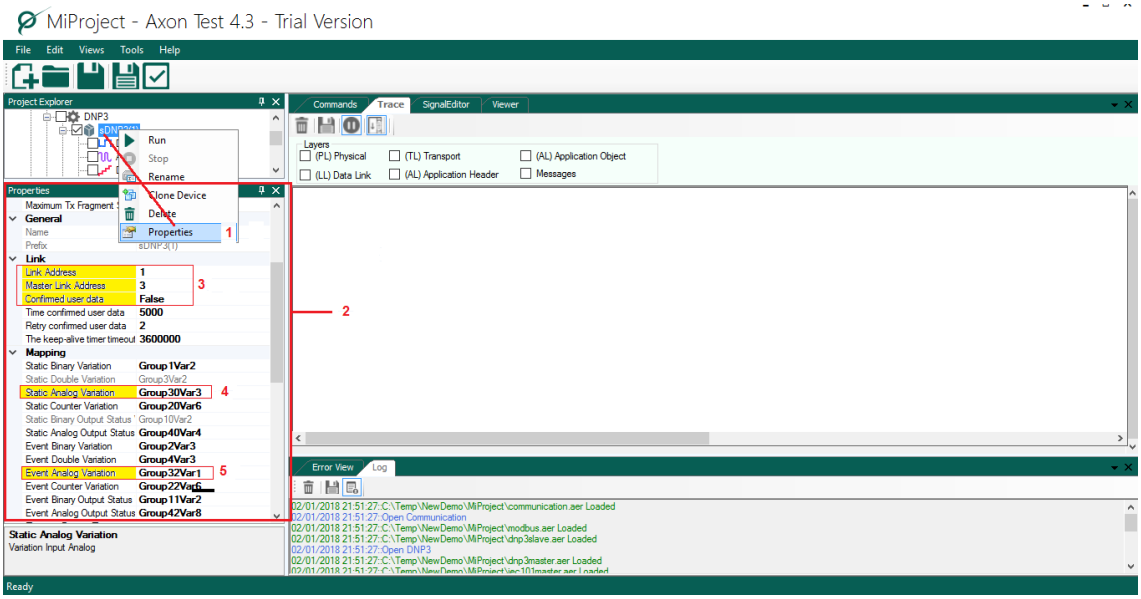


Figure A -4 Channel DNP3 Slave – Changed Properties

Click with the right bottom of the mouse over sDNP3 (1) and select the option properties, as showed in (1) of the figure A-4. The default properties of the channel will be presents as in (2). Change the fields (3), (4) and (5) with the information´s presented in the respective numbers:

(3) Link Address: **1** <Link address defined in Action$_0$NET client channel>
Master Link Address: **3** <master link address defined in Action$_0$NET client

channel>

Confirmed User data: **False** <Defined in the Action0NET client channel>

(4) Static Analog Variation: **Group 30Var3** <Static Analog Variation of Analog Input (AI) type>

(5) Event Analog Variation: **Group32Var1** <Event Analog Variation of AI type>

The figure A-5, show the Action0NET client channel parameters, where we define ***confirmed User Data*** and ***Master Link Address***.

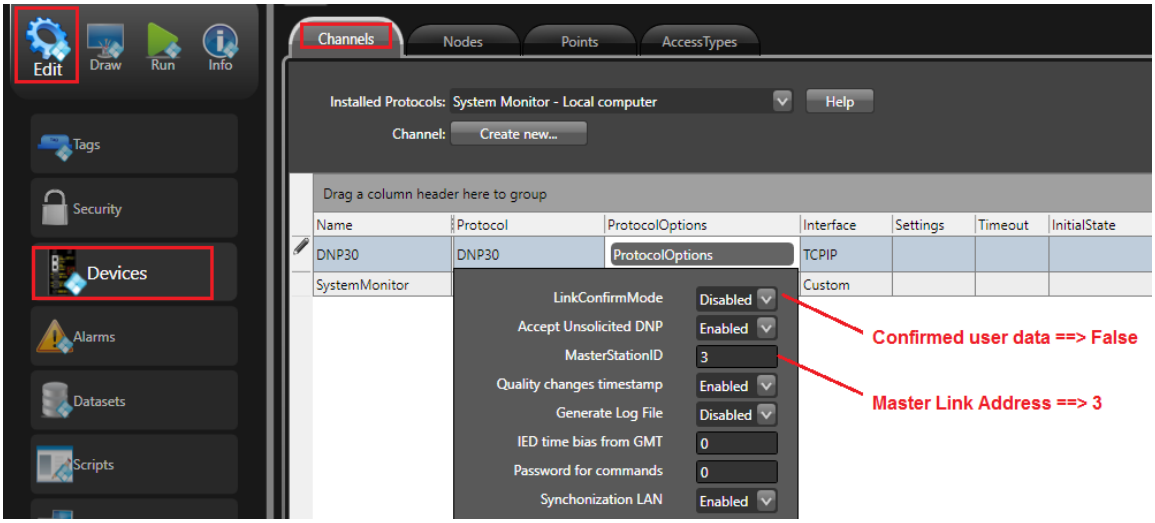


Figure A -5 Action0NET client DNP30 channel definition

The figure A-6, show the Action0NET client node parameters, where we define ***Link Address***.

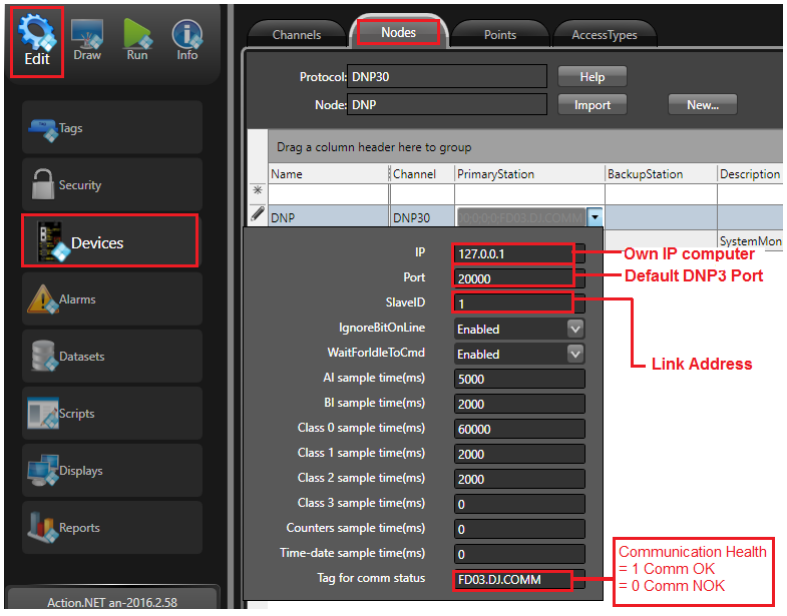


Figure A –6 Action0NET client DNP30 node definition

5.2 I/O Points

As you can see below there are four Analog Input (AI), two Binary Input (BI) and two Controlled Relay Output (CRO) defined in the DNP30 Client Channel. We need to create these points in the DNP3 Slave simulator.

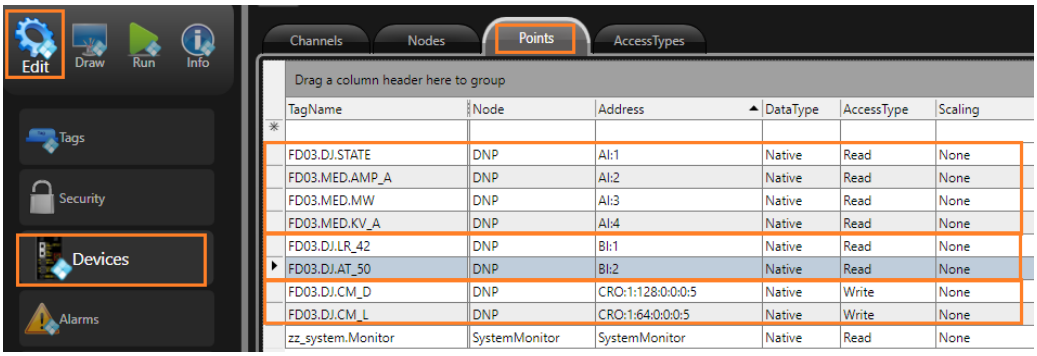


Figure A –7 Action₀NET client DNP30 Points definition

5.2.1 Connection Definition

The TCP connection was defined as the figure below, according Action₀NET client node definition (figure A - 6):

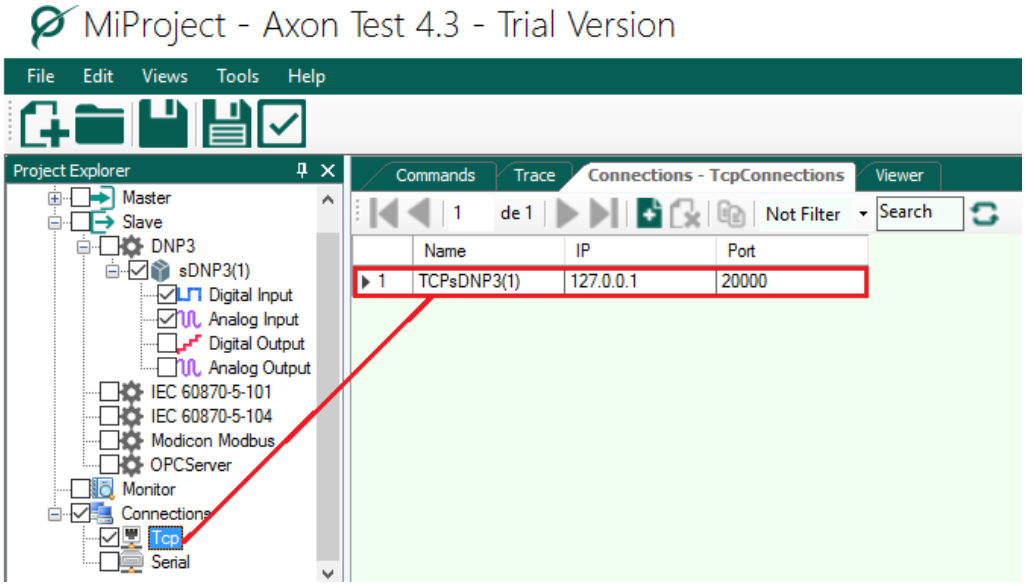


Figure A –8 TCP Connection of the DNP3 Simulator

5.2.2 Creating Binary Input

To create a Binary Input, double click in ***Digital Input (1)*** and do the following in each created line:

(2) Define the Digital Input as DI1, DI2, etc. with the respective telegram address;
(3) Enable simulation of the line;
(4) The simulation option is set to ***Toggle*** (invert the value each simulation time);
(5) The time simulation is set to ***10 seconds***;
(6) In the comment was added the tag of the variable associated to each line.

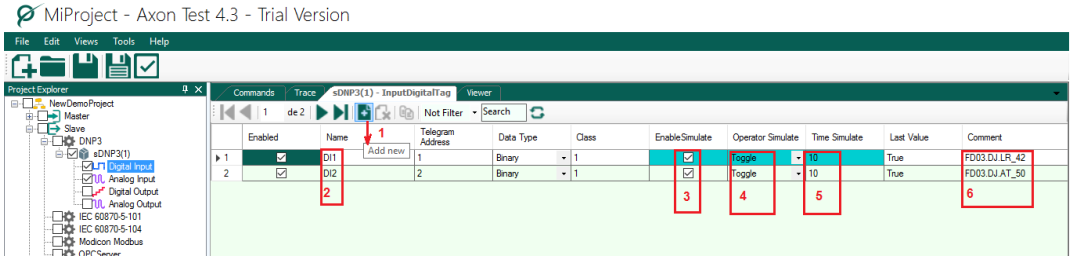


Figure A –9 Binary Input of the DNP3 Digital Simulator

5.2.3 Creating Analog Input

To create Analog Input, double click in ***Analog Input*** and do the following in each created ( ) line:

(1) Define the Analog Input as AI1 to AIn with the respective telegram address;
(2) Enable simulation of the line;
(3) Set time simulation to ***5 seconds*** (each 5 seconds the value will be incremented by 1);
(4) The simulation option is set to ***Increment*** (will vary from lower to upper limit);
(5) Each simulation add 1 (step) to current value;
(6) Define to each line the low-high limit. Observe that <FD03.DIG.STATE> vary from:
    0 – undefined;
    1 – Opened
    2 – Closed
(7) In the comment was added the tag of the variable associated to each line.

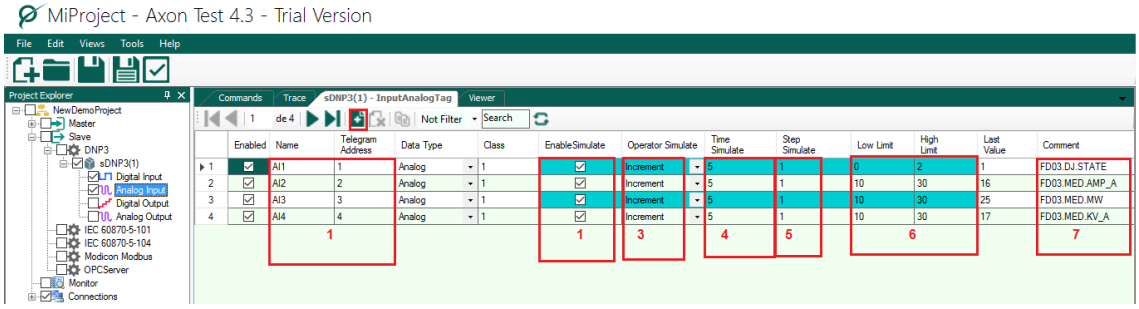


Figure A –10 Analog Input of the DNP3 Digital Simulator

5.3 Running DNP3 Slave Simulator

To run the simulator, click with the right bottom of the mouse over sDNP3 (1) and select the option ***Run,*** as sowed in the figure below:

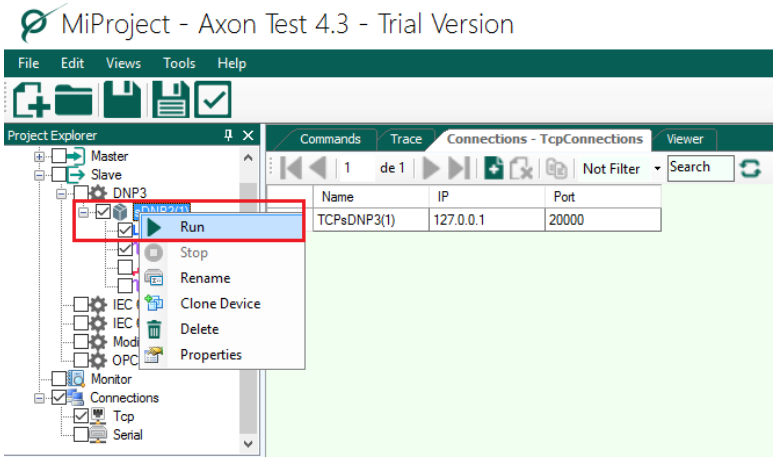


Figure A –11 Running the DNP3 slave simulator

The figure A -12 show the Action₀NET running with the DNP3 slave simulator.

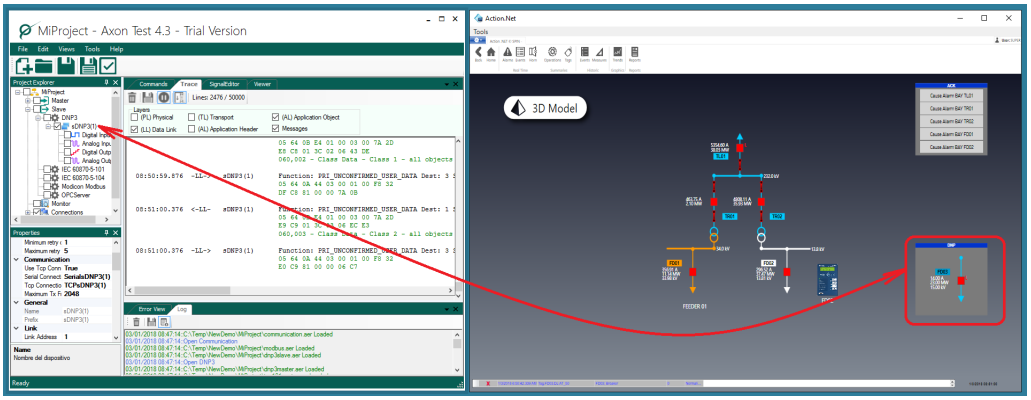


Figure A –12 Action₀NET running with DNP3 slave simulator

If we stop the DNP3, the quality of read variables became zero (≠ 192) and the visualized variables are crossed out, as showed below:

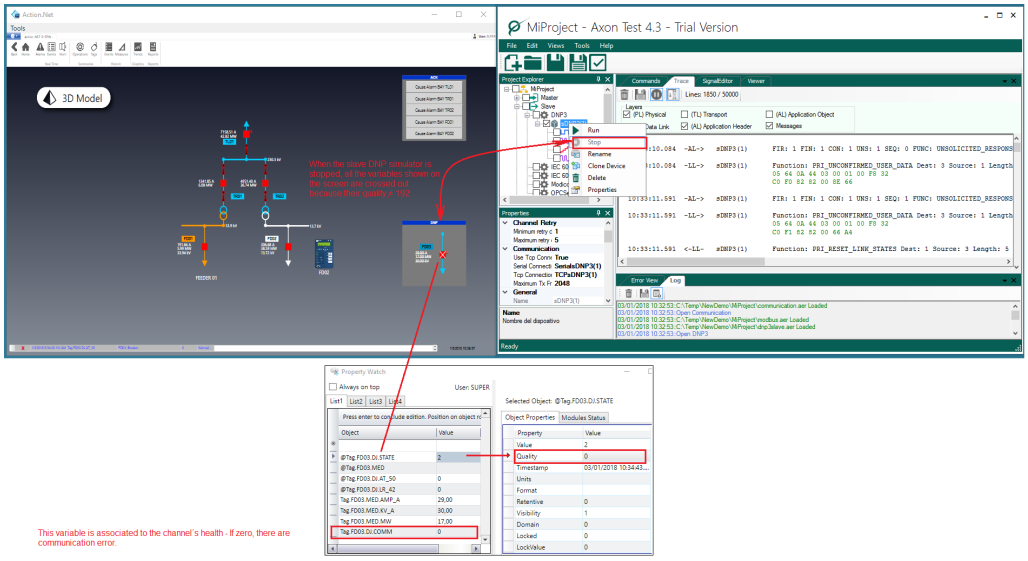


Figure A –13 Action₀NET running with DNP3 slave simulator stopped