Example of a Smart Object Template with the motor symbol and its released properties.

SERIES: EFFICIENT ENGINEERING WITH ZENON
PART 2

HOW TO MANAGE PROJECTS CENTRALLY WITH ZENON

IU

Published in
INFORMATION UNLIMITED
THE COPA-DATA MAGAZINE
No. 36, November 2020

www.copadata.com/iu

The first part of this series focused on the fundamental philosophy of zenon. It looked at the concepts based on this philosophy and discussed topics such as how to manage elements centrally. We will now use a practical example to show you how centralized management can be optimized with the aid of data types, symbols, links and – with the new zenon version 8.20 – Smart Objects and Smart Object Templates.

Let's imagine you want to visualize three motors of the same type – used for conveyor belts in a production facility – in your zenon project. These motors are switched on and off via an HMI and supply the system with information about the operating status, speed and temperature. To map the motors in zenon, you will need variables, functions and one or more screens. This is where zenon data types and symbols come into play, since they offer numerous possibilities when it comes to engineering.

WHAT ARE DATA TYPES?

Data types allow you to centrally define the properties of variables – for example, measuring ranges or limit values. In this example, you create multiple simple data types: "MotorTemperature" (BYTE), "MotorState" (BOOL), "MotorSpeed" (UINT). For "MotorTemperature", you enter "°C" as the measuring unit and create a limit value for "critical" at 80°C and "overheating" at 100°C. You then create a structure data type called "Motor" with the structure elements "Temperature", "Speed" and "State", which are each based on the simple data types you've just created. Make sure that "Link data type" is selected under "Structure options" so that changes made to the data type later will automatically apply to the structure elements.

The next step is to create the variables for the three motors with the names "Motor_Band1" (..2, ..3); each of these is based on the "Motor" data type. A total of nine variables have now been created and activated automatically with your specifications.

USING SYMBOLS SENSIBLY

A centrally defined symbol is used for the graphical visualization of the motor. In this case, it is a good idea to use a button with labeling and value displays for temperature and speed. These are to be linked to the corresponding variables from the "Motor_Band1" structure data type.

Drag the symbol you have just created into a screen where it can be used directly for "Motor_Band1". For "Motor_Band2", drag the symbol into the screen again and replace the variables and functions via a linking rule in the dialog that opens automatically. Alternatively, you can open this dialog under the linking rule in the property window. In this example, it is sufficient to use "*1*" as the source and "2" as the target for the replacement function. You can check the result of this linking rule immediately using the Preview button.

CUSTOMIZING SYMBOL PROPERTIES

Properties such as the color or text of the motor are not taken into account by the linking rule, however. To enable individual customization of these properties directly in the screen and whenever a symbol is used, you can release individual properties of elements within a symbol.

To do this, open the symbol, highlight the element and release the desired property (e.g. Text) using one of the three options below. You must always be guided by how the property is labeled in the property window.

- Drag the property into the area below the the symbol's drawing area
- Open the context menu by right clicking and select Release 'Text'
- Highlight the property and select Release Property in the toolbar

If you select the symbol that is now on the screen, you will see a new node – "$_<Elementname>" – in the tree view in the property window, which contains the property that has just been released. This allows you to overwrite individual definitions locally whenever the symbol is used. At the same time, it reduces complexity for the engineer, as only the properties that are relevant to them are displayed.

If you want to switch the motor on and off directly from this symbol, you need to create two buttons and two Write set value functions with the names "Motor_Band1.Start" and ".Stop" on the "Motor_Band1.State" variable. Saving

the symbol automatically updates all points of use in the screen. If you now open the linking rule dialog again for "Motor_Band2", you will see in the target column that the Start and Stop functions both have a "(?)" at the end. This means that these functions do not exist in the project. As the user of this symbol, you therefore need to know which variables and functions the symbol expects, create them in the project and adjust the function parameters.

THERE MUST BE AN EASIER WAY!

The new zenon version 8.20 makes life easier in a number of ways. You can now create Smart Objects and Smart Object Templates which, alongside displaying symbols and screens graphically, also bring together data types, reaction matrices, functions, interlocks, files and other elements.

The zenon Editor features a separate area for the Smart Object Templates. In the area on the left-hand side, create a new template called "Band" (belt). A tree will now appear on the right-hand side of the window showing the available zenon modules. Here, you should create the same objects as described above, except that "Band1" can be omitted from the names in all cases.

In the symbol, you should release the released property "Label" for the Smart Object Template as well by highlighting the property in the symbol editor and releasing it again by clicking on the button in the toolbar. Now switch back to the project tree, select the Smart Objects node and create a new Smart Object based on the template you have just created. In the list, you will see the Smart Object and, below this, the symbol you have defined. If you drag this symbol into a screen and then open the linking rule dialog again, the correct rule will already be implemented for the replacement and all targets will be found in the preview.

Thus, when you create a Smart Object, all engineered variables, functions, etc., are automatically created in the project and adapted at all points of use.

HOW TO EXPAND SMART OBJECT TEMPLATES FOR NUMEROUS PURPOSES

If you also need motors in other assembly groups in your facility, you can add the Smart Object Template as a reference in other Smart Object Templates and use it as often as you like. And if you want to add a detail view with trend curves to the motor later on, for example, you can easily do so with Smart Object Templates as well. Simply create a new screen and engineer one trend element for "Motor.Speed" and another for "Motor.Temperature" along with a button with the Close Frame function. As soon as you save the Smart Object Template, all Smart Objects in the project will be updated and the necessary functions and screens will be created. The new functionality will be available throughout the entire project instantly.

DISCOVER THE BENEFITS OF THE NEW ZENON VERSION 8.20 NOW

The concept of linking, as explained above in relation to symbols, is also available in the Screen Switch function. With Smart Objects, linking is automatically adapted when the Screen Switch function is created so that the correct variables are always linked in the detail screen.

This part of the series has explained how you can use the zenon philosophy to create and reuse reusable components with the help of data types, symbols, linking and – with the new version – Smart Objects. The next part of the series will explain the concept of linking in more detail.

GERO GRUBER
Product Manager

As Product Manager and Product Owner for the zenon Software Platform, I am particularly interested in the user interface and the interaction design of the whole platform, as well as the graphical visualization in zenon Runtime.