



The CODESYS Application Composer: From module creation to plant engineering

CODESYS Users' Conference 2014, Dieter Hess

**1**

What is the CODESYS Application Composer?

2

How to use the CODESYS Application Composer

3

Demo

What is the CODESYS Application Composer?

- The CODESYS Application Composer is a development tool for efficient creation of applications consisting of predefined function blocks.
- The CODESYS Application Composer allows to engineer and parameterize complete controller applications out of previously created software modules.
- It is possible to create a whole working application, without programming.



What is the CODESYS Application Composer?

In detail...

- Two aspects of application engineering:
 - Development of technology modules
(e.g. control algorithms, motion control etc.)

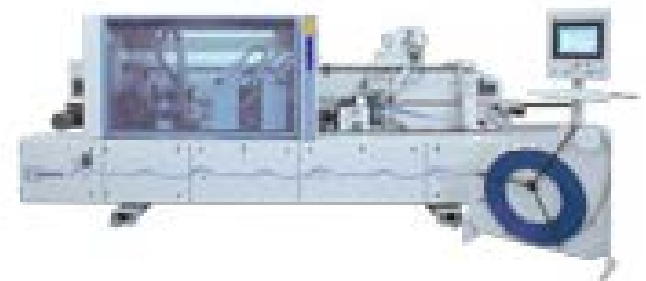
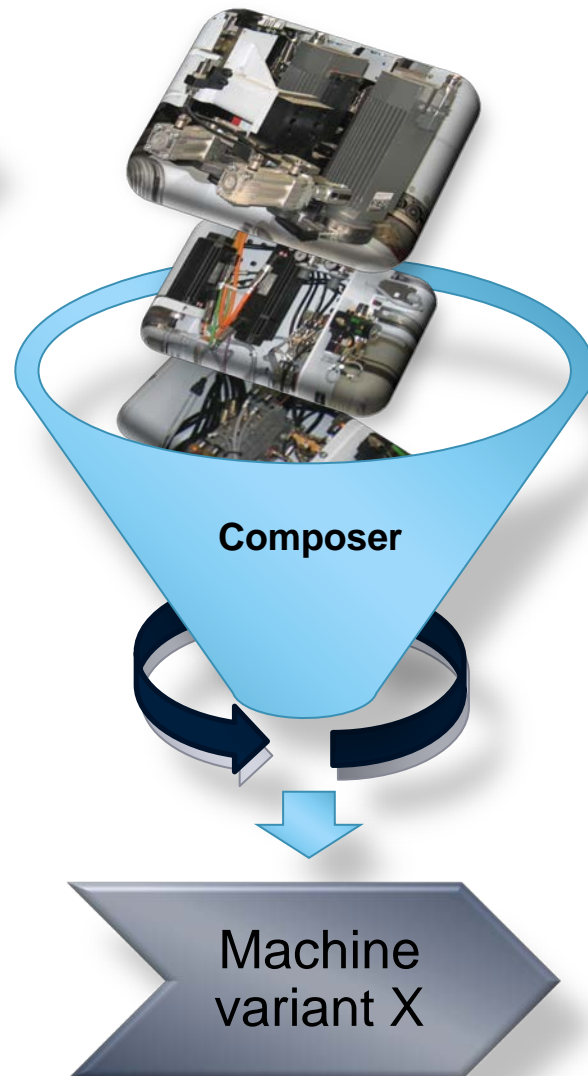
=> Demanding, innovative development work
 - Machine engineering (“Composing”)
 - Selection and parameterization of technology modules
 - I/O assignment
 - Distribution to controllers
 - Standard infrastructure (diagnosis, alarms etc.)
=> Routine tasks with high rationalization potential



What is the CODESYS Application Composer?



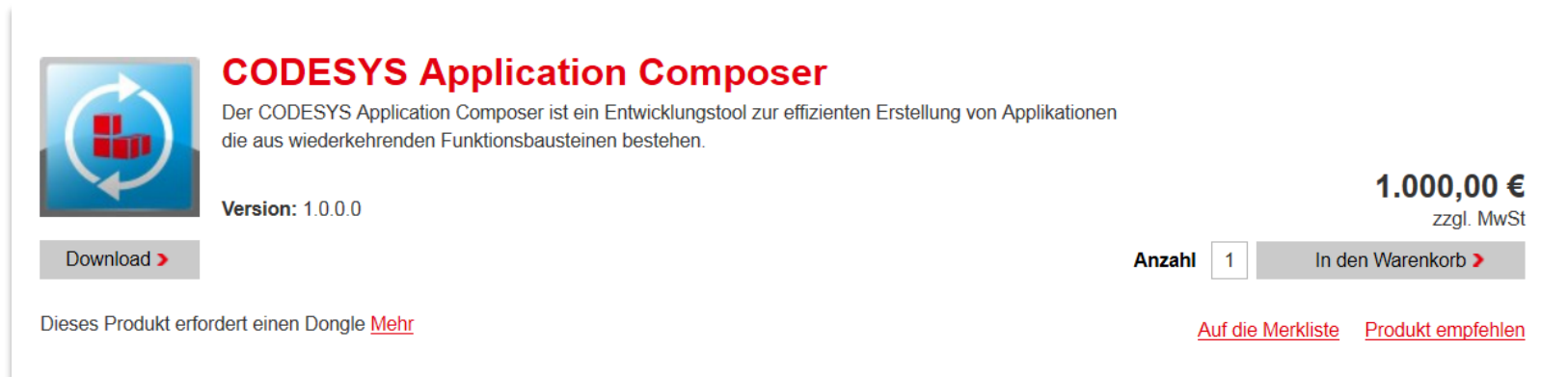
Technology
modules




What is the CODESYS Application Composer?

In which form is the CODESYS Application Composer supplied?

- In the CODESYS Store, you can purchase a CODESYS Package along with a ticket allowing for the installation of the license key on a Wibu dongle.

A screenshot of a product listing for "CODESYS Application Composer" in an online store. The product image shows a blue square with a white circular arrow and three red cubes. The title "CODESYS Application Composer" is in red. Below it, a description states it is a development tool for creating applications from reusable function blocks. The version is listed as 1.0.0.0. The price is 1.000,00 € including VAT. There is a "Download" button and a quantity selector set to 1. A "In den Warenkorb" button is also present. At the bottom, there are links for "Auf die Merkleite" and "Produkt empfehlen". A note at the bottom left states that the product requires a dongle.

 **CODESYS Application Composer**
Der CODESYS Application Composer ist ein Entwicklungstool zur effizienten Erstellung von Applikationen die aus wiederkehrenden Funktionsbausteinen bestehen.
Version: 1.0.0.0

Download >

Anzahl In den Warenkorb >

1.000,00 €
zzgl. MwSt

Dieses Produkt erfordert einen Dongle [Mehr](#) [Auf die Merkleite](#) [Produkt empfehlen](#)

- With this workplace license, you can create and use modules.

1

What is the CODESYS Application Composer?

2

How to use the CODESYS Application Composer

3

Demo

How to use the CODESYS Application Composer

Example: Gas station

CODESYS Users Conference 2014

CODESYS GasStation

GasStation OPACServer

NIGHTTIME

Optmode Spots Frame

CODESYS

Diesel	0,39	l
Super 95	1,53	l
LPG	0,89	l



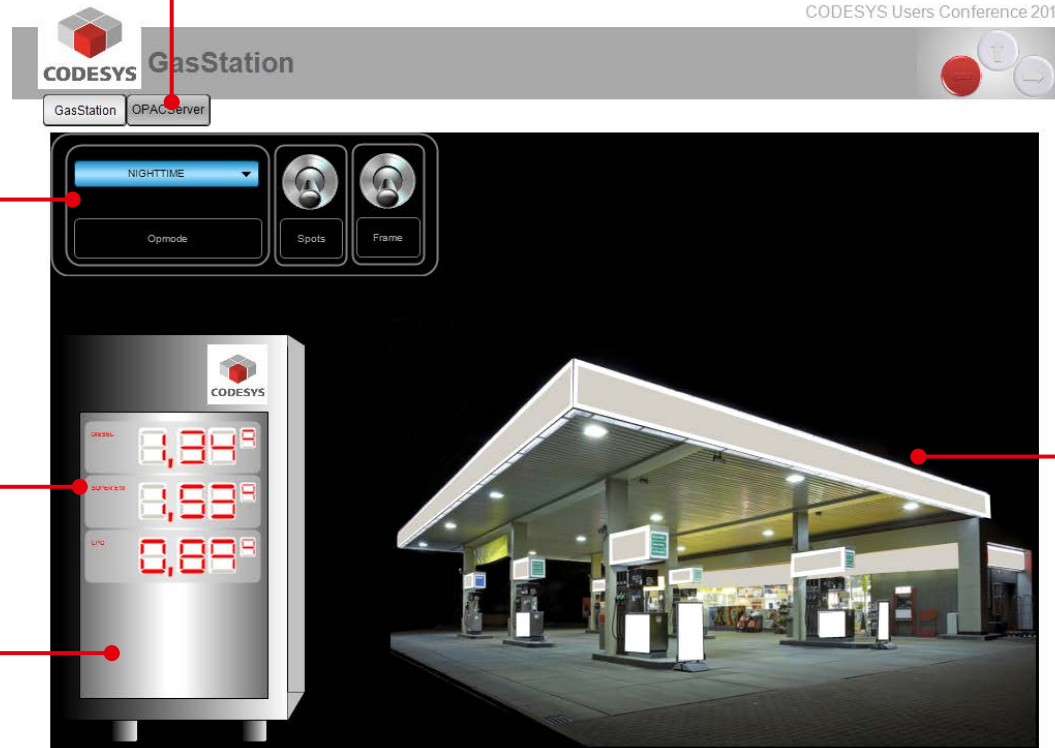
13:34:30

© 3S-Smart Software Solutions GmbH

How to use the CODESYS Application Composer

Units of a gas station

Central target prices



The screenshot shows the CODESYS GasStation control interface. At the top, there is a header with the CODESYS logo, the text 'GasStation', and 'CODESYS Users Conference 2014'. Below the header, there are two tabs: 'GasStation' and 'OPACServer'. The main interface area contains several controls: a 'NIGHTTIME' dropdown menu, an 'Optmode' button, two 'Spots' buttons, and a 'Frame' button. On the left side, there is a 'Price information system' with a display showing three prices: 'DIESEL' at 0,39, 'SUPER 95' at 1,53, and 'LTD' at 0,89. Below the price display is a 'with display' label. On the right side, there is a photograph of a gas station at night with its canopy illuminated. A 'gas station with illumination' label points to the photograph. The CODESYS logo is also visible on the price display unit.

13:34:30

© 3S-Smart Software Solutions GmbH



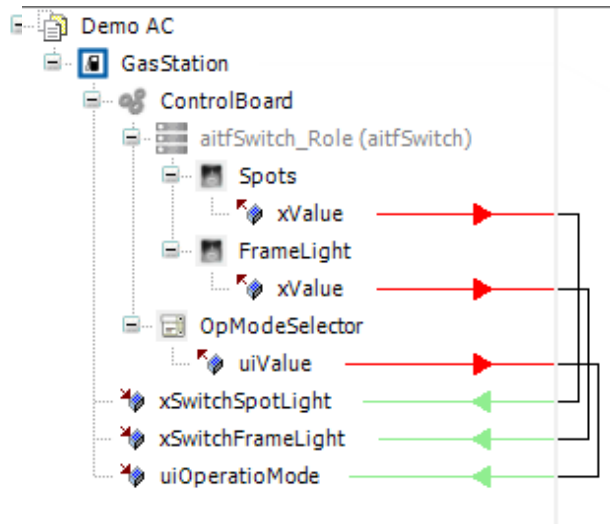
How to use the CODESYS Application Composer

Situation

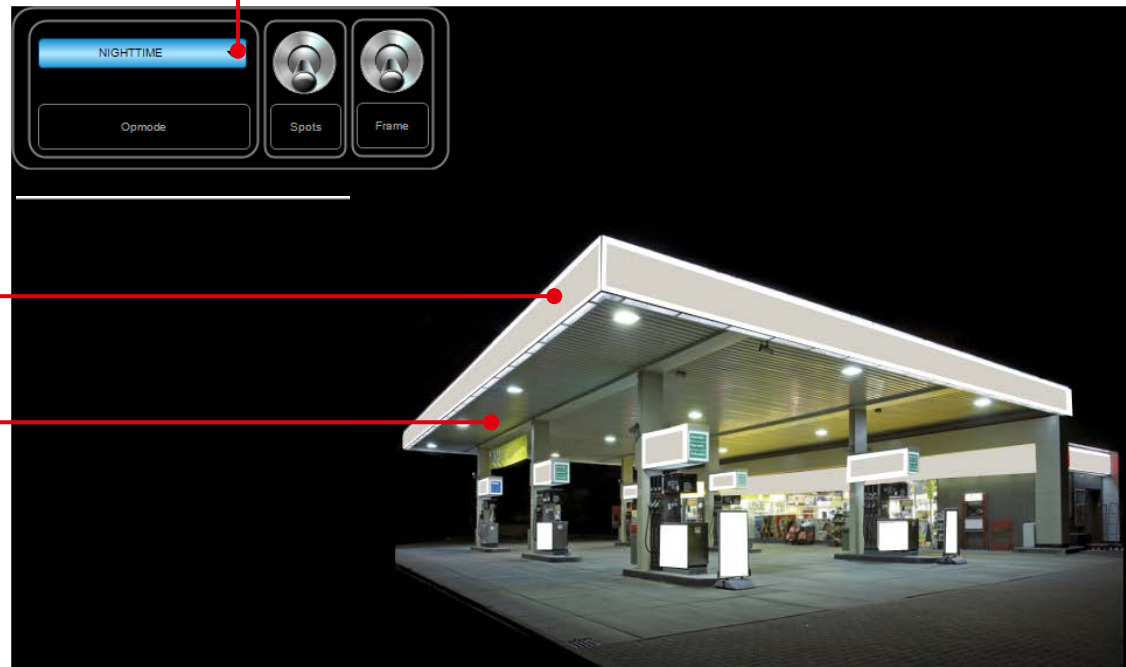
- As a system supplier, you have a wide product variety.
- You offer various configuration options and your customers' wishes can be realized at very short notice - even during commissioning.
- Even after many years, phased-out products should not cause any problems. You need to support new, compatible devices.
- Your service staff must engineer new plants or maintain existing plants with only little training.

How to use the CODESYS Application Composer

Development gas station - control center



Control center

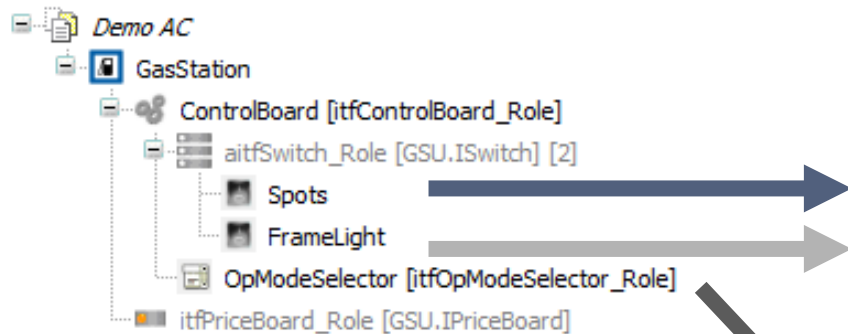



Front illumination

Spots

How to use the CODESYS Application Composer

Development gas station - control center




GasStation.ControlBoard.Spots x

Parameter	E/A	HMI	Information			
Parameter	Typ	Wert	Beschreibung	> Min	< Max	
sID_Name	STRING(25)	'Spots'	sID_Desc			

GasStation.ControlBoard.FrameLight x

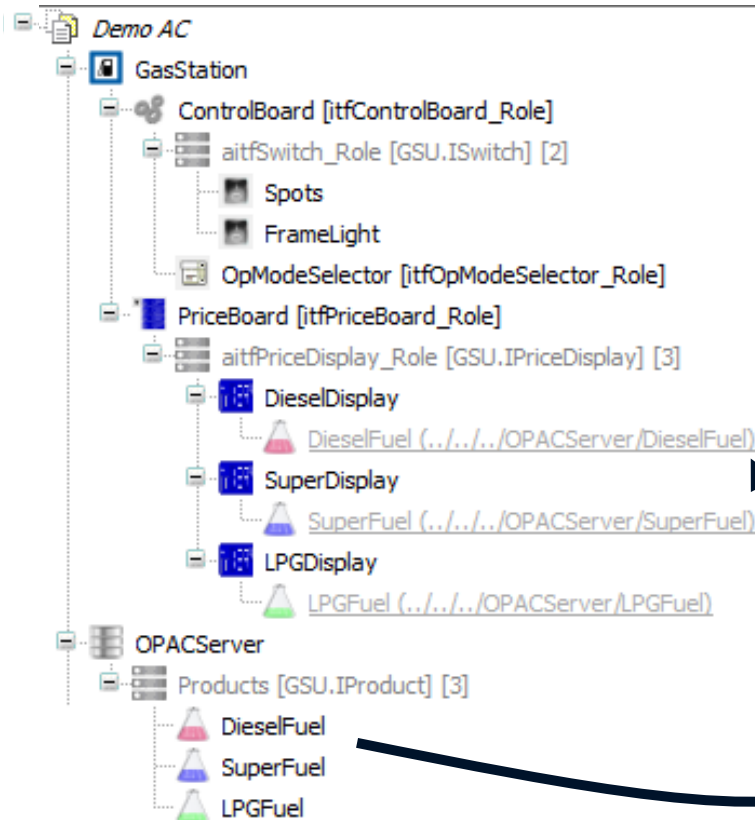
Parameter	E/A	HMI	Information			
Parameter	Typ	Wert	Beschreibung	> Min	< Max	
sID_Name	STRING(25)	'Frame'	sID_Desc			

GasStation.ControlBoard.OpModeSelector x

Parameter	E/A	HMI	Information			
Parameter	Typ	Wert	Beschreibung	> Min	< Max	
sID_Name	STRING(25)	'Opmode'	sID_Desc			

How to use the CODESYS Application Composer

Development gas station

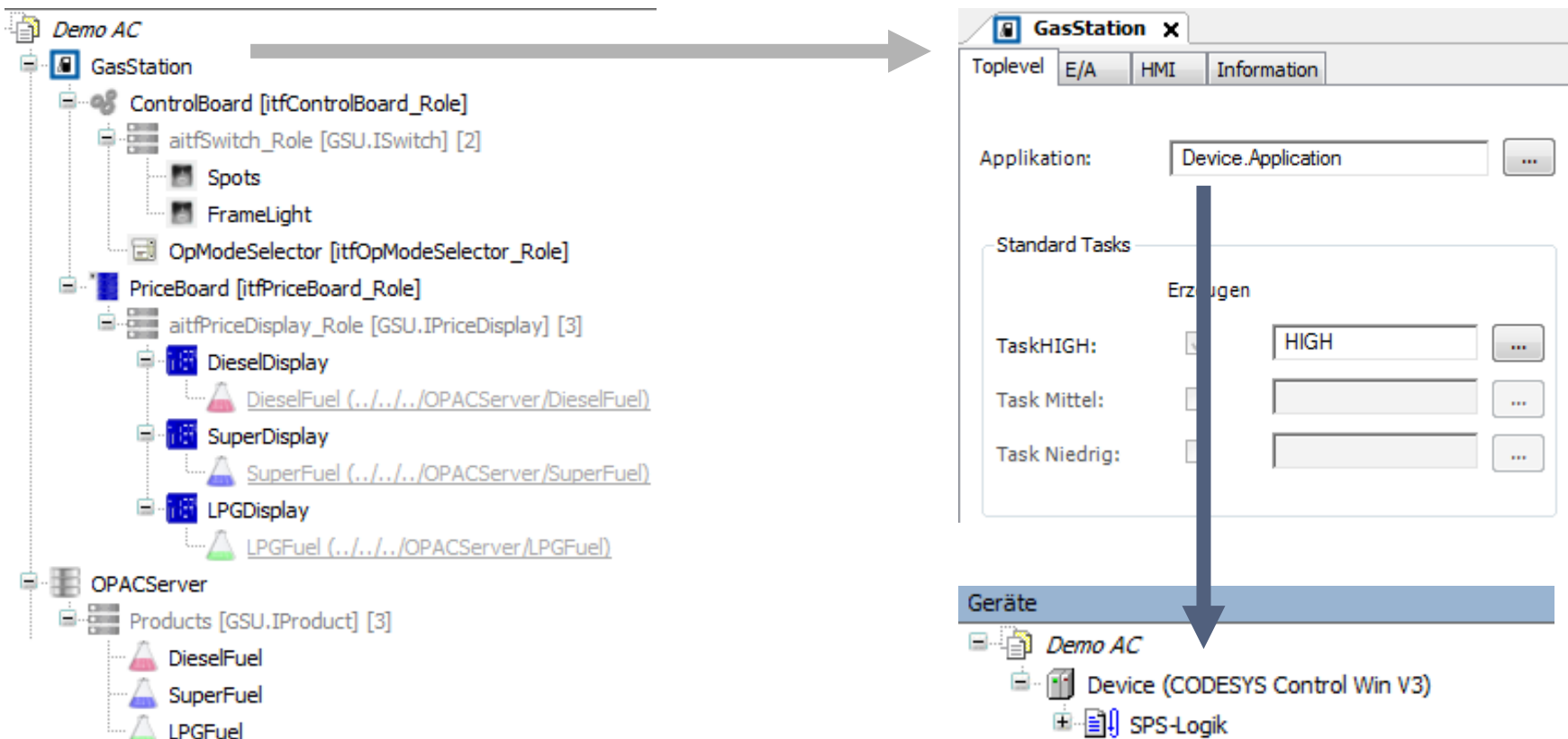


Drag and Drop

How to use the CODESYS Application Composer

Development gas station

- In the top level object you decide on which controller the code will be processed.



The image illustrates the configuration of a gas station application in the CODESYS Application Composer. On the left, a project tree for 'Demo AC' shows the 'GasStation' object selected. The tree includes components like 'ControlBoard', 'PriceBoard', and 'OPACServer'. A grey arrow points from the 'GasStation' object in the tree to the configuration window on the right.

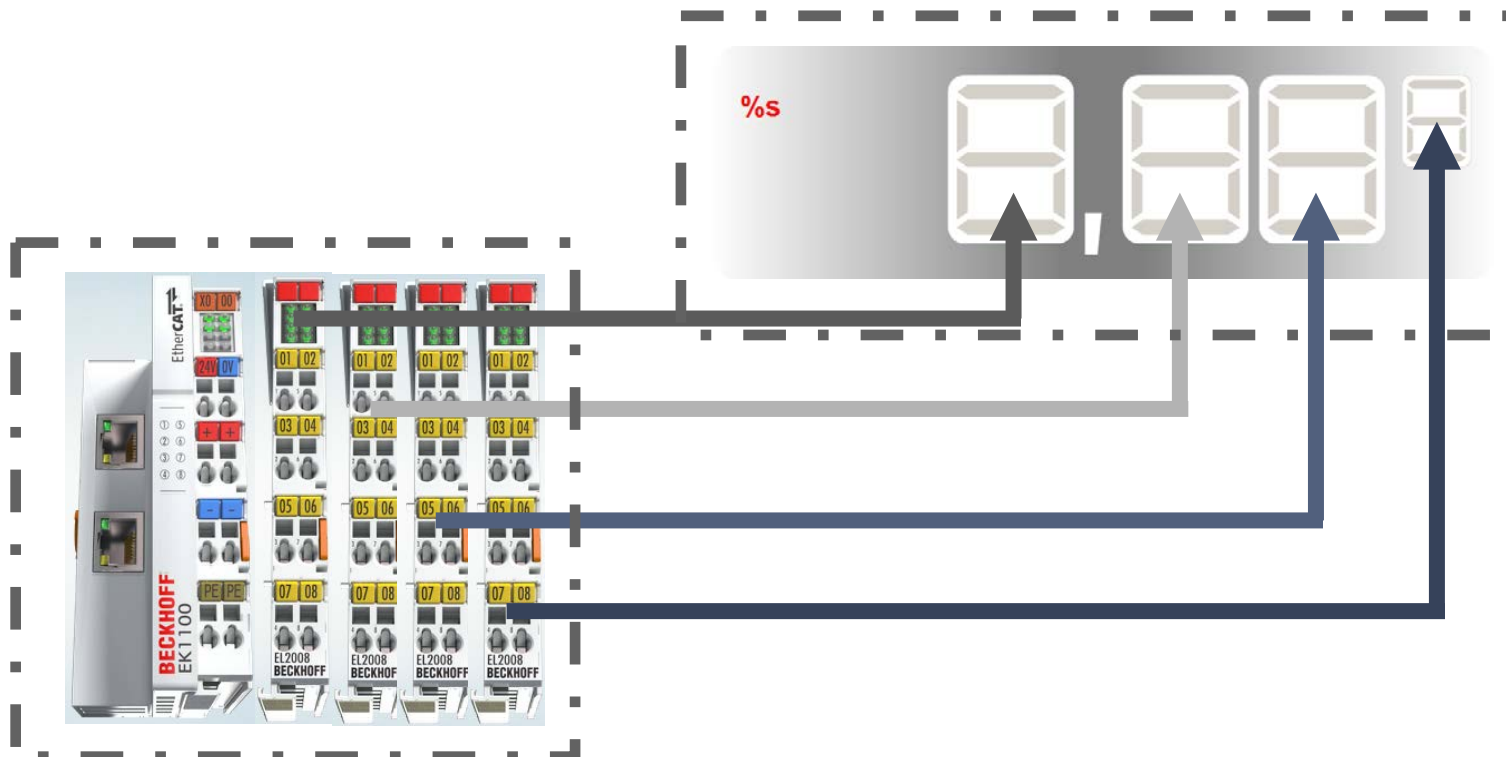
The configuration window, titled 'GasStation', has tabs for 'Toplevel', 'E/A', 'HMI', and 'Information'. The 'Toplevel' tab is active, showing the 'Applikation:' field set to 'Device.Application'. Below this, the 'Standard Tasks' section is visible, with a 'Erzeugen' (Generate) button and three task entries: 'TaskHIGH', 'Task Mittel', and 'Task Niedrig', each with a corresponding input field and a menu button.

At the bottom, the 'Geräte' (Devices) section is highlighted in blue. It shows the 'Demo AC' project containing a 'Device (CODESYS Control Win V3)' and 'SPS-Logik'. A blue arrow points from the 'Applikation:' field down to the 'Device' entry in the 'Geräte' list, indicating the selection of the target controller.

How to use the CODESYS Application Composer

The device configuration including I/O mapping can be generated automatically.

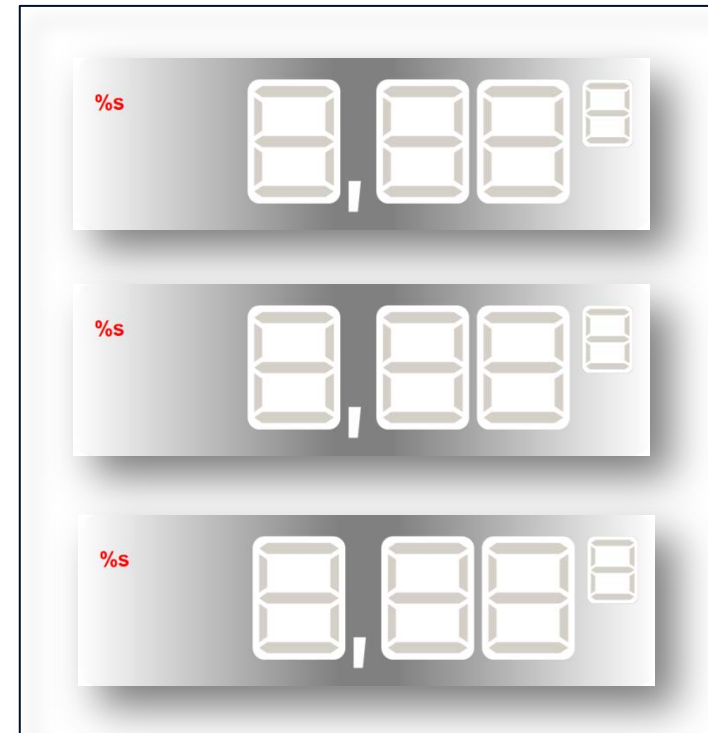
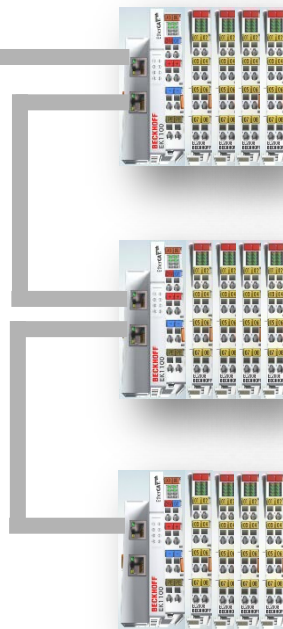
- Each display on the display element could be realized using an EtherCAT slave EK1100 + an EL2008 clamp per digit.



How to use the CODESYS Application Composer

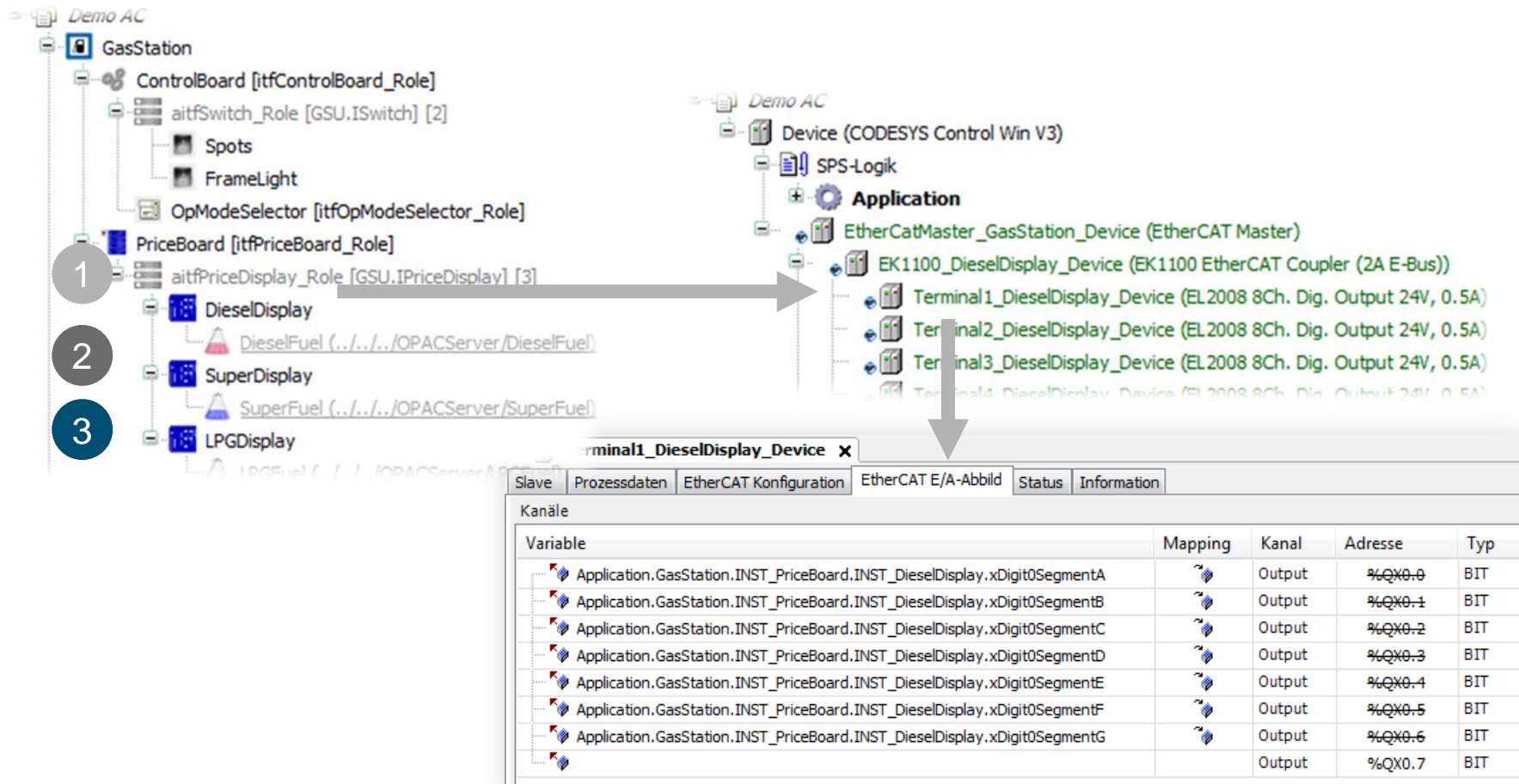
Three displays require a higher configuration effort.

- The Composer can take over this task for you.



How to use the CODESYS Application Composer

The Composer is ideal for recurring configuration tasks.



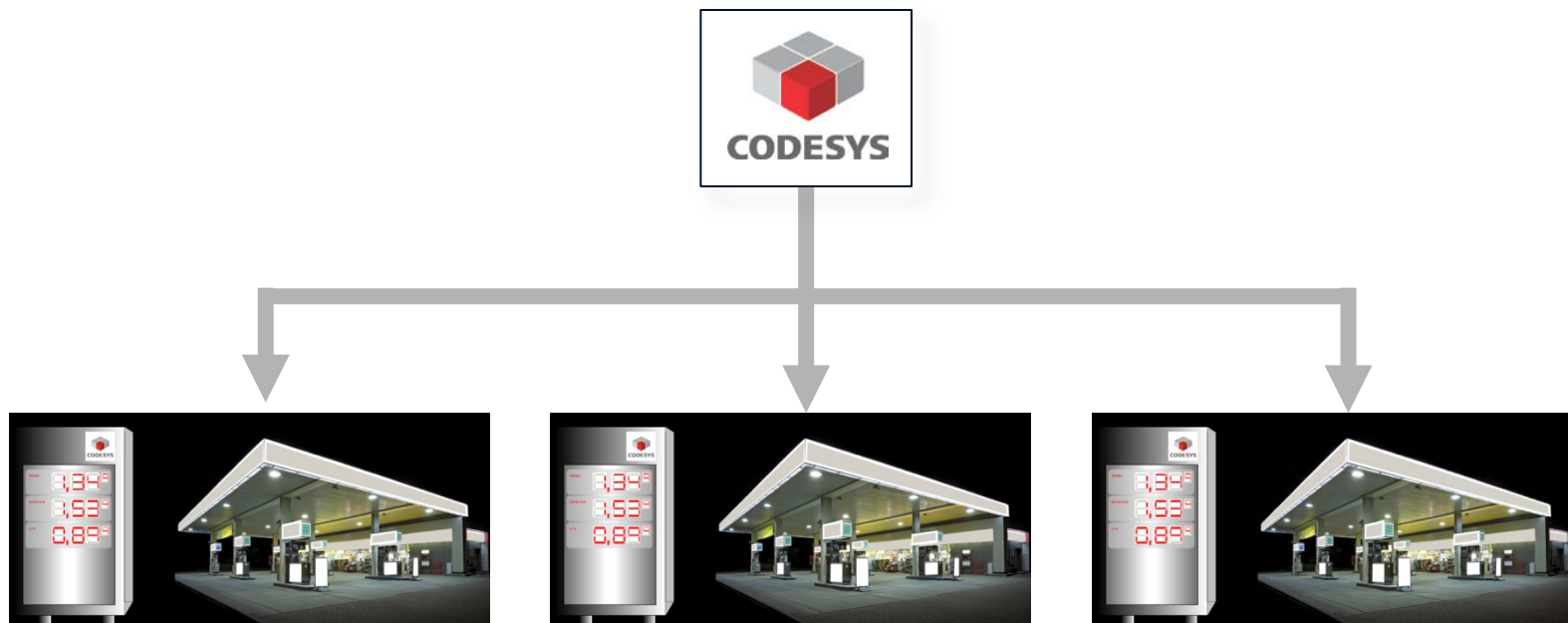
The screenshot illustrates the configuration process in the CODESYS Application Composer. On the left, the project tree shows a 'GasStation' object containing a 'PriceBoard' and an 'aitfPriceDisplay_Role' (with a count of 3). A 'DieselDisplay' device is selected under the 'aitfPriceDisplay_Role'. On the right, the 'EtherCAT Master' configuration is shown, including an 'EK1100_DieselDisplay_Device' and four terminals. A dialog box for 'Terminal1_DieselDisplay_Device' is open, displaying the 'EtherCAT E/A-Abbild' tab. This tab contains a table mapping variables to EtherCAT channels.

Variable	Mapping	Kanal	Adresse	Typ
Application.GasStation.INST_PriceBoard.INST_DieselDisplay.xDigit0SegmentA	↔	Output	%QX0.0	BIT
Application.GasStation.INST_PriceBoard.INST_DieselDisplay.xDigit0SegmentB	↔	Output	%QX0.1	BIT
Application.GasStation.INST_PriceBoard.INST_DieselDisplay.xDigit0SegmentC	↔	Output	%QX0.2	BIT
Application.GasStation.INST_PriceBoard.INST_DieselDisplay.xDigit0SegmentD	↔	Output	%QX0.3	BIT
Application.GasStation.INST_PriceBoard.INST_DieselDisplay.xDigit0SegmentE	↔	Output	%QX0.4	BIT
Application.GasStation.INST_PriceBoard.INST_DieselDisplay.xDigit0SegmentF	↔	Output	%QX0.5	BIT
Application.GasStation.INST_PriceBoard.INST_DieselDisplay.xDigit0SegmentG	↔	Output	%QX0.6	BIT
		Output	%QX0.7	BIT

How to use the CODESYS Application Composer

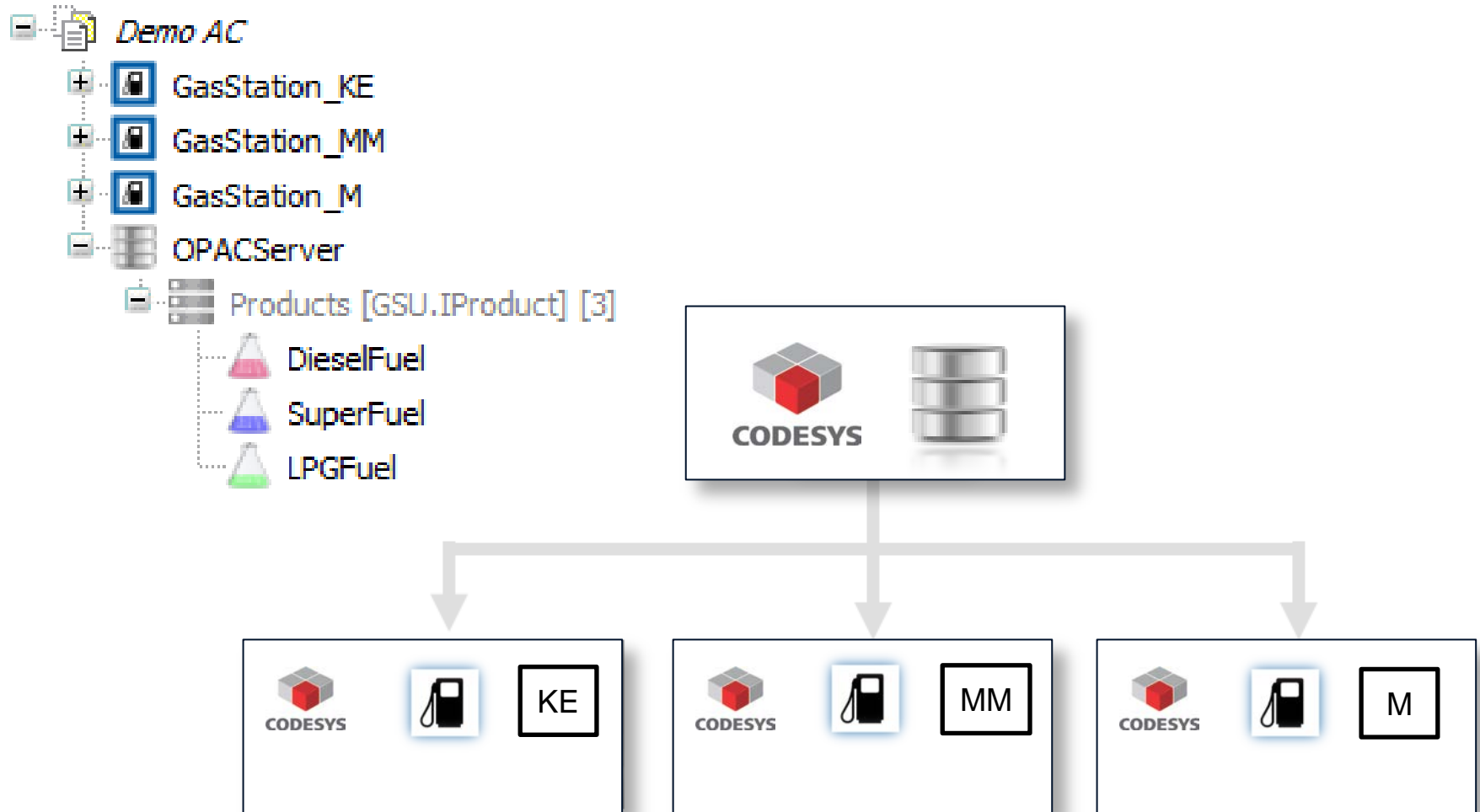
You receive a new order.

- The service station operator has meanwhile expanded his network.
- You delivered gas stations in different configurations.
- Price displays are to be controlled from the headquarters.



How to use the CODESYS Application Composer

Which changes have to be made in the project?





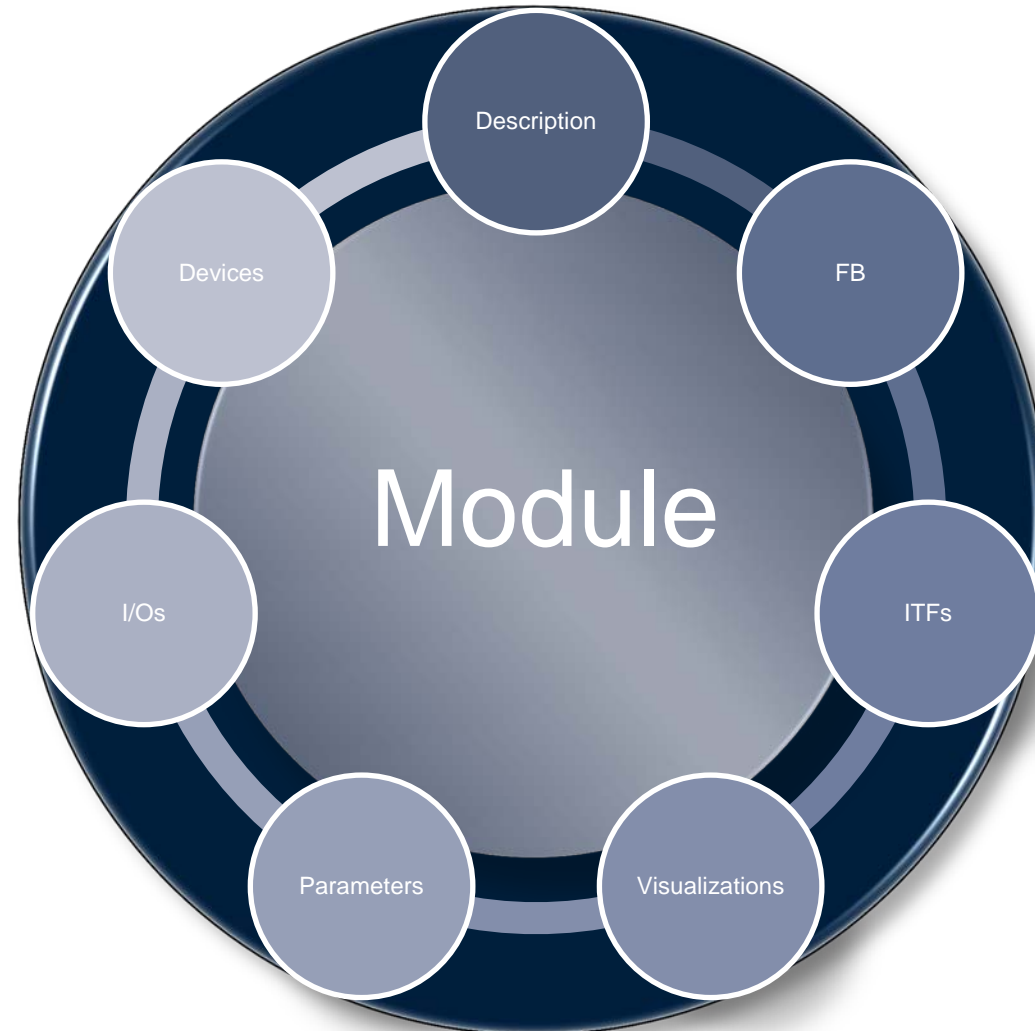
How to use the CODESYS Application Composer

Distribution of tasks to controllers

- With CODESYS, you can use controllers of different manufacturers in one project.
- Over 350 device manufacturers on the market use the CODESYS Runtime System.
- These devices can easily be connected to each other, and there is a communication interface even to external products.

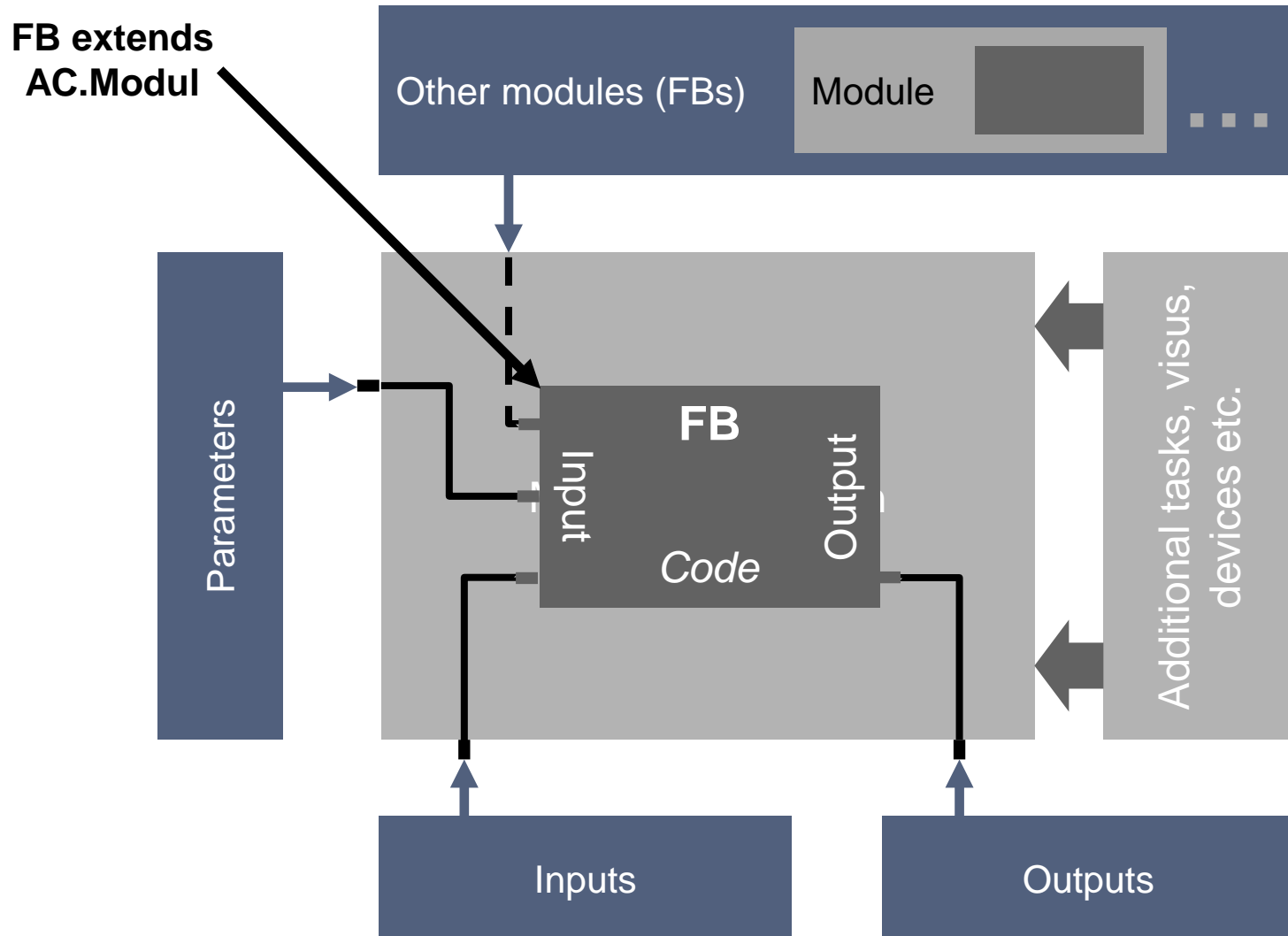
How to use the CODESYS Application Composer

How are modules created?



How to use the CODESYS Application Composer

Module implementation

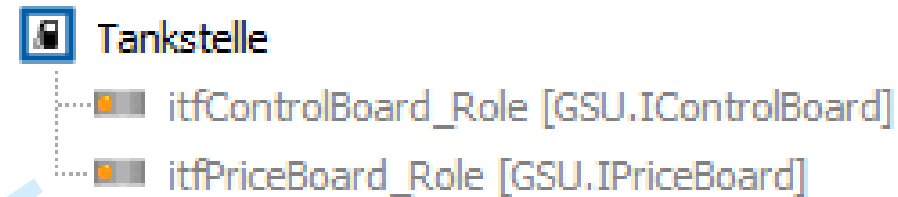


How to use the CODESYS Application Composer

Let's look at the module "GasStation".

```

1  MODULE GasStation IMPLEMENTED_BY GasStation
2  SEC std.MetaData
3     Desc      := TL.GasStation_Desc ;
4     Category  := 'Misc' ;
5     Icon_16   := ImagePool.GasStation_16 ;
6     Icon_32   := ImagePool.GasStation_32 ;
7  END_SEC
8  SEC Toplevel
9     SEC STANDARD_TASK : HIGH [3 lines]
10    END_SEC
11    GVL_NAME := 'GVL_%InstanceName%' ;
12  END_SEC
13  SEC Io
14     SEC Input : xSwitchSpotLight [3 lines]
15     END_SEC
16     SEC Input : xSwitchFrameLight [3 lines]
17     END_SEC
18     SEC Input : uiOperatioMode [3 lines]
19     END_SEC
20  END_SEC
21  SEC Slots
22     SEC Slot : itfControlBoard [5 lines]
23     END_SEC
24     SEC Slot : itfPriceBoard [5 lines]
25     END_SEC
26  END_SEC
27  SEC std.Visu [2 lines]
28  END_SEC
  
```



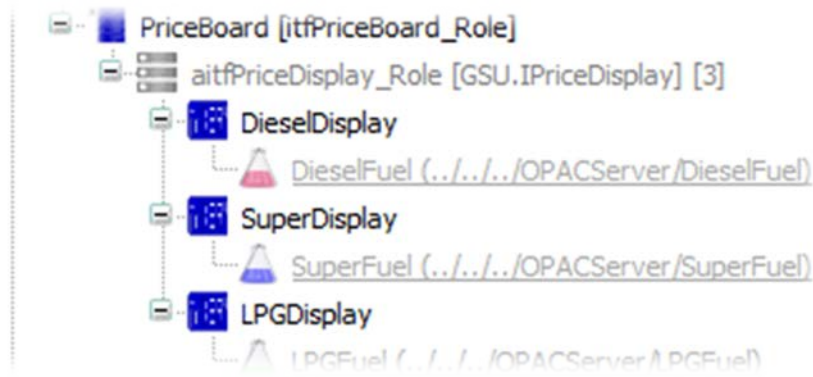
```

1  FUNCTION_BLOCK GasStation EXTENDS AC.Module
2  VAR_INPUT
3     xSwitchSpotLight : BOOL;
4     xSwitchFrameLight : BOOL;
5     uiOperatioMode : UINT;
6     itfControlBoard : IControlBoard;
7     itfPriceBoard : IPriceBoard;
8  END_VAR
9  VAR_OUTPUT
10     xFrameLight : BOOL;
11     xSpotlight : BOOL;
12  END_VAR
  
```

How to use the CODESYS Application Composer

Now let's take a look at the "PriceBoard".

- In the configuration phase, you define the number of displays.



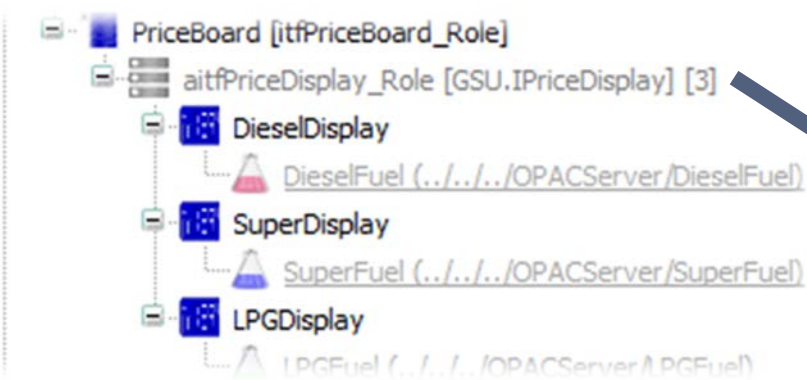
- In the previous presentation, the number of displays was programmed in CFC.

- A complex management mechanism had to be implemented.

=> The CODESYS Application Composer already covers these management mechanisms.

How to use the CODESYS Application Composer

The necessary code is generated automatically.



```

MODULE PriceBoard IMPLEMENTED_BY PriceBoard
SEC std.MetaData
    Desc      := TL.PriceBoard_Desc ;
    Category  := 'Misc' ;
    Icon_16   := ImagePool.PriceBoard_16 ;
    Icon_32   := ImagePool.PriceBoard_32 ;
END_SEC
SEC Slots
SEC Slot_Multi : aitfPriceDisplay
Variable := aitfPriceDisplay;
Var_Count := uiNumberOfPriceDisplay;
Role := TL.aitfPriceDisplay_Role;
Cardinality := [0 .. 5];
Type := SUBMODULE;
Name := TL.aitfPriceDisplay_Name;
Desc := TL.aitfPriceDisplay_Desc;
END_SEC
END_SEC
    
```

```

FUNCTION_BLOCK PriceBoard EXTENDS AC.Module IMPLEMENTS IPriceBoard, IOperation
VAR_INPUT
    itfOPACServer : IOPACServer;
    aitfPriceDisplay POINTER TO IPriceDisplay;
    uiNumberOfPriceDisplay : UINT := 0;
END_VAR
    
```

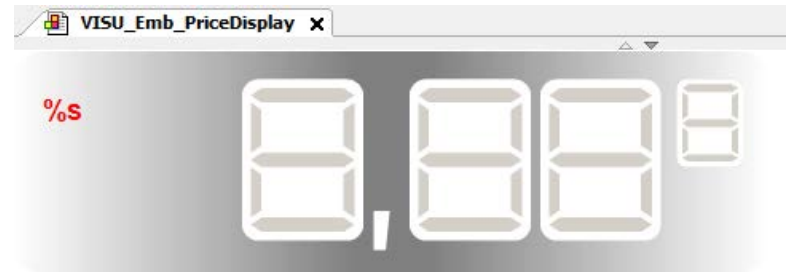
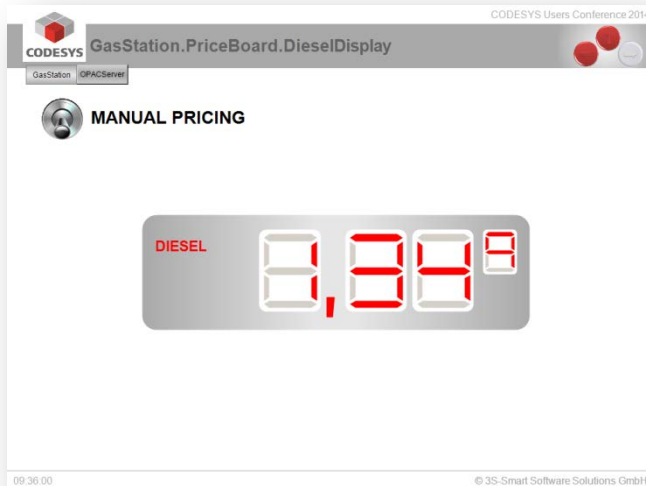
How to use the CODESYS Application Composer

We distinguish between two forms of visualization:

```

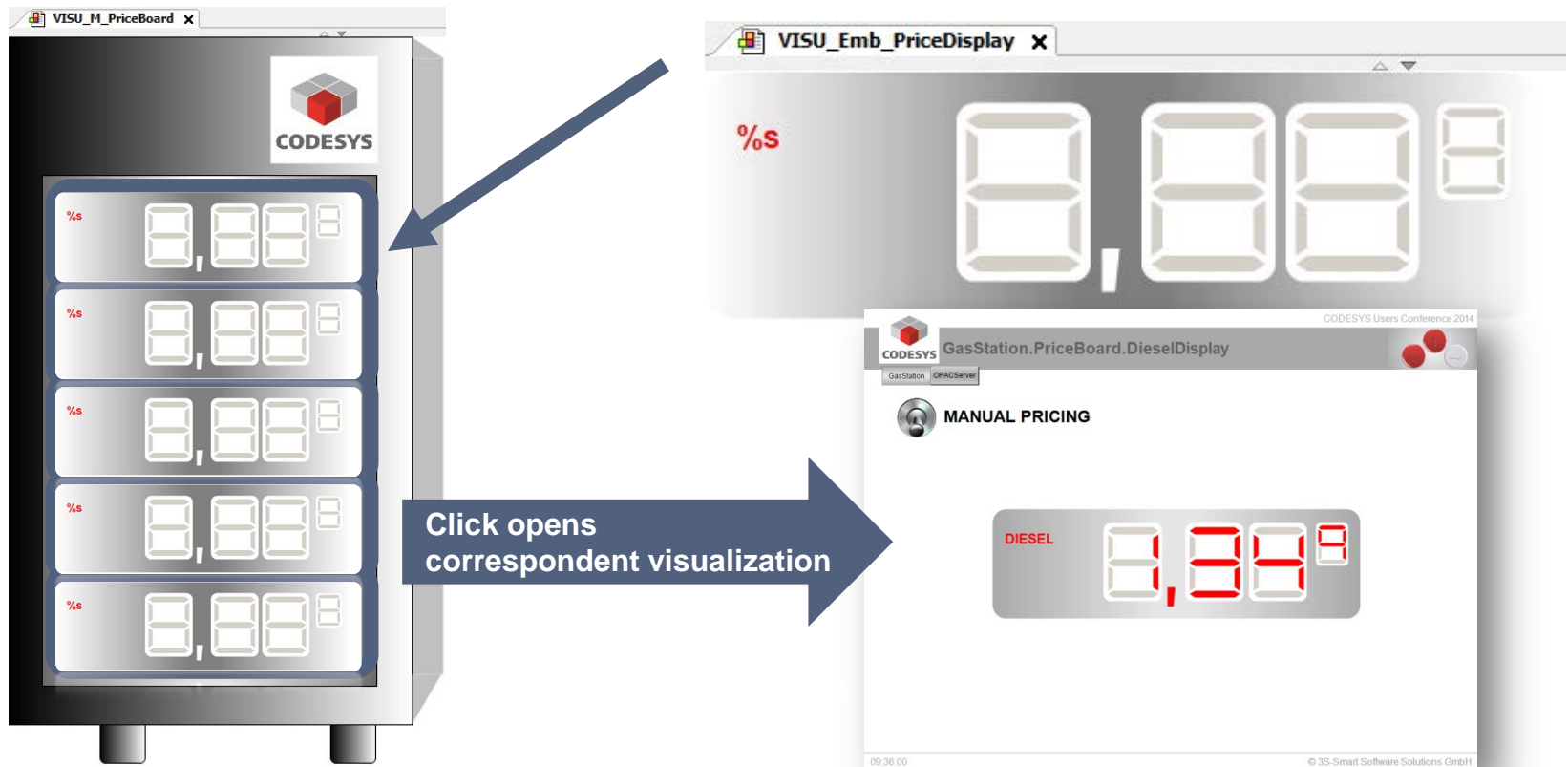
SEC std.Visu
  Page := [ VISU_M_PriceDisplaySetValue ] ;
  Embedded := [ VISU_Emb_PriceDisplay ] ;
END_SEC

```



How to use the CODESYS Application Composer

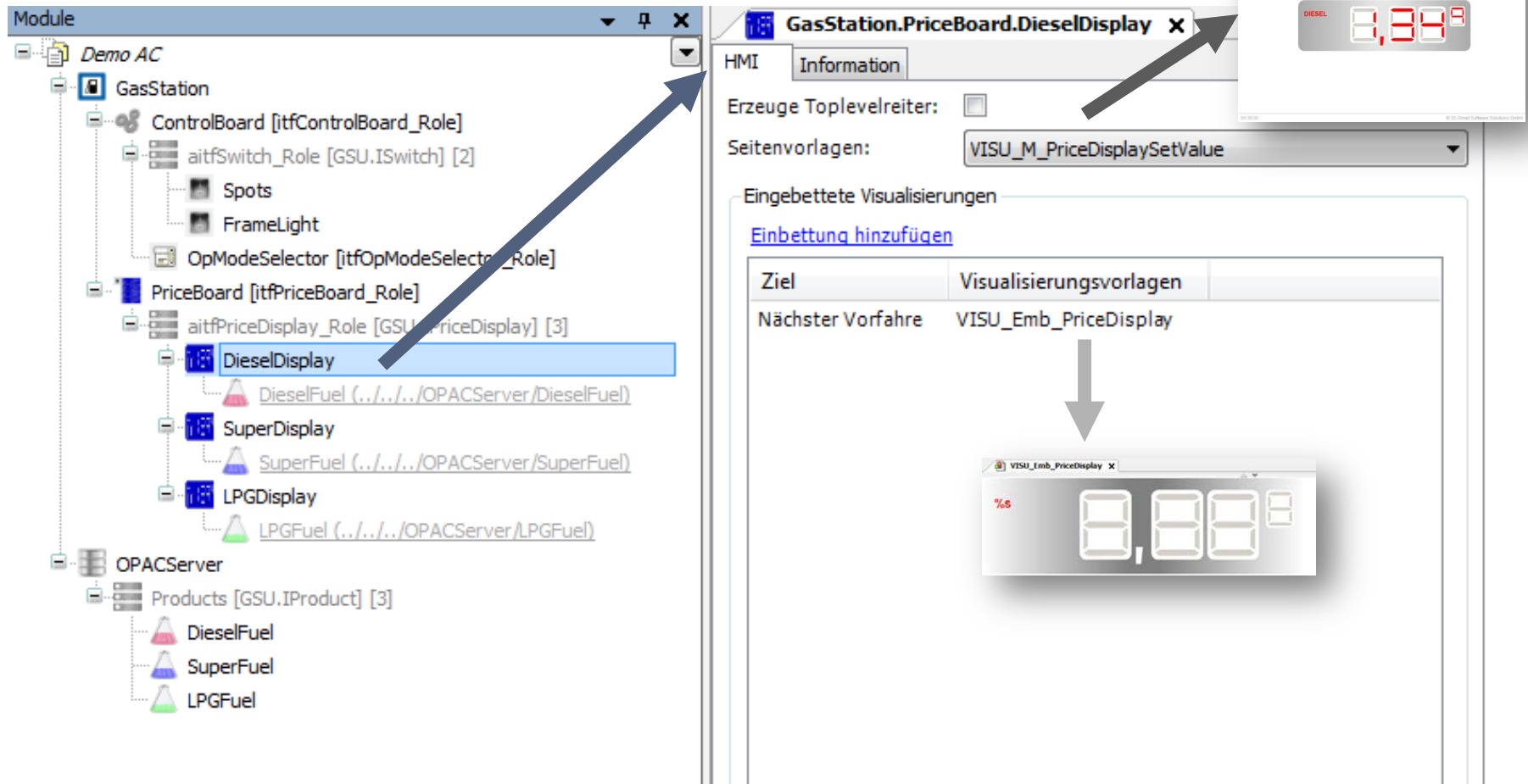
Combination of visualizations



Click opens
correspondent visualization

How to use the CODESYS Application Composer

The visualization configuration



The screenshot illustrates the configuration of a visualization in the CODESYS Application Composer. On the left, the 'Module' tree shows a project named 'Demo AC' containing a 'GasStation' module. Under 'GasStation', there is a 'PriceBoard' module which contains a 'DieselDisplay' visualization. A blue arrow points from the 'DieselDisplay' element in the tree to the configuration panel on the right.

The configuration panel for 'GasStation.PriceBoard.DieselDisplay' shows the following settings:

- HMI:** Information
- Erzeuge Toplevelreiter:**
- Seitenvorlagen:** VISU_M_PriceDisplaySetValue
- Eingebettete Visualisierungen:**
 - [Einbettung hinzufügen](#)
 - Table with 2 columns: Ziel, Visualisierungsvorlagen
 - Row 1: Nächster Vorfahre, VISU_Emb_PriceDisplay

Below the table, a grey arrow points to a preview window titled 'VISU_Emb_PriceDisplay'. This window shows a digital display with the value '1,84' and a red 'DIESEL' label. A second arrow points from the 'Seitenvorlagen' dropdown to this preview window.

**1**

What is the CODESYS Application Composer?

2

How to use the CODESYS Application Composer

3

Demo



Inspiring Automation Solutions

Thank you for your attention.