# Explanation of the combined technologies of PLCopen and OPC Foundation

## *Introduction*

The cooperation between PLCopen and OPC Foundation not only resulted in specification activities, these results were also shown in live demonstrations.
This cooperation merges two technologies:

- The PLCopen technology is based on the world wide IEC 61131-3 programming standard.
- The OPC Foundation technology is based on their OPC Unified Architecture as specified by their members. Be aware that this specification is submitted to IEC for standardization.

Overall this combination eases the communication in production lines and plants and the engineering effort to create the right interfaces and information.

## *PLCopen and IEC 61131-3*

The well-known programming standard IEC 61131-3 specifies several items that are useful for other systems as information and data. The first items are shown in the so called *Software Model*. Items like *Configuration*, describing the overall control components, the *Resources* as processing facility that is able to execute IEC programs, the *Tasks* controlling the execution of a set of programs and/or function blocks, and *Programs* typically consisting of a network of *Functions* and *Function Blocks*, able to exchange data. *Functions* and *Function Blocks* are the basic building blocks, containing a datastructure and an algorithm.
The prefix *Ctrl* is there to avoid conflicts with the OPC UA terminology. For instance, the definition of the 'Program' is a little different in both environments.

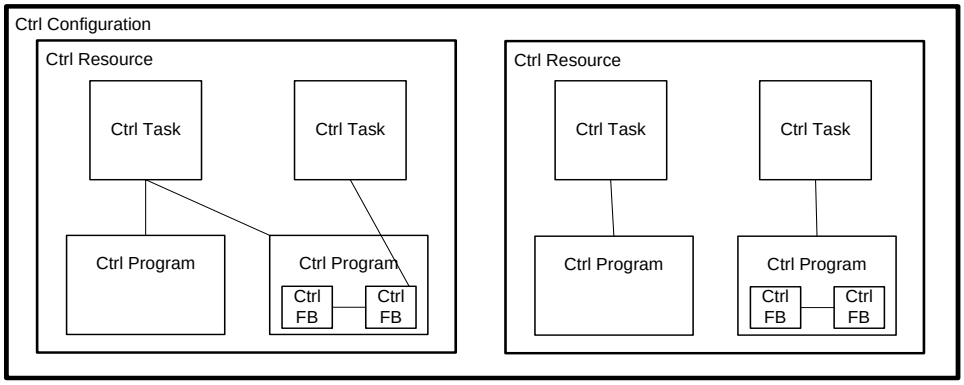


Figure 1: The IEC 61131-3 Software Model

The data exchange in this model is via variables, which are normally called by their name. Variables can have complex structures to better describe the content. Variables are also used in function and function blocks within the datastructure.

## *OPC Foundation and their Unified Architecture*

What the *Software Model* is for IEC, is the *Information Model* for OPC UA.
The main use case for OPC standards is the online data exchange between controllers or devices and HMI or SCADA systems, where the controller or device data is provided by an OPC server and is consumed by an OPC client, as is shown in the demonstration. OPC UA is platform independent

and the servers and clients can be directly integrated into devices and controllers. Features like security, access control and reliability are directly built into the transport mechanisms.
The OPC UA *Information Model* provides a standard way for *Servers* to expose *Objects* to *Clients*. *Objects* in OPC UA terms are composed of other *Objects*, *Variables* and *Methods*. OPC UA also allows relationships to other Objects to be expressed. *Objects* are used to represent components IEC 61131-3 software model like *Ctrl Program*, *Ctrl Task*, *Ctrl Resource* and *Ctrl Function Blocks*, and *Variables* are used to represent values.
The set of Objects and related information that an OPC UA Server makes available to Clients is referred to as its *AddressSpace*. OPC UA provides functionality to browse through a hierarchical namespaces containing data items and to read, write and to monitor these items for data changes.

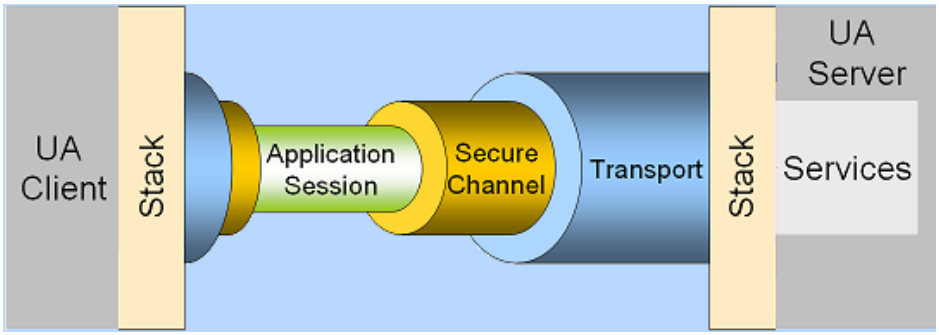


**Figure 2: The client / server architecture of OPC UA**

If we now represent the IEC software model in the OPC UA information model, it could look like the following picture (figure 3).

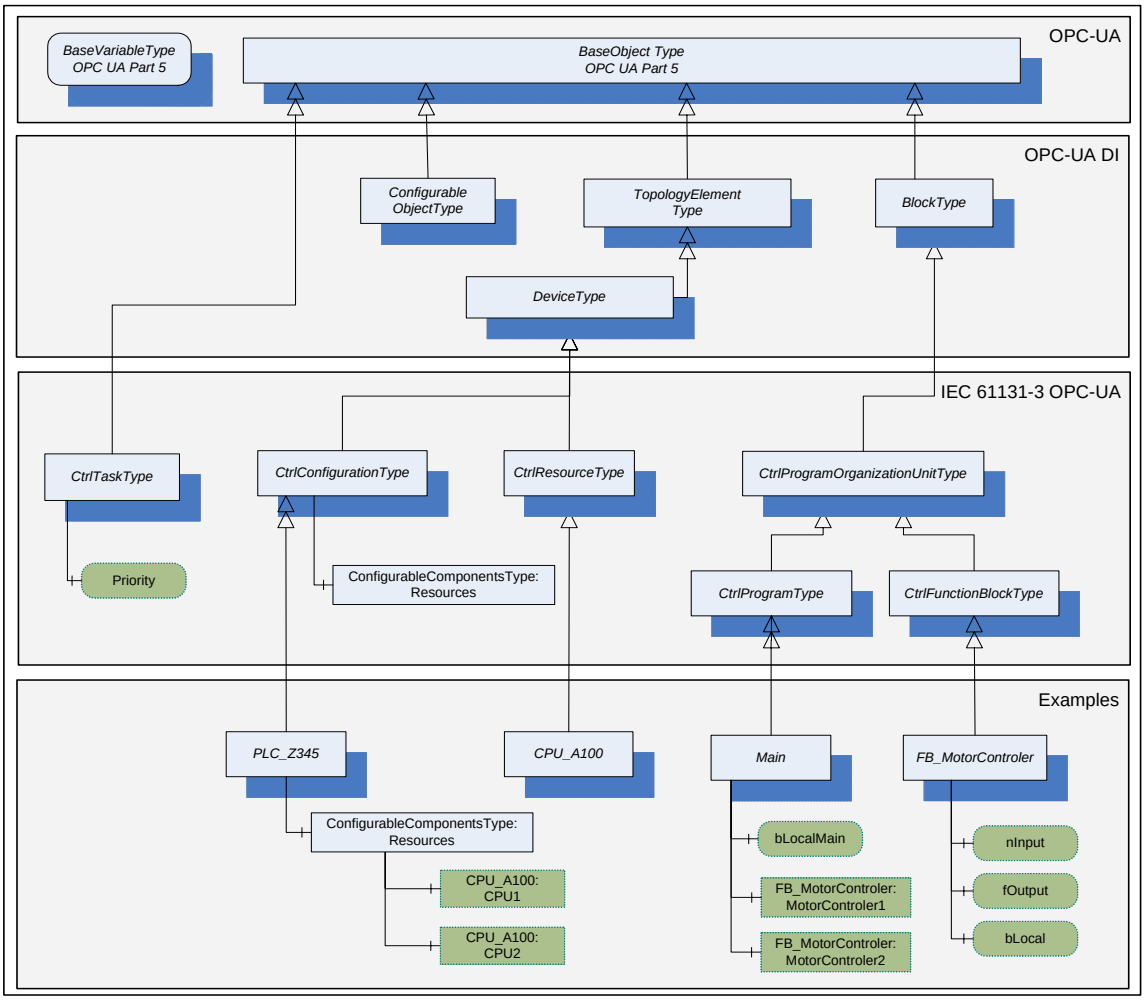


**Figure 3: The IEC model is using OPC UA Device Integration as base**

In this example we see four layers, with on top the 2 OPC UA related layers. In the 3rd layer, the link between IEC 61131-3 and OPC-UA is depicted. Here the elements of the IEC Software Model are shown, which are mapped on the lowest layer to the control architectures.

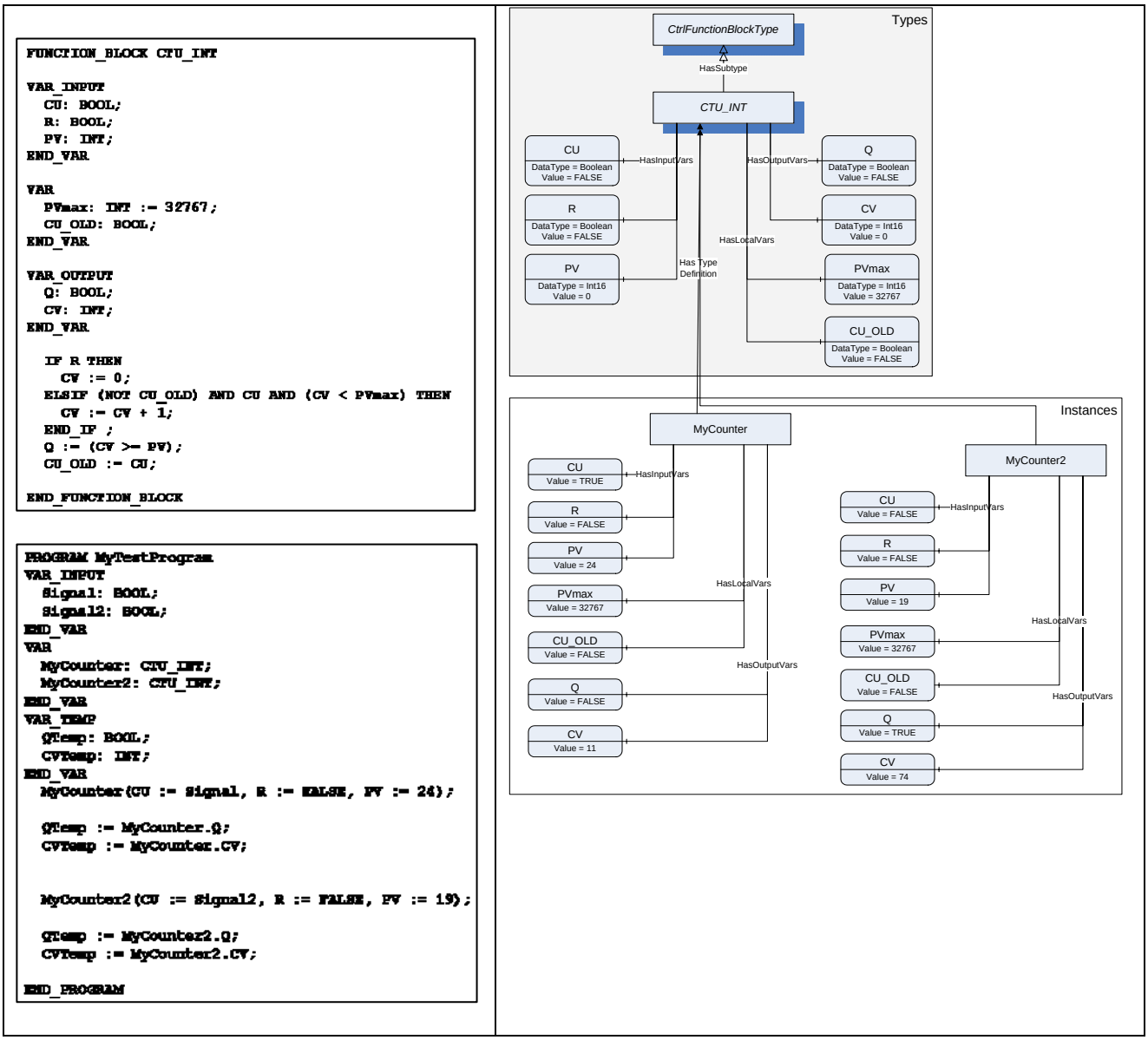


**Figure 4: A Function Block and Program …and the representation in the OPC UA information model**

In figure 4 we see the mapping of a self-defined function block (FB) to the OPC UA information model. On the left we see the definition of the FB with a counter functionality, starting with the name 'CTU_INT', and just below that the definition of the input variable, the internal variables, and the output variables, and below that the algorithm or code to define the functionality. Under this we see a small program using two instances of the same function block via MyCounter and MyCounter2.

On the right side we see the information model, with also on the top the '*Types*', in this case the *CtrlFunctionBlockType*, with the name of the FB, and below that on the left the inputs and on the right the outputs and internal variables. Just below that we see the two '*Instances*' of the function block.

With this information model, the OPC UA server on the controller can provide all the information shown here on the right side regarding the function block type and instances to the client. This makes a transparent communication possible. Since the information model can be discovered during

run-time, and the function block instance can be coupled to a graphical template prepared for the FB type on the client side. The reusability of PLC and visualization modules and the efficiency of the engineering process will be increased.

## *About OPC Foundation*

The OPC Foundation defines standards for online data exchange between automation systems. They address access to current data (OPC DA), alarms and events (OPC A&E) and historical data (OPC HDA). Those standards are successfully applied in industrial automation.
The new OPC Unified Architecture (OPC UA) unifies the existing standards and brings them to state-of-the-art technology using service-oriented architecture (SOA). Platform-independent technology allows the deployment of OPC UA beyond current OPC applications only running on Windows-based PC systems. OPC UA can also run on embedded systems as well as Linux / UNIX based enterprise systems. The provided information can be generically modelled and therefore arbitrary information models can be provided using OPC UA.
More information on www.OPCfoundation.org

## *About PLCopen*

PLCopen, as an organization active in industrial control, is creating a higher efficiency in your application software development: in one-off projects as well as in higher volume products. As such it is based on standard available tools to which extensions are and will be defined.
With results like Motion Control Library, Safety, XML specification, Reusability Level and Conformity Level, PLCopen made solid contributions to the community, extending the hardware independence from the software code, as well as reusability of the code and coupling to external software tools. One of the core activities of PLCopen is focused around IEC 61131-3, the only global standard for industrial control programming. It harmonizes the way people design and operate industrial controls by standardizing the programming interface. This allows people with different backgrounds and skills to create different elements of a program during different stages of the software lifecycle: specification, design, implementation, testing, installation and maintenance. Yet all pieces adhere to a common structure and work together harmoniously.
More information on www.PLCopen.org