UML in controller programming

Preface
Modern automation tasks are becoming ever more complex. The use of object-oriented programming (OOP) is presenting itself increasingly and is already supported by leading IEC 61131-3 tools. So that users can master OOP more simply, the integration of UML (Unified Modeling Language) in software systems according to IEC 61131-3 is the next logical step.

Introduction
UML has successfully established itself in many industrial sectors for the specification, design and documentation of software and, for example, mechatronic systems. IEC 61131-3 is currently the authoritative standard for application programming in industrial control and automation technology. The integration of both models under one roof is almost overdue.

Main part
OOP integrated seamlessly into IEC 61131-3
The growing demand for the integration of modern programming paradigms has led to the fact that, today, object-oriented programming in IEC 61131-3 programming systems is about to be standardized by the IEC committee. CODESYS V3 already fully implements the new programming paradigms, so that the user can create interfaces, methods and classes in all languages of the standard, enable the inheritance of functions or dynamically link blocks. The new required keywords thereby correspond to the standardization proposal.

Of course, OOP is not the means of the choice for everybody, for which reason users can also project functionally as before. In practice it has proven to be the case that system programmers use object-oriented programming for the basic blocks for the machine and plant automation and that users use these blocks via a functional call interface. Due to this combinability, all user groups can use IEC 61131-3 in exactly the same way as they are used to when developing applications.

UML for the description of technological procedures
Due to the use of OOP, the complexity of the relationships of the software blocks with one another can increase considerably: interfaces can be implemented several times, basic classes derived and extended or individual methods added or existing ones overwritten. The Unified Modeling Language (UML) is suitable as a uniform graphic modeling language for obtaining a quicker overview of these relationships or even purposefully creating them. At the same time UML offers the possibility to describe technological sequences graphically and thereby serves as an interface between technologists and the software developers for the application. Hence, the
technologists describe the process sequence of machine and plant and concentrate, for example, on production steps and chains, whereas the application developers for their part understand these procedures and code them in software.

Integration of UML in an IEC 61131-3 tool
The integration of UML in an IEC 61131-3 programming system was realized in the context of the research project "Increasing the efficiency and quality in the software engineering of automation technology for the domain of the machine and plant construction" at the Universities of Kassel and Munich respectively under the direction of Professor Dr. Birgit Vogel-Heuser. As industrial partners, 3S-Smart Software Solutions GmbH, Beckhoff Automation GmbH, ELAU AG, SIG Combibloc Systems GmbH and teamtechnik Maschinen und Anlagen GmbH initially sifted through the 14 available types of UML diagram. For the integration of UML in CODESYS, the approach is limited to the three most important diagrams: the class diagram, the state diagram and the activity diagram. The implementation thereby takes place on basis of the CODESYS Automation Platform, a kind of automation framework. This platform provides the UML plug-ins to the entire infrastructure of the tool: hence, the user can generate executable code from the UML source code using the native code generators and engines integrated in the tool and execute it directly on the controller.

Class diagram as alternative project view
The class diagram is ideal for the structured illustration of a project consisting of function blocks, interfaces, methods and their declaration sections. Beyond that the user can directly edit the class diagram and the relationships. The changes are automatically adopted into the linked program blocks. The class diagram thus actively supports the user in projecting within the IEC 61131-3 interface. The alternative illustration of the automation project in the class diagram alone simplifies communication between technologists and application programmers.

[Fig. 1: class diagram integrated in CODESYS V3]

State diagram for the programming of states and their transitions
As an independent implementation language, the state diagram enables the programming of states and their transitions. Similarities to the IEC 61131-3 language "Sequential Function Chart" SFC arise as a result of the elements "start state" (comparable to an init step), "end states", "states" (comparable with SFC steps), which consist of Entry, Do and Exit actions, as well as transition conditions. Decisions (corresponding to alternative branches in SFC) as well as branches (parallel branches in SFC) can also be inserted. In addition, the state diagram contains cycle-internal states, state groups, error transitions and orthogonal states. Like users of IEC
61131-3 are also used to in the Sequential Function Chart with CODESYS, the active states in online mode are highlighted in color on the controller. Comfortable debugging is thus enabled.

[Fig. 2: State diagram in CODESYS V3]

**Activity diagram for the programming of processes**
The activity diagram is also a self-contained implementation language and is perfectly suited to the programming of processes, because it enables a clear behavioral description due to a high abstraction level. With this diagram technologists in particular can already describe processes and sequences in their development sequence, which then have to be mapped in software by the developers of the application software. For its part, the activity diagram can of course access the IEC 61131-3 program blocks in order to fully cover the activity processing. In online mode the user is shown, in exactly the same way as in the state diagram, which activity the application is currently in or that, for example, a subsequent state has been switched to.

[Fig. 3: Activity diagram in CODESYS V3]

**Summary**
The extension of the IEC 61131-3 programming system CODESYS by the UML diagrams presented above offers the user a selection of available programming paradigms: on the one hand the classic, functional PLC programming in the IEC 61131-3 languages and on the other object-oriented programming with the same languages or with the UML aids. These break down communication barriers between technologists and IT specialists, thus enabling a faster implementation of the task while at the same time reducing errors of understanding. Beyond that the new UML diagrams facilitate access to OOP and represent the control sequences more understandably. On account of the complete integration in an IEC 61131-3 tool, genuine added value is thus created! And, since CODESYS in particular covers further classic automation tasks with integrated products, e.g. for process visualization, fieldbus connection or Motion Control, the user comes one step closer to fully integrated automation.

**Author:**
Roland Wagner
3S-Smart Software Solutions GmbH
Marketing Manager
Memminger Str. 151
87439 Kempten
Tel. 0831-54031-17
Fax 0831-54031-50
r.wagner@3s-software.com