# IEC 61131-3 + ACPLT = Dynamic Reconfigurable Models

Liyong Yu, Gustavo Quirós, Tina Krausser and Ulrich Epple
Chair of Process Control Engineering, RWTH Aachen University, 52064 Aachen, Germany
liyong.yu@plt.rwth-aachen.de

## 1. Introduction

IEC 61131 [3] is a well-established international standard for programmable logic controllers (PLCs). Programming environments applying the IEC 61131 follow the "traditional" development cycle: edit-compile-load-execute. In case execution code needs to be modified, interruption of the execution is necessary. In this paper, we present an approach that combines object oriented ACPLT technologies with the standard IEC 61131-3 languages Function Block Diagram (FBD) and Sequential Function Chart (SFC). In contrast to traditional systems, the entire object-oriented meta-model of FBDs and SFCs is available on the runtime system. A great advantage of this design is the possibility of online exploration of meta-information and dynamic reconfiguration of the system without interruption. This is especially useful for maintenance and re-engineering tasks for chemical and metallurgical plants, where interruption is often associated with high economic loss.

We first present a general description of the IEC 61131 and the ACPLT technologies. After this, our approach will be explained in more detail. A description of ongoing and future work concludes this paper.

## 2. IEC 61131-3

Part 3 of the IEC 61131 standard defines programming languages for the implementation of automation logic on PLCs. The standard defines the textual languages Structured Text (ST) and Instruction List (IL), as well as the graphical languages Ladder Diagram (LD), Function Block Diagram (FBD) and Sequential Function Chart (SFC). This paper will focus on FBD and SFC.

Function blocks are one kind of program organization unit (POU). They provide an interface in the form of input and output variables, and their state is kept in internal variables. An internal algorithm is executed periodically - commonly cyclically -, where the inputs are read, computations are performed, and internal and output variables are then updated. The internal algorithm is implemented in one of the standard languages. Function blocks can be connected with each other by signal connections and compose a FBD. FBD is also known as Continuous Function Chart (CFC).

SFC represents the sequential execution of steps and actions, therefore an activity flow. Conditional transitions between steps determine the conditions for the activation and deactivation of steps.

## 3. ACPLT Function Chart

ACPLT[1] technologies are reference models and software implementations that target application areas within the field of process automation. They are designed and implemented in a vendor- and platform-independent manner. ACPLT Function Chart (ACPLT/FC) has been developed with the goal of obtaining a general description language base for continuous functions and sequential processes in the field of automation. Due to the expressive graphical representation and the native support of existing automation hardware, CFC and SFC were chosen and further developed. By doing this, the position of CFC and SFC in classic basic automation can be strengthened. They can also be well-established in further application areas, e.g. engineering, batch-control, logistics, diagnostics and maintenance. The ACPLT/FC concept has been introduced in [6] in more details. Improvements of definitions in the standard have been suggested in [7].

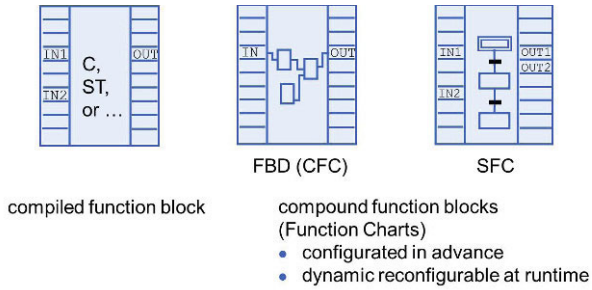


**Figure 1. Function blocks**

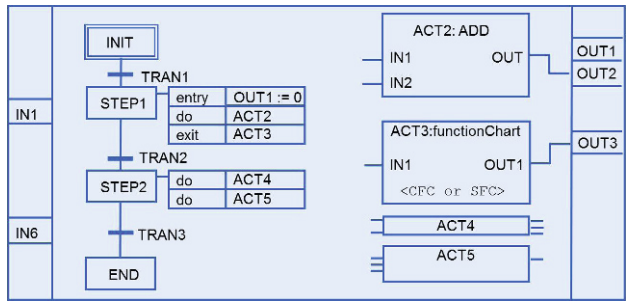


**Figure 2. SFC**

In the design of ACPLT/FC, the languages CFC and SFC were adapted from the "compilation model" assumed

[1] Aachener Prozessleittechnik: German expression for Aachener process control technique.

in the standard to “configuration model”. In ACPLT/FC, function blocks are defined as basic program organization units. As shown in Figure 1, two general kinds of function blocks have been conceived: compiled or native function blocks, which are defined by a specific object class, and configured or compound function blocks - also named as function charts in this work -, which are defined by a single class and contain a control logic which is defined by an underlying object structure. A CFC is a function chart nesting variable objects, internal function blocks and connections. The internal function blocks are executed following a fixed execution order which is given by an internal task list. An SFC is a function chart with variables, steps, transitions and actions. As shown in Figure 2, the execution of every step is divided in three phases: entry, do and exit. Actions can be implemented as function blocks.

## 4. Dynamic Reconfigurable Models

The ACPLT/FC has been prototypically implemented as class libraries in the object management system ACPLT/OV [1]. This approach presents several novel and distinctive features when compared to the common programming environments based on IEC 61131-3. These are shown in shown in Figure 3 and explained in the following.

**Complete meta-model available on the runtime system:** In traditional systems based on IEC 61131-3, the machine code running on the PLC lacks meta-information about the loaded instances, such as their type and the structure of their interconnection. In ACPLT/FC, the instance models and their meta-models are also present in the target system, meaning that this runtime system is self-descriptive. This is especially useful for maintenance tasks and reengineering tasks, and supports the development of adaptive systems and self-X technologies.

**Instantiate-parameterize instead of compile-load:** Instead of compiling diagrams and code into executable machine code to be loaded in a PLC, the user of ACPLT/FC performs the design of the control logic by instantiating and parameterizing objects - such as function blocks, connections, steps, transitions and actions - directly in the controller. This can be accomplished through an engineering client which communicates with the runtime system. In this manner, there is no distinction between design and implementation of the control logic, and the running system can be edited and redesigned at any point in time during its operation.

**Online model exploration:** The instance model and meta-model of ACPLT/FC may be explored and edited by local or remote clients. This allows great flexibility in performing engineering, operation and monitoring tasks. In our present implementation, clients supporting the communication protocol APLCT/KS[2] have been developed. ACPLT/KS has a similar philosophy to OPC UA [4].

**Dynamic reconfiguration via runtime model transformation:** A great advantage of having the instance model and the meta-model of the system available at runtime is the possibility of performing dynamic reconfiguration of the system in an automatic manner. The active objects of the system are able to explore and modify the system’s instance model in order to adapt the system to specific situations. This may occur internally in the system, and may also be carried out by an external client which connects to the system in order to perform the dynamic reconfiguration tasks.

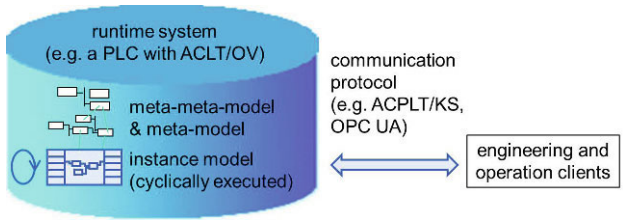


**Figure 3. Dynamic reconfigurable models**

## 5. Ongoing and Future Work

ACPLT Function Chart presented in this paper is being applied in the automation of a metallurgical melting furnace. In this project, all continuous functions and sequential processes are formally described and realized with the universal language base established by FBD and SFC. During a production of several days, the dynamic reconfigurability of instance models allows the user to load new batch recipes and modify plant start-up and shut-down processes. Functionalities of diagnosis, engineering and data archiving can also be extended online without production interruption.

A future topic we have addressed is the safety during modification, meaning that the running control program should never be led to critical situations. Additionally, an approach for Rule-based Engineering [5] is being developed for the implementation of dynamic system reconfiguration and system restructuring in a guided and safe manner.

## References

[1] D. Meyer. *Objektverwaltungskonzept für die operative Prozessleittechnik*. PhD thesis, RWTH Aachen University, 2001.

[2] H. Albrecht. *On Meta-Modeling for Communication in Operational Process Control Engineering*. PhD thesis, RWTH Aachen University, 2003.

[3] IEC. IEC 61131 Programmable Controllers (2nd Edition) Part 3: Programming Languages, 2003.

[4] IEC. IEC 62541 OPC Unified Architecture - Part 1: Overview and Concepts, 2010.

[5] T. Krausser, G. Quirós, and U. Epple. An IEC-61131-based Rule System for Integrated Automation Engineering: Concept and Case Study. In *INDIN 2011*, Lisbon, July 2011. IEEE.

[6] L. Yu, T. Krausser, G. Quirós, and U. Epple. SFC und FBD - Ein durchgängiges Beschreibungsmittel über die Grenzen der Basisautomatisierung hinaus. In *VDI-Berichte 2143, Automation 2011*, Düsseldorf, June 2011. VDI Verlag.

[7] L. Yu, G. Quirós, S. Grüner, and U. Epple. SFC-based Process Description for Complex Automation Functionalities. In *EKA 2012*, Magdeburg, May 2012. ifak.