This chapter goes beyond the specifications of IEC 61131-3 and outlines further requirements placed on \textit{programming systems} in the marketplace. These mainly stem from the special conditions to be met in the PLC environment; requirements and solutions are presented based on the IEC 61131-3 programming culture.

7.1 Requirements of Innovative Programming Tools

The performance of a PLC programming system can be judged by three criteria:

- Technological innovation,
- Fulfilment of PLC-specific requirements,
- Cost/benefit ratio.

The fact that IEC 61131-3 no longer makes a strict distinction between PLC, process computer and PC has a major effect on programming. With the aid of various backend compilers, one and the same programming system can be used to generate user code for

- \textbf{Compact PLC} (device, mostly in DIN rail format, with integrated I/O connections; proprietary operating system),
- \textbf{Modular PLC} (device, mostly 19” PCB, communicating via a multitude of I/O cartridges in the backplane; proprietary operating system),
- \textbf{Embedded Controller} (front-end computer with network connection (e.g. Profibus or CAN); proprietary operating system)
- \textbf{Embedded PC} (PC hardware-based; mostly Windows CE or (Realtime) Linux)
- \textbf{Standard PC} (extended by I/O boards, Windows realtime extension, uninterruptible power supply, ...)

All of the diverse combinations can be handled with a single programming system or programming method. As a rule, the programming systems run on Windows XP or Vista and are connected to the target PLC via serial connections or LAN.
This chapter discusses these features and outlines the most important components of a PLC programming system.

![Diagram of PLC programming system components](image)

**Figure 7.1.** Important components of modular PLC programming systems

Supplements to the above-mentioned subpackets (not part of IEC 61131-3) may include:

- Tools for plant project planning,
- Simulation tools,
- General data management systems,
- Parameterisation editors,
- Logic analyser,
- Plant diagnostics,
- BDE system, SCADA system, logging devices,
- CAD connection,
- Network administration.

### 7.2 Decompilation (Reverse Documentation)

*Reverse documentation* is a traditional requirement of the PLC market. Ideally, it should be possible to read out a PLC program directly from the controller, in order to modify it on-site at the machine, for example, far away from the office where it was developed. Maintenance personnel want to be able to read, print and modify the PLC program without having the original sources available on a PC.

*Decompilation* is the ability to retrieve all the information necessary to display and edit a POU directly from the PLC.