More choice in controls with Open systems

THE RELATIONSHIP BETWEEN USERS OF PLANT AND MACHINERY AND CONTROL SYSTEMS INTEGRATORS HAS CHANGED MARKEDLY SINCE THE UPTAKE OF IEC 61131. THIS STANDARD HAS GIVEN MACHINE BUILDERS AND USERS GREATER FREEDOM TO CHOOSE BEST OF BREED CONTROL EQUIPMENT, AS BOB DOBSON EXPLAINS.

In the early 1990s machine users felt increasingly trapped into always having to use the same control equipment vendor. The different makes of controller could not interface with one another, or to use the jargon they were not 'interoperable'. Further, each company's software was so uniquely its own that it was virtually unintelligible to anyone else.

So, once you committed to a particular vendor for one control system, you found you had to use them for every subsequent system and they seemed gradually to take over your entire factory. Quotes from other systems houses became virtually impossible to consider.

What was needed was a common platform of operation through the development of an International Standard. The International Electrotechnical Commission (IEC) took up the challenge, published draft proposals and began the long process of developing what became designated as IEC 61131.

While it focused on PLCs (programmable logic controllers) it took great care to embrace other technological solutions too, such as SCADA (supervisory control and data acquisition) and DCS (distributed control systems). It also established a standardisation body called PLCOpen (www.oacg.co.uk) to oversee conformance testing and administration.

Initially, the IEC reviewed PLCs in terms of their hardware design, programming and communications, then drew up performance specifications and design requirements. This allowed it to develop standard definitions and terminology so that all vendors all sang from the same hymn sheet, where previously buyers were left trying to compare very different quotes.

At this time PLCs were programmed in a plethora of different languages, Basic, Forth, C, C++, Structured English, Instruction List, Ladder and innumerable proprietary languages. Each one was good for certain things, but hopeless for others – no one language was good for everything, and most could not communicate with other languages. Plant operators thus found that they had to maintain fluency in a multitude of languages.

More freedom to choose: *IEC 61131 gives a structured and flexible approach to plant programming*

Programming language

So 1993 saw the publication of IEC 61131, Part 3, in effect a structured yet flexible approach to programming language. This embraced five different languages and defined how and where each should be used, so that their various strengths were utilised, their weaknesses avoided and none were used inappropriately. It explained how they could be integrated in a defined and structured way. The five languages are:

Ladder: A graphical depiction of the switches in a control system which allows easy visualisation of the system's operation. It is popular with shop floor engineers and technicians and also with older engineers who were brought up with relay-based control technologies. IEC Ladder also encompasses user-defined function blocks, so can be used in a hierarchical control system.

Complex algorithm or structured text: Like PASCAL this uses syntax so is ideal for decision based programming of the 'If-then' format (for example: If the temperature has exceeded 70degC, then activate the fan).

Instruction list or instruction set: This relatively low level language is used as a programme assembler, connecting different parts of the programme together.

Sequential instruction chart: The time and event-driven sequences within a programme are most easily dealt with using SIC. For example: If the temperature of the machine

exceeds 60deg C (event-driven), switch on the cooling fan then turn it off again after 600 seconds (time-driven).

Function block diagram: Another graphical language used for charting signal and data flows. The blocks are reusable mini-programmes, useful if you have, for example, a bank of 15 identical or near-identical mixers.

In subsequent years further Parts to IEC 61131 were developed to cover other aspects of control engineering and more will undoubtedly be added in future to refine the standard further. The current parts are:

Part 1: Definitions and terminology.

Part 2: Design and performance specification for hardware.

Part 3: Software languages and programme structures.

Part 4: Selection, installation and maintenance of PLCs.

Part 5: Inter-device communications protocols.

Part 6: Fieldbus interconnectivity.

Part 7: Fuzzy logic programming.

Part 8: Application and implementation guidelines.

So with a detailed Standard having been available for some years, how has it affected industry? Do users rely on IEC 61131? Have suppliers accepted it or do they try to avoid it? Is it still relevant today, or has industry moved on? What is the future for the standard – will it continue to evolve or has it already reached it final form?

Continue to contribute

In short, IEC 61131 has been a great success and looks like continuing to contribute to industry for many years to come. Most machine users are familiar and comfortable with the concept behind the standard and no longer feel so locked-in to just one supplier.

The larger end-users tend to specify it as mandatory; the smaller ones, being less formal, may or may not ask for it (and even if they don't ask for it, may get it anyway).

All the control equipment companies now conform to the Standard almost as a matter of course. However it should be noted that they will often suggest doing something beyond IEC 61131, saying that their suggested solution is better or more elegant, or cheaper than going by the book.

In many cases their suggestion will be true and for the good of the project, although of course cynics are free to take a different view.

"One of the great advantages created by IEC 61131 – and realised by control engineers – is the structured programming environment that has provided a protocol to follow. This reduces programming time considerably."

A typical example of this is in the use of micro-controllers or logic blocks. These have only become available in the last five or so years, so the Standard was drawn up before they were common.

They do not quite fit into the 61131 'landscape' and because they are of limited capability and used very much for local control it is very tempting for the systems integrator and client to agree to them running a non-standard mini-programme.

One of the great advantages created by IEC 61131 – and realised by control engineers – is the structured programming environment that has provided a protocol to follow. This reduces programming time considerably and, although they may not admit it, makes programming easier, yet ensures a more consistent quality to the end-product.

For instance a lot of the code is reusable, so can be copied – and quickly modified if necessary – from one part of a programme to another, or even into a completely different programme. Libraries of standard programmes have now been built up and are usable time and again in any number of jobs. Notably, fault finding has become very much easier, not only reducing commissioning time – and stress – considerably but also making it more predictable.

Easier to compare

This is manifest to the end-user in that it has become much easier to compare bids from different control companies as they now all tend to allow much the same amount for programming time.

With programming standardised it is now possible to have a rule of thumb, that programming should cost £10-20 per network node. (However, like all rules of thumb this is a guide only and you probably have to add and subtract sums for other issues, but at least it gives you a starting point.)

If the end user has in-house control engineers, they have had lifted from them the burden of understanding many different programme structures and proprietary approaches, now only having to be conversant with IEC 61131 Part 3.

Continuing to expand

The Standard is also continuing to expand, develop and evolve. For instance, Part 6 is keeping up with fieldbus developments as they emerge, while special function blocks for specific purposes such as motion control, inverter drives and energy management are becoming increasingly common currency.

It is notable that while older engineers still turn to Ladder as their first choice for programming the new generation of engineers much prefer the C or C++ with which they were brought up. Another generation shift is the now ubiquitous laptop computer for programming – the days of the separate programming tool are probably gone for ever.

It should be noted that complete programme portability has not been achieved, largely down to differences in the hardware platforms and small localised non-conforming mini-programmes.

However there is now a level of transportability, interoperability and interchangeablity that did not existing 10 years ago, and industry has come to realise that total portability is really rather academic.

While it may not be possible to mix and match different vendors' equipment at will, it is now far easier to interface different systems and to replace a supplier with a competitor when it come to a systems upgrade project.

Machine users now benefit from the fact that their controls suppliers are having to operate in a free market and so can get competitive quotes from several possible suppliers, compare like with like, and be fairly confident that the quote is not being packed out with excessive programming costs. They have reduced their perceived dependence on any given supplier and are happier for that. ■