

## Software Standardization Provides Key Benefits for Manufacturers, OEMs

by Vern Bolton, Program Manager, Industrial Solutions Center, Schneider Electric



Facility managers know efficiency is crucial when it comes to discrete manufacturing processes. After all, even in a sluggish economy, there are orders to satisfy, deadlines to meet and customers to please, so any edge a facility manager can gain to keep processes moving is appreciated.

Time was, if a machine went down, it required a member of the maintenance staff to drop what he was doing and connect a laptop computer to the machine to troubleshoot what went wrong. But there is a faster way to accomplish that same task today – through root-cause diagnostic logic built into the software that operates the machine. Essentially, software logic can be written to stop a machine if there is a problem and simultaneously send a message to a human-machine interface (HMI) or Supervisory Control and Data Acquisition (SCADA) screen, noting what's wrong and how to address it. From there, the issue can be attended to by a line operator if it's a simple fix, or a maintenance staff member if it's more comprehensive.

While this feature (typically called integrated guaranteed root-cause diagnostics) helps reduce or even eliminate potentially long periods of downtime – and the financial consequences that could result – it is most effective when incorporated into the software of all machines. The only way that can be accomplished is through development and implementation of a software standard, which can vary from a simple set of rules and guidelines for all programmers to follow, to the incorporation of function blocks, pre-tested logic for a specific task, training manuals and even automatic code generation methods. There are many benefits to software standardization:

- It allows a facility manager to reconfigure processes with less impact on time required to rewrite logic, meaning the software can easily evolve based on changing conditions as the years pass.
- OEMs are able to get machines to market faster because logic is pre-written and tested by the software supplier.



- If all OEMs are applying the same standard, the final destination for a given machine doesn't matter, because line operators need only understand the nuances of the software logic. Plus, machines can be moved from facility to facility as necessary with no learning curve for operators.

But accruing all the benefits of a software standard takes implementation, of course, which can't be endeavored without astute project management and communication. A successful approach requires buy-in from perhaps dozens of OEMs; a means of disseminating and teaching the standard, which can be provided by the software supplier; and perhaps most of all, the true leadership of the end-user manufacturer to ensure accountability. In short, it's a team effort.

## Standard Evolution

For most end-user manufacturers, software standards can be compared to a seed. The initial iteration of a standard usually covers basic functionality, like memory maps, logic section definition and operator interfaces. But as time goes on, and more projects and machines are added, the standard is augmented to make it more robust to meet changing conditions. In other words, changes to a standard attempt to add uncontrolled pieces of an application. For example, a standard initially may not have had logic to cover an interface to a data scanner, because it simply wasn't needed, so an update might add a function block to include that scanner's specific network address. Many times, a standard grows based on large projects – the software supplier will accumulate potential changes over time, and then implement some, or even all, of them when the next large project arrives. Of course, budget availability may predicate which changes are made and which ones are held for a later date.

Thus, a software standard rarely, if ever, stays static. The reason is simple – the potential for downtime,

and particularly a long period of unplanned downtime, is also never static, particularly when new projects or processes are added. There is simply too much at stake, financially and otherwise, for the end-user manufacturer to stand pat with a standard.

Hence, one of the primary goals of a software standard is to reduce potential downtime by helping line operators and maintenance staff resolve issues quickly. A carefully considered standard should point a line operator directly to the device within a process that is causing problems and also give some indication as to what to do to resolve it. The value can be measured in time – consider the following approaches when a process suddenly stalls:

### Scenario A:

Time a line operator spends trying to self-diagnose the problem	5 minutes
Time to call maintenance and have a staff member arrive	10 minutes
Time for the maintenance staff member to diagnose using a PC	15 minutes
Time to rectify the problem	5 minutes
<b>TOTAL:</b>	<b>35 minutes</b>

### Scenario B:

Time a line operator takes to view an HMI screen, understand what has happened, and what steps to take	1 minute
Time to rectify the problem	5 minutes
<b>TOTAL:</b>	<b>6 minutes</b>

Obviously, there is no indication of what the problem actually was in this example – it could have been anything from a simple obstruction to a motor suffering from an overcurrent condition. But in Scenario B, the software's logic was carefully crafted to ensure that a programmable logic controller (PLC) published an appropriate message to the correct HMI, allowing the line operator to know within seconds what the issue was, and how to rectify it.

Concerns about software degradation over time can be greatly diminished with standardization as well. A standard usually has a set of rules that must be followed for machine control and root-cause diagnostics to work. If the logic is written in such a way that when these rules are not applied, the end-user manufacturer can then be made aware of discrepancies, which can then be immediately addressed. Additionally, a control strategy can be designed that requires the diagnostic functionality to be in place to work, which eliminates the possibility of a maintenance staff member "jumpering" out a section of logic to avoid a component (e.g., a switch) to get a process running, and then never going back to fix the problem later, which opens the door to

unplanned downtime. At the same time, it reduces the possibility of root-cause diagnostic messages being ignored by line operators because they are simply no longer trusted.

For end-user manufacturers that frequently change processes, a modular approach to software standard development can be a good approach, because sections of the logic can be moved or reconfigured easily with no risk to the rest of the machine's logic. For example, if a software's logic was written in such a way that Task A is done before Task B, and a line operator realizes it should be the other way around, those pieces of logic can be quickly swapped. Some standards can be written so this task is automatically done by the software, though this takes a substantial amount of logic design up front.

## Project Management Crucial

Development of a software standard, based on the needs and budget of an end-user customer, is only one part of the overall equation. The others are testing of the completed standard and dissemination to OEMs, which both require careful coordination.

Typically, standard testing is completed by the software supplier for every conceivable use. If the software will eventually operate a hydraulic pump, for example, a supplier will either connect it to a pump or simulate a connection in order to make sure the logic works. This provides both validation to the software supplier and peace of mind for the end-user manufacturer. The alternative for manufacturers without software standardization is assuming the OEM's programmer wrote the software logic correctly – which could be an incorrect assumption.

Dissemination should be done by the software supplier, and typically includes training courses and documentation. The goal of this aspect of project management is to ensure OEM programmers can write the logic and thus comply with the standard. But the end-user manufacturer has a key role in dissemination of the standard, in that it must make its OEMs understand that the standard is in place for many reasons and there are multiple benefits for everyone, including the OEM. Too often during a training session, an OEM programmer will challenge the presenter from the software supplier about why the standard is necessary. It can create an uncomfortable situation, because the presenter is usually not empowered to “overrule” an OEM, but this can be avoided for the most part through pre-emptive messaging from the end-user manufacturer to its OEMs that application of the standard is non-negotiable. In short, manufacturer support of the software supplier's effort is a crucial success factor.

## Getting Started

It's important to note that a software standard may not make sense for all end-user manufacturers. Software standards require an up front investment for design, training, documentation and testing, and part of the payback is reduction of downtime by having better-written logic, quicker diagnosis of issues and a more seamless project launch. But keep in mind that there is also the monetary return on investment that should be considered from the outset; a manufacturer with a few machines in a single facility may not permit a reasonable ROI, if any.

But for those with hundreds of machines spread out over multiple facilities, perhaps globally, a software standard could mean the difference between meeting customer deadlines and expectations, or losing business to a competitor. The best rule of thumb is to make contact with a supplier with proven expertise in the development of software standards, and more importantly, project management and dissemination of those standards. Doing so can provide substantial benefits in both the short term and years, or even decades, into the future.

## About the Author

Vern Bolton is a program manager for the Schneider Electric North American Operating Division's Industrial Solutions Center, working out of the Troy, MI, office managing a team supporting many different power train projects for a major U.S. automotive manufacturer.



**Schneider Electric USA, Inc.**

1415 S. Roselle Road  
Palatine, IL 60067  
Tel: 847-397-2600  
Fax: 847-925-7500  
[www.schneider-electric.us](http://www.schneider-electric.us)

