CRACKING THE CODE ON STANDARDIZATION

Manufacturer makes the choice for standardization.

You've got a complex system that represents millions of dollars in capital investment. But it's running slowly, creating bottlenecks, and racking up downtime. The code used to control the entire system has been changed so many times by so many different people that nothing is standard. Even hardware problems are difficult to correct, because they're masked by code issues. Standardization could make everything run smoother, but it can be costly and include risks of its own. So what do you do?

Background

A global consumer products manufacturer in the Midwest was facing the question of standardization for a complex conveying system that carried finished product to palletizers. The equipment, worth tens of millions of dollars, most of it was 10-15 years old, was running 24/7 and had become increasingly difficult to maintain and support.

A complex system without a standard.

The system itself was anything but simple. It had over 20 outputs from packaging going to seven palletizers. This included six PLCs, around 400 motors, two dozen barcode-scan points, plus numerous merges, switches, diverts, and more along miles of conveyors. To complicate things even further, the plant's wide variety of product formats meant that small, light, poly-wrapped product was running on the same line as large heavy cased product.

Over the years, different integrators and plant personnel had programmed and reprogrammed the logic for the system. Now, nothing was standard. Productivity was suffering, and troubleshooting the system required excessive downtime.

CHALLENGE: Time for a full code rewrite?

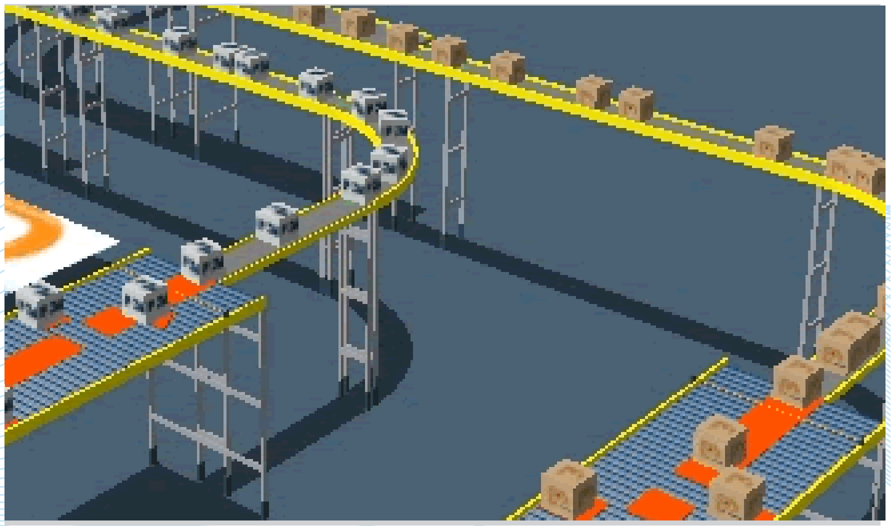
The only answer to these issues would be a full code rewrite. Effectively, the logic for the system would have to be totally rewritten, and all naming conventions would have to be redone. Further, a project of this scope often requires that even some of the hardware structure would need to be updated. An undertaking of this magnitude would raise a lot of questions for the manufacturer.

After systematically working through all of the tough issues with the Polytron engineers, the manufacturer decided standardization would enable five key benefits that would far outweigh the risks and costs of the project.

Key goals and benefits of standardization? Standardization would boost productivity through a variety of factors such as higher throughput, more reliability, faster speeds, and less downtime. The five key benefits identified by the Manufacturer and Polytron team would include:

1. Major reduction in downtime. The plant was experiencing excessive downtime because problems with the conveying systems were backing up and stopping production. Too often, viewed as "just conveyors" in warehousing operations, this area easily became a serious bottleneck, choking productivity for the entire system. Standardization would dramatically reduce downtime. Operations has a target of no more than 0.5% downtime.

2. Major reduction in partials. Productivity was suffering because the system was creating partials – splitting a full pallet-sized slug of product into smaller slugs – or when extra product had to be pulled manually. For this consumer goods manufacturer, partials associated with one section of conveyor were as high as 23% with a stated goal to reduce partials below 10%. Standardization would allow for an effective reduction in partials and increase in productivity.



3. Faster, more accurate troubleshooting. Without a common programming structure, troubleshooting had become difficult and time-consuming. Standardization would eliminate excessive downtime needed to find and fix problems in the system using the new code to identify the real cause of the productivity problem.

4. Easier optimization and scalability. The old code had been modified so many times that it varied significantly from its originally intended purpose. It had become difficult for revamped packaging lines and warehouse conveyors to support the new systems. Standardization would not only optimize their current system, but would support future growth more easily.

5. Better identification of hardware issues and limitations. Standardized code makes it easier to spot equipment issues, correct them, and bring the hardware back to centerline. The new logic was written expecting the equipment to behave and perform as designed. . For this manufacturer, the team completed a full system audit while conducting the code rewrite.

What's the most efficient, cost-effective process for such a project?

Obviously, a full code rewrite is a major undertaking, but if handled correctly, the disruption to production is surprisingly minimal and the ROI can be enormous. To duplicate pre-existing logic in a standardized format, our Polytron team followed a seven-step process that led to a robust new solution with nearly vertical start-up.

1. Identify the standard. First, the standard that the client wanted to use going forward was developed. The team customized Polytron's code standard to meet their specific needs and generated a document outlining specifics of the programming structure, nomenclature, tags and standard routines. A functional specification was developed as a way to communicate the standard in a written form that would be easy for anyone to use. The team then created the sample code and routines that would be used as the new standard. A library of generic routines was built that could be configured and applied to every equipment type whether it be belt, roller or accumulation conveyor, a divert or switch, or merge section, and so on.

2. Develop a logic narrative for the system. The existing code was reverse-engineered and listed out as a logic narrative to tell how each and every piece of equipment and conveyor would work together. The team determined how the PLC would interact with other equipment, naming conventions, and other factors. Once everything was clearly laid out and approved by the client, work on the new code was started.

3. Develop the code. Using the agreed-upon standard and the sample routines, the Polytron team developed the code in an entirely new program file. The code itself would essentially be object-based; ensuring strict adherence to the standard. Developing the code takes place in a two step process. First, repetitive logic is developed by using our code generator tool – in essence guaranteeing that there are no mistakes made during data input and editing. Finally, our engineers focus their main efforts and time on writing the intricate, custom portions of the code as detailed in the logic narrative.

4. Create the emulation model. The manufacturer's real factory situations were then emulated by creating a model that is controlled entirely by the new PLC and HMI code. This allowed the team to fully test the new controls system. An emulation is a complete model of the system controlled by the actual PLC code in real time – just as the real system would be in the factory itself. This enables our teams to test more variables and receive more data faster than running the system in the field. And, of course, a major benefit is that all this is done without affecting real production. By reducing the efforts needed on-site and achieving a more vertical startup, Polytron nearly eliminated program-related risks during startup and created major savings for the customer.

5. Conduct in-house Factory Acceptance Test (FAT) with the customer. At this stage, Polytron brought the customer to our offices and began to demo everything. The system was pushed to its limits - essentially trying to "break the system", so that the team and the customer would be absolutely confident in the ability of the new code to properly manage their product. Our emulation FAT process proved to be so valuable for this manufacturer in past projects, that senior management for the company has said, "we will never do another project without it."

6. Field installation and testing. Now, the logic was ready to download and be put through a rigorous validation process. The same scenarios were validated in the field that had been validated using the emulation model. The code was solid, and the team recognized a significant improvement in system performance both in reduction of partials and downtime the first week. The majority of problems that showed up in the field were equipment and hardware related; issues that had, until then, been masked by the old code. Because the team used Polytron's emulation model to conduct the FAT, a nearly vertical start-up was achieved – with no need for weeks of late nights and weekends trying to get the system up and running properly. In fact, the manufacturer estimates that the emulation-based FAT cut the onsite validation time by 50%.

7. Formal training and documentation. Since success depends on the ability of plant personnel to support the system, Polytron conducted detailed, code-specific training. Originally, this training had been scheduled to last four hours. However, the training was so helpful that the company extended it to include two full days.

Conclusion: Spreading the success

The newly standardized system is delivering higher throughput, greater reliability, and faster speeds. The code rewrite exceeded the manufacturer's objectives! Partials are now less than 15% - and falling, and downtime has been cut in half.

With these improvements in productivity, the operation has recognized significant improvement in servicing manufacturing and packaging - their upstream customer - and a notable decrease in disruption to their warehouse workforce.

With the success of the first phase of the standardization project, the manufacturer raised the priority in deploying this solution throughout the remainder of their systems. Currently engaged in improvement processes throughout the plant, Polytron has now completed the third of four phases of rewriting code and optimizing their remaining systems.

The manufacturer estimates that the emulation-based FAT cut the onsite validation time by 50%.

Meanwhile, if a problem occurs on a standardized system, it's much easier for the manufacturer to resolve. Now that the code rewrite is complete, programmers and technicians are able to use the code to identify the real cause of productivity problems (factors such as hardware issues or product handling issues).

Want to learn more about standardizing your systems?
Standardization works. And it can work for your system. To find out how, give us a call at 1-800-622-4326.

855.794.7659	Toll Free	3300 Breckinridge Boulevard
678.328.2999	Local	Suite 100
678.328.2880	Fax	Duluth, GA 30096-8983
rsklamm@polytron.com		