Implementing ISA-TR88.00.02 (PackML)

How OEMs Turn Promise into Benefits

Four OEMs Share Their Experiences Implementing PackML in Machine Design and Build. Mettler Toledo, and Axon, ID Technology, and Matrix built by Pro Mach share their stories.

Introduction

Many packaging machinery manufacturers are finding that implementing PackML/ISA-TR88.00.02 across a machine product line isn't as difficult as first thought and that the rewards exceed expectations.

By implementing PackML/ISA-TR88.00.02 OEMs are experiencing:

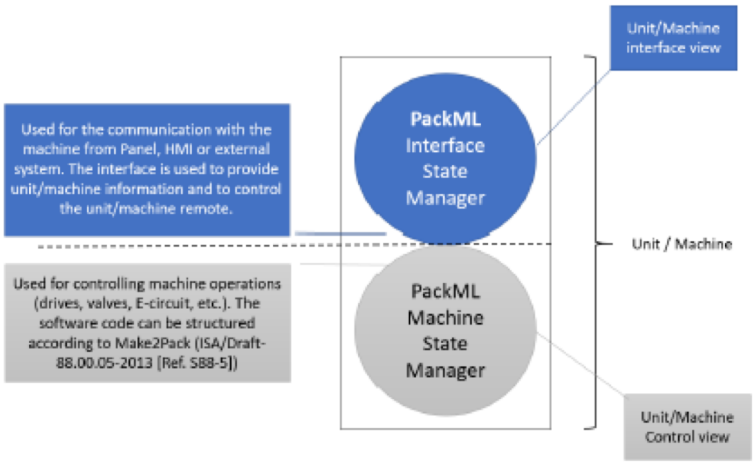
- Faster specification, design, build and engineering processes
- Easier product testing and shorter commissioning times
- Quicker product turnaround and payment cycles
- Lower warranty costs
- Easier customer training, aftermarket problem resolution and service
- Faster new product development times
- Higher quality final products

Getting Started

OEMs wanting to use the PackML state model for their internal Machine State Manager have two scenarios shown in Figure 1:

1. PackML gateway unit/machine
2. Full PackML compliant unit/machine

Figure 1: PackML Interface vs State Manager

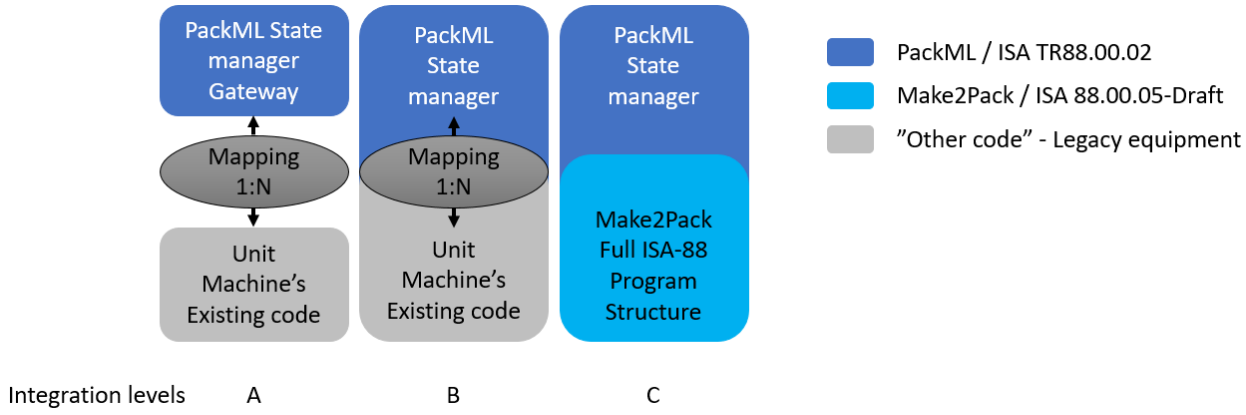


The PackML Interface State Manager is used for communication with the machine from Panels, HMIs or external systems. The Interface is used to get status, start, stop, pause, or change parameters of the machine. The PackML Interface State Manager provides a single communication interface between the machine and the HMI or other external control system (for example other integrated machines or supervisory control system) and any PackML enabled machine.

The PackML Machine State Manager is a supplier defined state model to control machine operations. The code can be structured according to Make2Pack (ISA/Draft 88.00.05-2013).

The PackML Interface State Manager can either be implemented in the same control system (CPU) as the Machine State Manager or in a separate control system. A full PackML machine will have one common Interface and one Machine State Manager that is totally integrated.

Figure 2: PackML Implementation Types



Mettler-Toledo

Tom Dorward, Product Manager, Mettler Toledo

In 2006-2007, Mettler Toledo was developing a new control platform with a requirement that machine conditions must be communicated from the checkweigher to the customer's control system. Mettler Toledo started looking at PackML, which had been advanced by several major customers. Mettler's team started working directly with them to understand PackML's value. PackML is now the standard for all Mettler Toledo Product Inspection machinery, and a fundamental requirement for all higher-level communications products.

Once the decision was made to go with PackML, it was a very small portion of overall control development. The machine communications requirement was already in the spec, and PackML was a defined solution. Implementing PackML saved Mettler Toledo time and money.

The reduction in engineering cost and time that Mettler Toledo experiences is mainly through eliminating custom code to meet a customer specification or the need to document necessary commands to implement a request. With standard code and tag definition, customers either know, or its easy to illustrate what they are getting from the outset. Since Mettler Toledo's implementation of PackML is always the same, documentation remains consistent. PackML standards provide a base for documentation of PackTag definitions, commands, and data.

Following a standard has allowed Mettler Toledo to create a standard product offering, and reduce creation of custom solutions. Engineering design and test time are no longer required once it had the software written. Mettler Toledo's focus is now on improvement rather than repetitive development.

Mettler Toledo's support team can focus on connectivity solutions using a common knowledge base for data and commands, making it easier for them to quickly identify issues and resolve problems.

Extending PackML across product lines benefits OEMs and End Users

Mettler has extended PackML across multiple product lines—Checkweighing, Product Inspection machines (Metal Detection, Vision, and X-ray). PackML use has broadened from Fieldbus, to OPC, and the product inspection data collection system, ProdX. Following a standard that is applied across multiple products and manufacturers provides customers with a consistent platform to integrate their lines.

Axon

Christopher Thomas, Director of Controls Technology
Ken Nyren, Engineering Manager, Axon

By implementing PackML, Axon cut debug testing time by 60 percent. Overall engineering and manufacturing time was reduced 40 percent, including building and debugging. Total allotted testing time for Axon machines is a fixed period, but the time from machine start up and testing it also went down 40 percent. Now, 80 percent of their code is reusable and standardized across all of their machines.

Axon found that once they absorbed the initial upfront cost of implementing PackML, the payoff was well worth it.

Five years into using PackML, Axon can turnaround machines faster. Modules are already tested, which comprises 80 percent of the electrical system.

In service and support, electrical engineers now rarely have to travel to customer sites. Service technicians can frequently make a few minor adjustments or debugs remotely with a laptop. The modularity and simple standardization have resulted in higher quality machines and lower warranty costs.

Now that Axon has a mature PackML code base to use, it can implement a new machine in about a week. This time includes a fully developed PLC program and HMI. The added benefit is few, if any bugs, because most of the reused code has already been tested and debugged. Previously, a new machine built from scratch would take maybe 4 to 5 weeks to fully develop and test.

Since Axon machines are so modular now, they can easily turn extra features on and off. This allows them to prebuild a machine with a base set of features. Then, when a machine is sold, Axon can introduce additional features as modules and enable them in the software. This has had an extraordinarily positive effect on lead times. Axon has seen lead times reduce to as short as 4 weeks.

Axon is experiencing easier post-sales support because programs are the same and the quality of the code is better. PackML has helped Axon use a common HMI design language. This means greater consistency among models. All machines look the same and setup the same. Service technicians don't have to worry about every machine being different, which significantly reduces training and support calls.

Internally, the Axon engineering team likes using PackML. It enforces programming standards. Different programmers work with the same set of rules instead of being allowed to add their own ideas into how the basics of a machine work.

Axon develops all their machines to be PackML compliant, even if it is not requested by customers. For end user customers who know the benefits of PackML and want it, Axon feels their experience with PackML gives them a competitive advantage. Customers who don't specify PackML typically realize the benefits it provides once they have more experience with it.

Axon Advice to OEMs

Limit your focus on what your needs are for your machine. Start with the OMAC website. They have much if not all the resources needed to begin.

A good starting point is to begin with something common across multiple machines and understand how the code works and how you can transfer it from machine to machine. This opens the door to the possibilities and gets your team and others in your organization excited about pushing forward. A real benefit of PackML is the ability to reuse modules. PackML also helps engineers identify commonality between machines and increase code reuse from across machines. Over time the code base will grow and by the second machine you don't have to do it all over again.

Matrix

Marc Wilden, Vice President, Matrix

Matrix adopted PackML when they were developing their first continuous motion, vertical form fill seal machine, the Morpheus line. Matrix found PackML architecture a convenient way to segregate different areas that needed to be developed. PackML's inherent modularity made it easier to work with and divide tasks among external and internal partners in product development. Matrix felt the learning curve involved in adopting PackML was trivial compared to these real benefits.

Matrix has been able to reuse most of the code they wrote using PackML. Matrix found that the 10 percent more effort it took to introduce PackML to internal and external partners allowed them to reuse 90 percent of the code.

PackML also helped Matrix gain a huge degree of cross compatibility among products. Matrix took 90 percent of the code they wrote for the continuous motion Morpheus machine and reused it on a new intermittent motion machine, saving them design, engineering, and build time.

Matrix found that PackML reduces engineering time, is scalable, and efficient. This has resulted in shorter commissioning time for machines and significant savings in costs and time. Instead of building machines from scratch, Matrix can add options and turn a machine around in 3-4 weeks compared to 10-12 weeks previously. Lead times have been reduced 25-30 percent.

Before PackML when Matrix turned off a feature, it was often considered dead code. Because of the standardized, modular nature of PackML, code can be reused elsewhere. It's simple to add in and reuse code in different situations.

Customers like PackML Enabled Machines

During Factory Acceptance testing for Matrix' first PackML machine, the customer — which wondered why PackML was used — discovered quickly how easy it is to identify issues, check a module and resolve it. Now that customer requires PackML on all the machines they buy.

ID Technology

Mark Will, Automated Systems Engineer, ID Technology

ID Technology found PackML to be an ideal solution that could be scaled down to fit in the control systems of their Labeling and Marking equipment.

PackML's modularity allowed ID Technology to isolate the essential states required for its systems. The ability to use just the required states in PackML that were necessary for ID Technology were a real benefit in quickly getting up to speed on implementing PackML. That benefit has carried over into faster coding and reduced development time.

ID Technology now reuses code across platforms, resulting in reduced engineering hours and consistency across product lines. ID Technology plans to use PackML in new machines going forward.

Conclusion

Mettler Toledo, Axon, Matrix and ID Technology design and build vastly different types of packaging machinery and took different approaches to implementing PackML. The results from implementing a modular, consistent standard across their product lines were similar. All four OEMs realized cost and time savings, as well as product consistency that benefitted them as well as their customers.