



AutomationWorld®

TACTICAL
BRIEF

THE POWER OF PACKML

CONTENTS

- 02 Designing your first PackML implementation for machine control
- 17 P&G puts PackML through its Pace
- 23 Evolving strategies for packaging machinery specifications
- 26 Nestlé makes packaging a strategic priority

Document Number: PR.AW.01.

SPONSORED BY



DESIGNING YOUR FIRST PACKML IMPLEMENTATION FOR MACHINE CONTROL



 | SPONSORED CONTENT

Three key design decisions to get you started

By [Douglas Meyer](#), Senior Project Engineer

There can be no doubt that a majority of controls engineers in the packaging industry have at least heard of the term “PackML” by now. For the past two years, Dr. Bryan Griffen, co-chair of the OMAC Packaging Workgroup and Electrical and Automation Group Manager at Nestlé, has led a renewed effort to communicate the benefits of PackML at nearly every industry trade show and conference (Campbell, 2011). But understanding how PackML is actually implemented on a real machine, and having experience in designing and coding such an implementation is likely a different story. For many engineers in the packaging industry, and for nearly all engineers outside the industry, PackML remains an amorphous concept, not completely understood and missing a clear best practice for getting started. This Tactical Brief will set aside the discussions of ‘Why PackML?’ and instead focus on ‘How PackML’ so that

controls engineers in any industry can better understand the design and implementation process.

PackML - not just for the Packaging Industry

It is important to say up front that, despite its name and origins, PackML can be applied to any automated machine, regardless of industry. Technically, PackML is known as ISA-TR88.00.02 and was born out of the need to improve production efficiencies at packaging end-user locations and to reduce the amount of time and expense involved with integrating machines from different suppliers into a cohesive production line. The intention was to build a global standard for automated machine code architecture that could be adopted by packaging OEMs to ease the line integration and support processes. What emerged is a code architecture



Designing your first PackML implementation for machine control



that applies to machinery beyond the packaging industry.

The heart of the current PackML v3.0 standard is the machine state diagram. Starting in the early 2000's, the OMAC (The Organization for Machine Automation and Control) Packaging Workgroup (OPW) spent a good deal of time breaking down the operational sequencing of automated machinery and creating a standard model of logical states and transitions. The resulting state diagram for ISA_TR88.00.02 was released in 2008 and is shown in Figure 1. (ISA, 2008). Note that this model contains nothing specific about the machinery used in the packaging industry. Instead, the model uses a simple flow diagram to define a generic, standard nomenclature for how machines operate. All machines will have a state where the machine is running the intended process (Execute) and not running (Stopped or Idle). Similarly, all machines must apply some sort of fault detection and recovery sequence (Aborting, Aborted, Clearing).

The additional Held and Suspended branches were added by the OPW to further identify those times when the machine may be capable of producing something, but other

conditions are impeding production. The Suspended branch is for use if the machine is waiting (starved) for material from an upstream process or blocked by a downstream process. The Held branch is intended for operator-induced production holds such as a Pause function (See Figure 1 on page 4).

The transitional flow of the model is important. Observe that the model forces states to be activated in a sequential manner with exceptions only for Stop and Abort. Therefore the model is useful not only for monitoring the current state, but also for driving to the next state, and thus can serve as the high-level master for machine sequencing.

In addition to the standard state model, PackML provides for user-defined *modes of operation* that may allow or restrict access to certain states or branches. Think of modes as a third dimension of the PackML State Model that gives the model layers of depth, like floors in a building. Like a building, not all the rooms (states) may be accessible, and movement between floors is only possible at stairway locations (transition states). It is with the definition of these operational modes that the implementation process begins.

Designing your first PackML implementation for machine control

YASKAWA™ | SPONSORED CONTENT

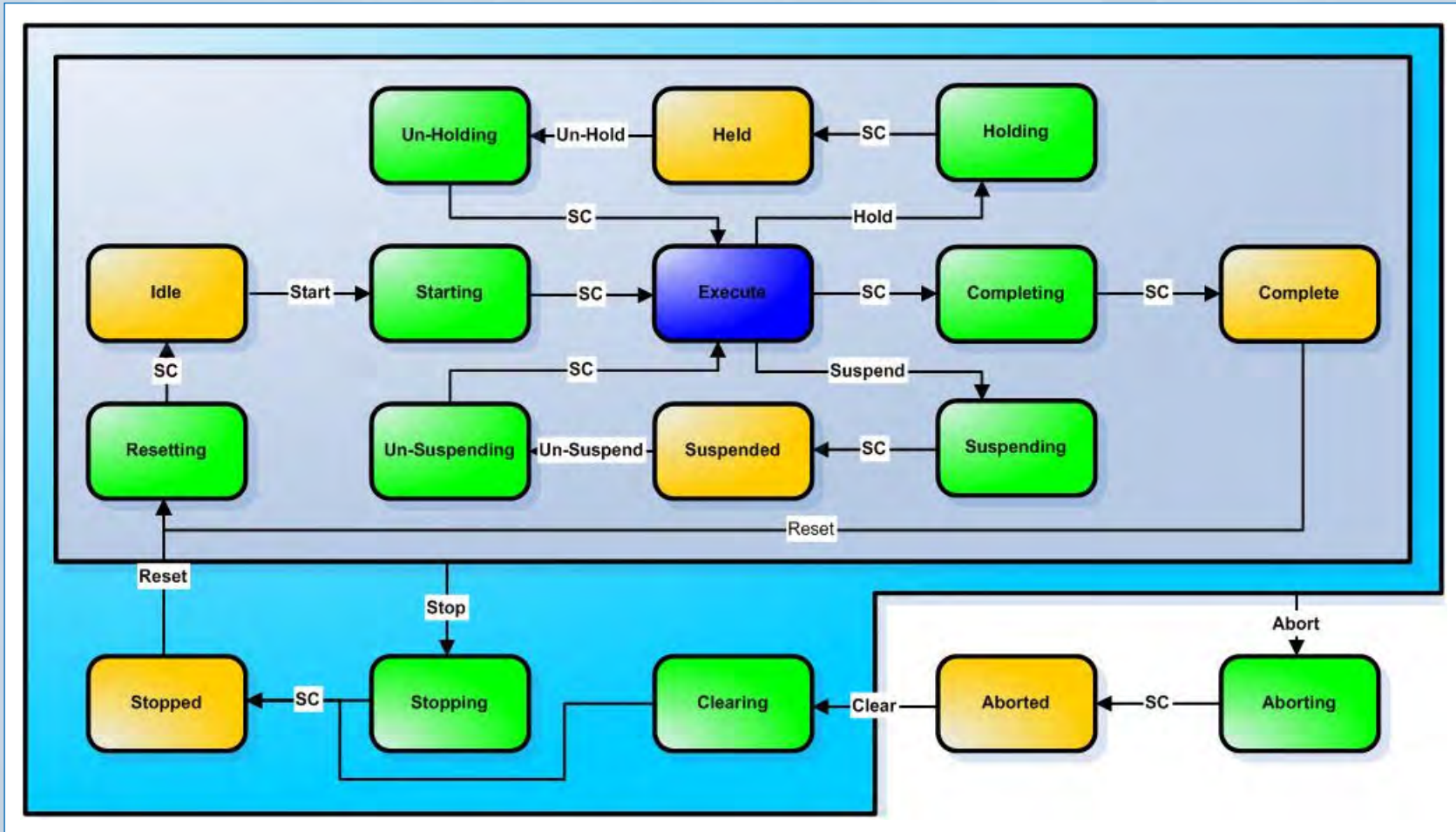


Figure 1: PackML State Model.



Designing your first PackML implementation for machine control



Your Implementation – Key Design Decisions

Define the PackML Modes to be used

The first major decision to make in your new PackML-based application is **which modes are included and what they are named**. Although PackML technically allows for an unlimited number of machine modes, most applications have some version of these three: Automatic, Manual, Maintenance. The names are user-definable and the programmer can also define which states are accessible when each mode is active.

Typically, *Automatic Mode* is used for normal production and includes full access to all the PackML states in the model. *Maintenance Mode* is often used to run all sections or individual sections of the machine in a ‘dry cycle’ manner for setup, debug, or testing. Certain branches such as the Held or Suspended branches are usually disabled while in Maintenance Mode. In many implementations, Maintenance Mode is entry-protected with a password that limits access to only authorized users. *Manual Mode* is used for manual operation of individual mechanisms on the machine, most often for setup, commissioning or debug by an authorized user. It may be best to disable virtually all of the PackML

States while in Manual Mode.

To complete the mode definition, the user will specify the states at which the mode can be changed. These transitional states are generally set to ‘quiet’ states such as Aborted, Stopped and Idle where the machine is not producing anything.

Assign functionality to PackML Modes and States

While the PackML standard does a great job at defining the state names and transitions, it leaves the decision of what happens in each of those states up to the user. Therefore, the second major decision is to **clearly define what machine functions happen in each state**. For example, if there are servo axes on the machine, when are they enabled – in the Stopped State?, during Resetting State? Where are servo axes disabled – during Stopping? Only during Aborting? When Aborted is reached? Another good example is in which state is a Homing function employed – Resetting? Starting? How about a CycleStop function? Should CycleStop code reside in Stopping or Completing?

All these decisions and more should be clearly identified



Designing your first PackML implementation for machine control

 | SPONSORED CONTENT

in a document so that all project engineers can achieve a uniform understanding of what-happens-when during the overall machine sequence. Using the PackML nomenclature and state model allows engineers to speak about machine operations in a common, high-level manner that improves communication and speeds development. Griffen comments that this step is crucial to realizing the benefits of PackML because it allows Nestlé to clarify the whole task of interpreting and implementing the state model. A clear specification can then be delivered to OEMs that is process-specific instead of hardware-specific. This allows OEMs to choose best-fit hardware so long as the machine controls can integrate horizontally and vertically into the Nestlé line. (Reynolds, 2011).

Modularize the machine code

The PackML State Model works best when implemented in a modular way. By that, it is meant that PackML provides the supervisory commands and status for high-level overall machine sequencing that can be passed down to functional code modules. The modules, in turn, send back completion status that the state model uses to move to the next state. By performing the next step to *separate the code into logical*

modules that match the physical machine construction, the foundation is laid for code that is more organized, more reliable, easier and faster to debug, and more easily reusable in other applications. For PackML, such a modular code model is further defined in S88:Make2Pack.

The ISA88 physical hierarchy for code modules contains six levels that range from the entire global company level down to each individual function. For this discussion, we will focus on the bottom three that relate to an individual machine:

- **Machine (also known as Unit, or UN):** a collection of related modules (mechanical and electrical assemblies) that carry out one or more processing activities
- **Equipment Module (EM):** a functional group of modules that carries out a finite number of activities
- **Control Module (CM):** the lowest level of control where a single function is executed. (ISA, 2008)

High-level PackML commands originate at the Unit

Designing your first PackML implementation for machine control

YASKAWA™ | SPONSORED CONTENT

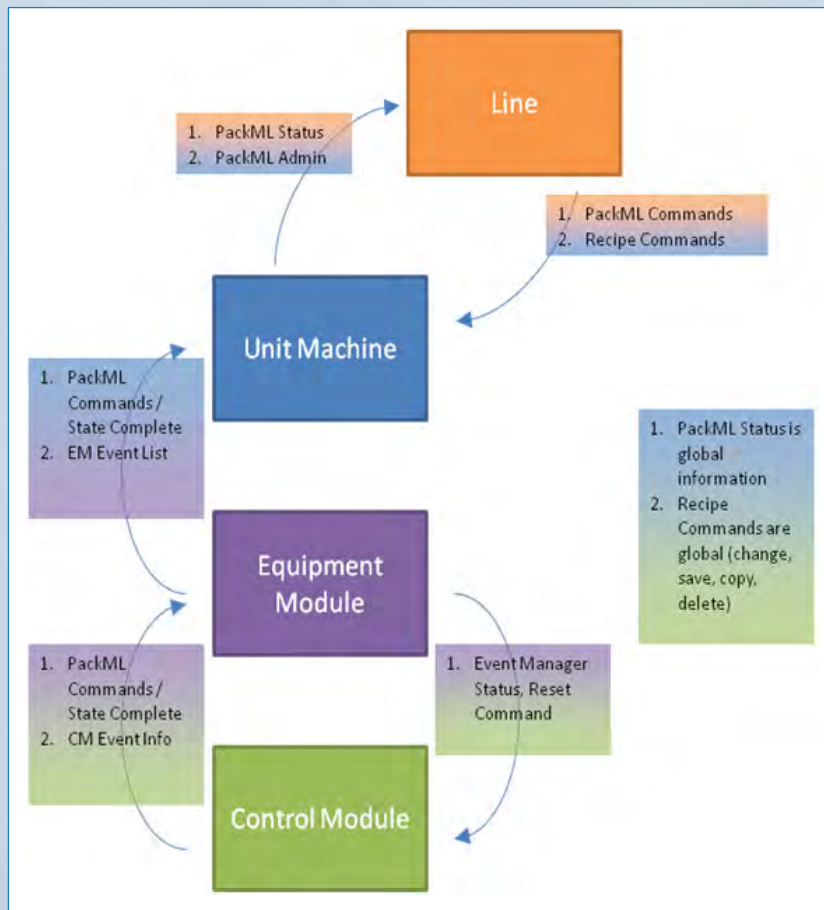


Figure 2: Data Flow in a PackML application (OMAC, 2009)

Machine level and flow down through all the Equipment Modules to the Control Modules as shown in Figure 2. In turn, the completion status of each module is reported and transmitted back up the chain.

For most applications, the code deals with only one single machine or unit. A machine is made up of one or more Equipment Modules, each of which contains one or more Control Modules. The real decision is how to define the EMs and CMs for logical and efficient operation. One could easily get carried away and define a Control Module for every device on the machine. However, given that a machine of medium complexity may have a hundred or so devices, this approach might be a bit impractical. Instead, it is better to focus on slightly larger groups of functionality where the *reporting of PackML State completion* is needed.

Try to avoid using a single CM in an EM. If that is the case, perhaps that piece of equipment performing a single function should become a CM in a larger EM. Likewise, if a CM has no individual reporting impact on the completion of a PackML State, then that CM functionality should be included elsewhere in a different CM.



Designing your first PackML implementation for machine control

YASKAWA™ | SPONSORED CONTENT

Sometimes, it may make sense to include multiple devices in a control module if they are linked in some way. For instance, if one servo is a master to another servo in a camming relationship, these two devices should perhaps be contained in a single CM since they are so closely tied and their *combined* status is what is most important to the high-level control.

Example: An automated case packing machine for snack bags has 9 servo axes and 7 key functions.

- Case Feeding
 - Sheet Lifter servo
- Case Erecting
 - Case Opener servo
 - Case Transfer servo
- Bag Sorting
 - Sort 1 servo
 - Sort 2 servo
 - Sort 3 servo
- Bag Loading
 - Bag Pusher servo
- Case Loading

- Case Elevator servo
- Case Closing
 - Case Closer servo
- Case Ejecting

One could set up 7 Equipment Modules in the machine. However, if that were done, only two would have more than one Control Module. Therefore it may be better to first divide the machine into larger functional groups such as ‘Bag Handling’ and ‘Case Handling’. If this were done, the resulting configuration might look like that in Figure 3 (see page 9).

This configuration splits the machine into two logical equipment groups which align with the two main inputs to the machine – bags and cases. All EMs contain more than one CM. Two of the CMs contain multiple devices that have high degrees of combined interaction and the number of EMs and CMs is reasonable for the project size.

Customize vendor-supplied templates

After making the three big design decisions just described it

Designing your first PackML implementation for machine control

YASKAWA™ | SPONSORED CONTENT

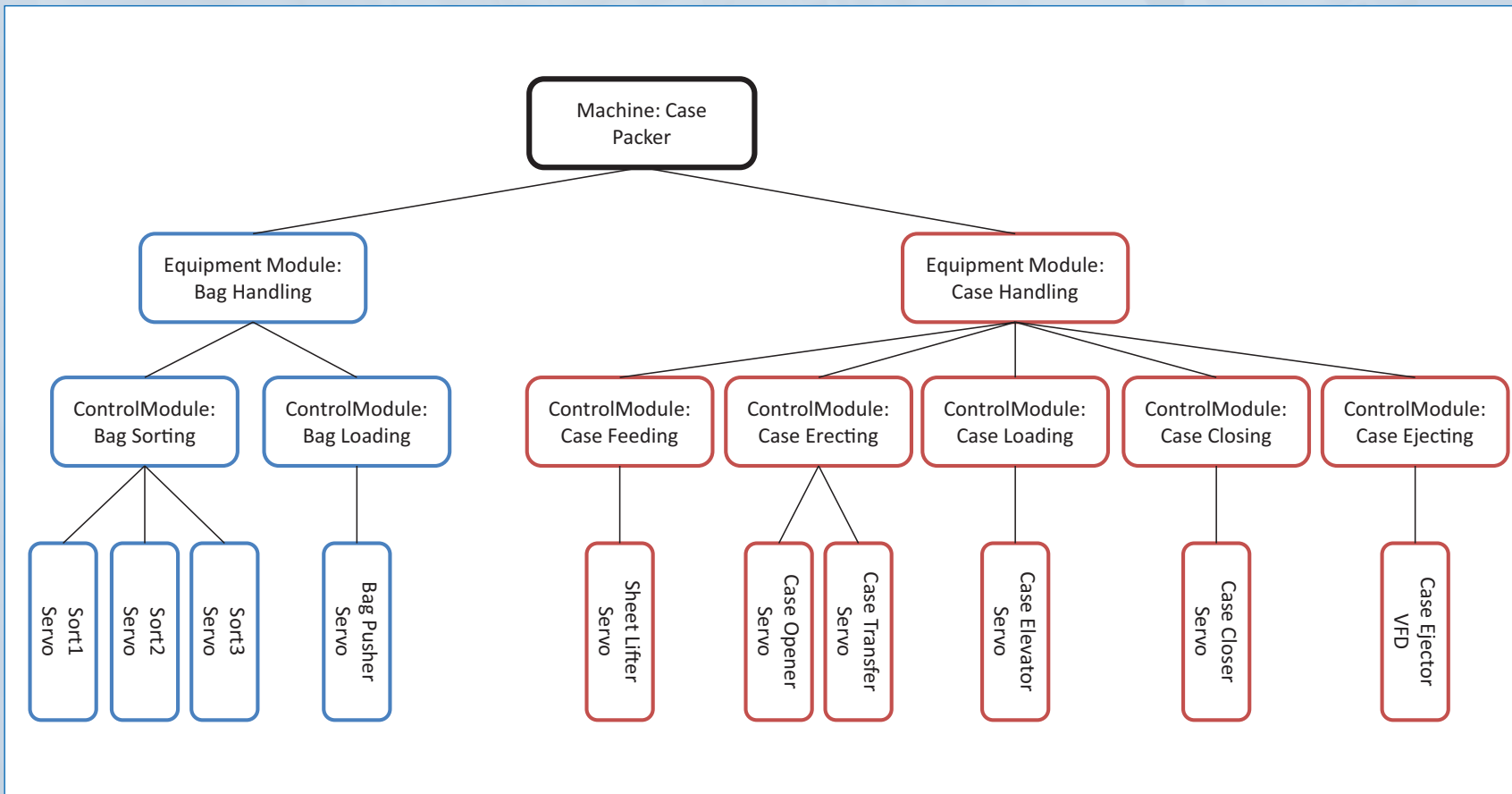


Figure 3: Module Architecture

Designing your first PackML implementation for machine control

YASKAWA™ | SPONSORED CONTENT

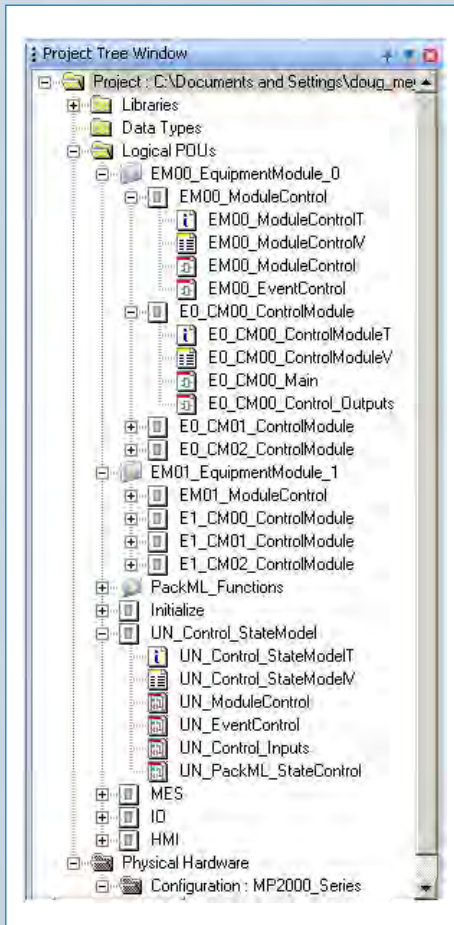


Figure 4: PackML Project tree

is time to write some code. This is an exciting, yet unnerving part of the process for one may wonder where to begin! Fortunately, several machine controller vendors have already created a starting point in the form of a PackML project template. These templates lay the foundation for the project by providing the PackML State Model code, a means to configure modes and states, a means for user definition of EMs and CMs, and example code for getting started. Depending on the vendor, these templates may be written in either pure ladder-based code or in the global standard IEC61131-3-based code.

Figure 4 shows a project tree from an IEC61131-3 based template that comes with two pre-defined Equipment Modules, each containing three Control Modules. Users can rename, add or subtract modules as necessary to fit the needs of the application. Key predefined code worksheets include PackML_Initialize, UN_PackML_StateControl, UN_Control_Inputs, UN_ModuleControl, EMxx_ModuleControl, and Ex_CMxx_Control_Outputs. (Yaskawa, 2012).



Figure 5: PackML State Control Function Block



Designing your first PackML implementation for machine control



```
32 (* Initialize elements being used*)
33 (* It is up to the User to determine how many valid modes are defined in the program and what
34 functionality each mode performs *)
35 PackML_ModeNames[1]           := 'Automatic';
36 PackML_ModeNames[2]           := 'Maintenance';
37 PackML_ModeNames[3]           := 'Manual';
```

Figure 6: Defining Modes in the PackML_Initialize worksheet

IEC-based templates can take advantage of the multiple languages available in IEC61131-3 to create a more understandable and readable version of code. The function block for UN_PackML_StateControl is written in Sequential Function Chart (SFC), a direct graphical representation of the state-transition block diagram. Control output worksheets

used for rolling up state completion status are written in Ladder Diagram (LD) since it is easiest to set and debug status coils in ladder-based code. Finally, configuration and command management functions are written in Structured Text (ST) since it is easiest to perform array manipulation and initialization using text commands.

```
91 (***** Initializing Equipment Modules and Control Modules to be used in the Unit Machine *****)
92 (* Each UNxx defined can have up to 16 EMs, each with up to 16 CMs *)
93 (* Enabled EMs and CMs must be consecutive and will be numbered starting from 0 up to 15 *)
94 (* Example: If 3 Equipment Modules are enabled, they will be numbered 0, 1 and 2 *)
95 (* The enabled EMs and CMs can be individually deactivated in the UN_ModuleControl and EM_ModuleControl
96 worksheets *)
97
98 UN00_Modules.EnabledEMs           := INT#2;
99 UN00_Modules.EM[0].EnabledCMs     := INT#3;
100 UN00_Modules.EM[1].EnabledCMs     := INT#3;
```

Figure 7: Defining the enabled EMs and CMs in the PackML_Initialize worksheet

Designing your first PackML implementation for machine control

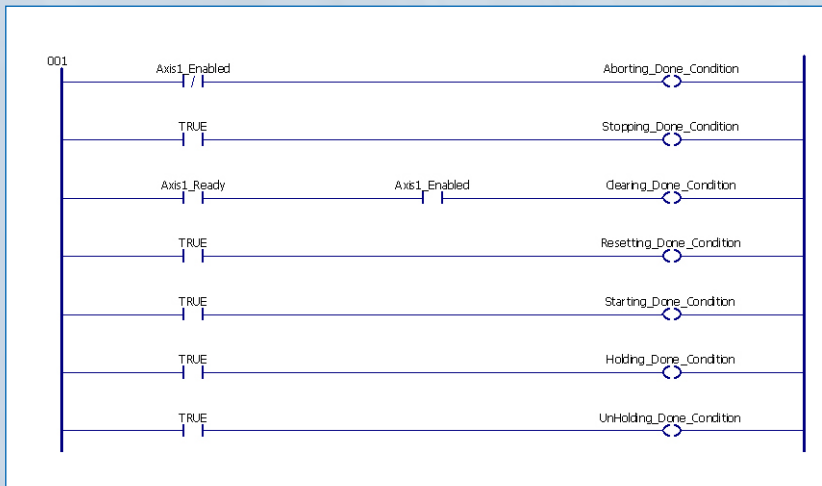


Figure 8: Ex_CMxx_Control_Outputs State Complete bits

In the template shown, the UN_PackML_StateControl worksheet contains the core code for the PackML State Model. The function block shown in Figure 5 (see page 10) accepts the high-level PackML Commands as inputs (the transitions), and sets the PackML States as outputs (the actual states). The block also monitors the completion status of each transitional state and moves the model to the next state. The function does not allow invalid transitions or

invalid changes of mode.

Users configure the system by editing the PackML_Initialize worksheet. Mode names can be customized and defined starting around line 32, as in Figure 6 (see page 11).

Likewise, users configure how many Equipment Modules and Control Modules are to be enabled starting around line 91, as in Figure 7 (see page 11).

Finally, users are able to programmatically determine when states are considered complete by editing the code in Ex_CMxx_Control_Outputs. As displayed in Figure 8, each Control Module has an output worksheet that sets a coil for each state complete bit that is fed back up to the main UN_PackML_StateControl. Users need simply add contacts to the ladder if there are other machine conditions that impact the process for each particular state.

Using PackML to Drive the Machine Operation

PackML is best used in a full implementation. By full implementation, it is meant that PackML is not just overlaid

Designing your first PackML implementation for machine control

YASKAWA™ | SPONSORED CONTENT

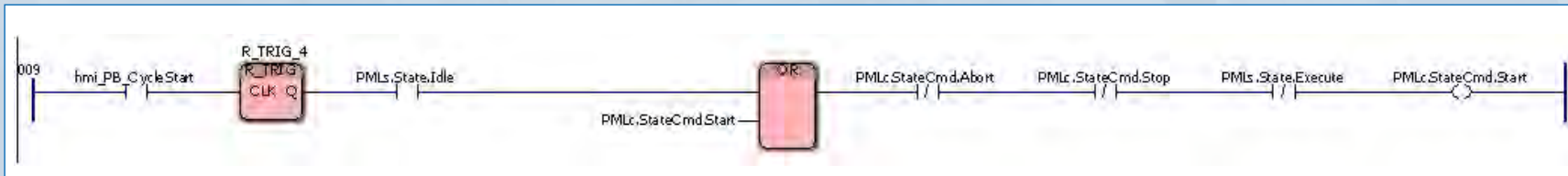


Figure 9: Start Command Logic

on top of an existing application for monitoring purposes only. Instead it is implemented from the foundation so that the organizational benefits of the modular approach can be fully realized. For this reason, inputs that bring the machine to an emergency stop condition should not just trigger actions independently, those inputs should instead trigger the PackML Command 'Abort', which will be automatically passed down to all active equipment modules. In the same manner, a Cycle Start input, if in the correct mode, should trigger the automatic sequence via the PackML Command 'Start'. All active equipment modules can then be programmed to respond either to the local PackML command 'Start' or the global PackML State 'Starting'.

Hooks for setting PackML commands

Most templates will have provision for users to patch into the PackML controls. For the template described, this is

done in the UN_Control_Inputs worksheet. The rung shown in Figure 9 allows the command 'Start' to be issued if the user conditions are right and the operator has pressed a button on the HMI. The command 'Start' latches in until state 'Execute' is reached, or 'Abort' or 'Stop' is issued.

Extracting useful data for production statistics

PackML-based applications make it easy to extract production statistics since the PackML Mode and State provide most of the necessary information. If the mode and state are 'Automatic' and 'Execute' respectively, then it directly follows that the machine is producing its intended output. If the state is 'Aborted', then the machine is faulted in some way. Many users construct their own statistical gathering functions, but some templates already have certain functions built-in. Figure 10 shows an example found in the IEC61131-3 template shown earlier, where a vendor-provided function

Designing your first PackML implementation for machine control

YASKAWA™ | SPONSORED CONTENT

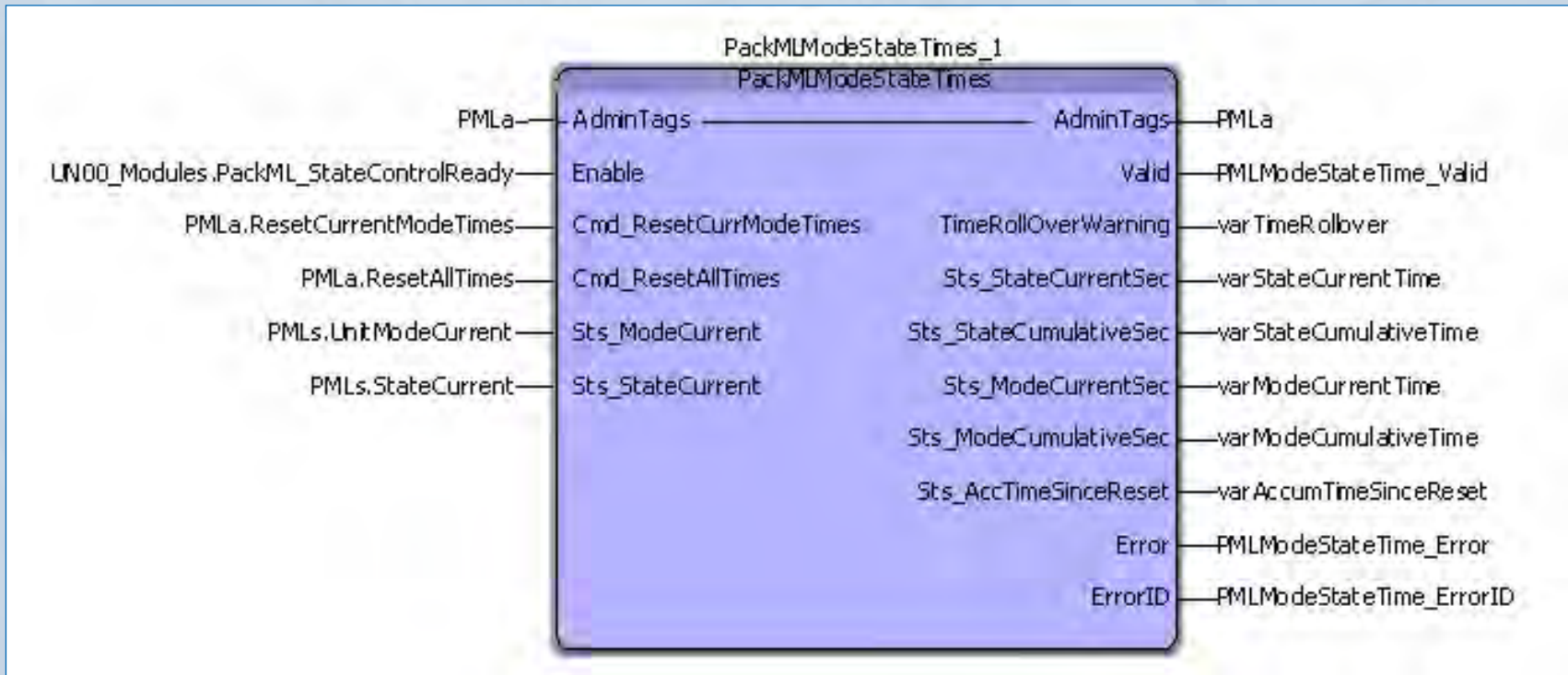


Figure 10: PackML Time collector for modes and states.

block called PackMLModeStateTimes continuously stores time information into the structured array of modes and states.

By comparing the current times in each state of Automatic Mode to the total elapsed time, the production manager can immediately view useful machine efficiency data.



Designing your first PackML implementation for machine control



Interfacing to upstream and downstream equipment


Although many machines can be considered ‘stand-alone’, a great many of them actually get installed into a production line among other machines. This is particularly true for the packaging industry, but often applies to other industries just the same. One of the promises of PackML is the ability to easily communicate status upstream and downstream to other equipment, even if that equipment was made by a different vendor and uses a different machine controller. A forthcoming specification from the PackConnect subcommittee of OPW will outline such a communication protocol standard over an industrial network. Until then, the basic PackML foundation provides for easy interfacing through digital I/O triggered by the current PackML Mode and State.

Conclusion

Although the mission to establish a standard for machine control code architecture was at first entirely based on the needs of the packaging industry, the PackML ISA-TR88.00.02

and ISA88:Make2Pack standards contain methodology that extends well beyond packaging to *all* automated machinery. There is a learning curve for proper PackML implementation. However, getting your first PackML project up and running is made easier by focusing on three critical design decisions:

- 1) which modes are included and what they are named,
- 2) which machine functions happen in which state, and
- 3) how can Equipment Module and Control Modules be organized into a configuration that logically matches the physical machine.

Beginning a project with pre-built vendor templates also reduces development time. By following the model of the PackML Standard, your application can be more organized, easier to commission and debug, more modular and transportable to other applications, and easier to add functions for data collection and production statistics. 



Designing your first PackML implementation for machine control

 | SPONSORED CONTENT

References

Campbell, Keith. (2012, April 3). OMAC Packaging – Is the Connect-And-Pack Message being heard?. Packaging World. Retrieved September 16, 2012 from <http://www.packworld.com/omac-packaging-workgroup-connect-and-pack-message-being-heard>

ISA. (2008). ISA-TR88.00.02 Machine and Unit States: An Implementation Example of ISA88. Retrieved September 16, 2012 from <http://www.isa.org/Template.cfm?Section=Standards2&template=/Ecommerce/ProductDisplay.cfm&ProductID=9999>

OMAC. (2009). P&G PackML Implementation Guide. Retrieved September 16, 2012 from <http://www.omac.org/content/packml>

Reynolds, Pat. (2011, April 9). Nestle makes packaging a strategic priority. Packaging World. Retrieved September 16, 2012 from <http://www.packworld.com/controls/strategy/nestl%C3%A9-makes-packaging-strategic-priority>

Yaskawa America, Inc. (2011). PackML Template. Retrieved September 16, 2012 from <https://www.yaskawa.com/links?type=documents&docnum=EC.MWIEC.34&name=>



P&G PUTS PACKML THROUGH ITS PACE

With help from an engineering consultancy service, P&G's corporate engineers re-programmed a Pace bottle unscrambler so that it would be compliant with the ISA-TR88.00.02 PackML Standard.

By **Dan Amundson**, a control systems engineer in P&G's corporate engineering group and **Jason DeBruler**, P&G engineer and author of the OMAC PackML Implementation Guide

Packaging machines have many more features today than in the past. Original Equipment Manufacturers (OEMs) are addressing issues that have an impact on their customers' bottom line. They design their machines for greater product diversity, higher rates & performance, easier changeover, improved human interfaces, built-in quality check systems—the list goes on.

Automation technology has opened the doors for significant innovation that addresses many of these issues. However, a number of hurdles remain, for both end users and OEMs. ARC Advisory Council identified some of the problems in a 2008 report:

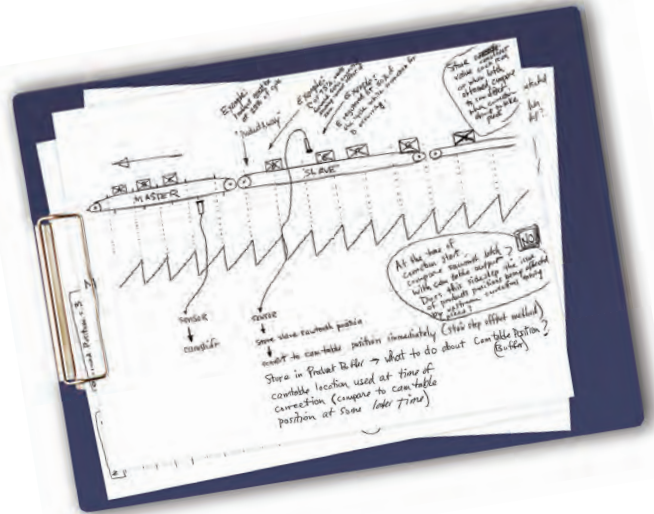
“Packaging machinery is increasing in complexity. More machines are including many axes of servo drives, integrated robotics, vision systems, machine fault/performance analysis, and communications for integration with line and production management systems. However, there are no consistent standards for software development—even between like machines from the same OEM. As a result, operator and technician training and support are becoming increasingly challenging; and horizontal and vertical integration remains difficult and time consuming.”

End users spend significant money and time engineering the integration of OEM machine control systems into cohesive and

SPONSORED BY



NOTEWORTHY



When our engineers work on a project with you, they share everything that's in their heads. Sometimes that means turning over 100 pages of notes, formulas and drawings. Other times, it's simply having straight forward conversations.

At the same time, Yaskawa customers share their proprietary knowledge because they can trust us. Because we give their challenges a lot of thought. Because when they talk to us, they know we are on their side.

Trusting the guys across the table from you with your most important product secrets. That's noteworthy.



YASKAWA AMERICA, INC.
DRIVES & MOTION DIVISION
1-800-YASKAWA YASKAWA.COM

continued

P&G puts PackML through its Pace

well managed packaging lines. Their engineers must ensure that OEM control code performs as needed on machines coming into their plants. The engineers also have to structure software interfaces between machines in a line to ensure line coordination. Overall Equipment Effectiveness (OEE) is another key priority, so the engineers must transfer OEE data from individual machines into business information systems.

Once this fundamental engineering work is complete, the end user has to focus on the operators who need to be trained on how the machine works; their knowledge of its operational states is crucial. Technicians are in the mix, too. They need to have some level of training in machine-control software troubleshooting for when the inevitable happens: the machine stops, it won't restart, and it's not obvious why.

The need for a standard

As the ARC document points out, there have been no consistent machine software structure standards. Without intervention, machine software will continue to become even less consistent, negatively affecting both end-user and OEM business bottom





P&G puts PackML through its Pace

lines. This lack of consistency drove the industry group OMAC (The Organization for Machine Automation and Control) to develop PackML (Packaging Machine Language) standards for improving machine-to-machine integration, extending operator capability, and improving enterprise integration. The official ISA-TR88.00.02 PackML Standard was released in August 2008.

PackML builds off a proven ISA industry standard in S88. Benefits include:

- A consistent look and feel for the operator and technician
- A foundation for vertical and horizontal integration
- Standard information in/out of a machine
- Packing line plug-and-play functionality
- More consistent end user specifications
- Faster software development time

- Reduced debug time through more robust and modular software programming
- Ultimately, efficiency in reusable hardware and software components and machines that cost less to build, use, and maintain.

It's important to point out that it isn't just the end user who stands to gain from PackML. The OEMs benefit, too, because by using the concepts provided by the standards, they can focus their software effort on competitive-advantage features instead of wasting valuable time on basic programming and integration features required for any machine.

P&G is a leader in adopting PackML concepts to reduce integration and support costs. P&G and other major end users of packaging equipment now demand use of this programming standard on their key OEM equipment. To aid machine automation software developers in achieving a clean and efficient implementation of PackML, P&G developed a PackML Implementation Guide (available at www.omac.org). The guide, software template, and help files aid end users, OEMs, and

SPONSORED BY





P&G puts PackML through its Pace

integrators alike to efficiently apply the PackML concepts based upon the ISA standards.

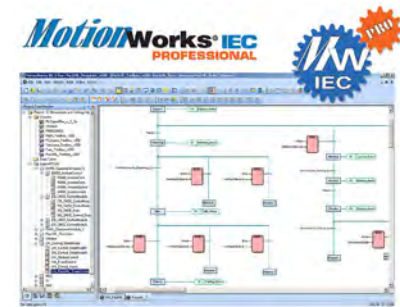
So what happens when a packaging machine comes into a P&G plant and it's not PackML-compliant? In the case of a Model 1000 plastic bottle unscrambler from Pace Packaging Corporation, P&G elected to convert it to the PackML standard. The machine, which has a rated capacity of 350 bottles/min, was installed in 2009 in a line capable of running a variety of liquid detergent products. P&G has worked with many consultants on applying PackML standards. On this particular project, the firm worked with Tata Consultancy Services (TCS) to convert the unscrambler's existing software to the concepts provided by the PackML and ISA standards.

Objectives

The project objectives included:

- Understanding how the machine worked
- Decomposing the machine into a modular structure as defined by the PackML and ISA standard (e.g., Equipment Modules, Control Modules, etc.)
- Implementing needed PackML states (e.g., Starting, Execute, Stopped, Aborting, etc.) and directing action to the appropriate Equipment Modules and Control Modules
- Building the PackML operator interface

» MotionWorks IEC Programming Software



- » MotionWorks® IEC, Yaskawa's IEC61131-3 programming environment, encourages the programmer to optimize the usage of several programming languages within one development package. Ladder logic, structured text, PLCopen function block, sequential function chart, and instruction list programming are all supported. Libraries of reusable code for camming, gearing, PackML, and others reduce development time and improve machine commissioning time.

[LEARN MORE »](#)

SPONSORED BY





P&G puts PackML through its Pace

- Following the OMAC PackML Implementation Guidelines
- Validating and commission

Jason DeBruler, P&G engineer and author of the OMAC PackML Implementation Guide, worked with TCS on this conversion to ensure the S88 concepts were well understood and executed. A week of effort was spent on the software conversion. Some time was also spent understanding how the machine worked.

TCS implemented a modular, structured, standard code on this machine and met the project objectives. TCS applied PackML using the PackML Implementation Guide and template endorsed by the OMAC organization. The PackML implementation resulted in PLC code and HMI screens that are simpler to understand and troubleshoot. Some of the benefits include:

1. Simplification of the machine operation. Per S88, the unscrambler now runs using defined states and modes. The key program conditions used to perform an action are based on the mode and state. The code is much simpler than conventional

permissive logic that typically varies from programmer to programmer.

2. PLC code now uses modular programming techniques. This means that the software code is organized into reusable blocks of code. This has greatly increased the reusability of machine logic while making the software easier to read and more robust.

3. Recipe control is simplified. The user can now do Save, Copy, Delete, and Save As for existing recipes. Previously, only predefined recipes existed and adding a new recipe was a big task.

4. Alarm management is simpler now. In the existing logic there was no categorization of alarms. PackML alarms are now divided into categories. Depending on the situation, the appropriate alarm category action is taken. For example, if it is a Category 0 alarm, the machine will go from the Execute to the Aborting State. If it is a Category 2 alarm, the machine will move from the Execute to Stopping State. In the existing logic, the user was not able to identify which alarm actually came first and stopped the machine. This can now be identified.

SPONSORED BY





P&G puts PackML through its Pace

Also, after implementing the PackML template, a unique code is given to each alarm that can be passed to the user's MES (Manufacturing Execution System) solution to monitor the uptime and downtime of the machine.

5. HMI screens provide more information. The PackML screen lets the user know what state the machine is in. This makes it easier for a technician to identify which part of the software code is currently active. Further, alarm and recipe screens are more informative and user friendly.


"The OMAC implementation guide and template simplified the process of applying PackML," says DeBruler. "These tools have helped engineering firms, in this case TCS, quickly get up to speed and apply PackML."

"This project also opened our eyes to how PackML can help technology providers offer a much better service to OEMs and end users by delivering easy-to-implement PackML templates," says Dan Amundson, a control systems engineer in P&G's corporate engineering group. "Additionally, MES suppliers can

aid end-users with OEE management by integrating PackML into their products, further simplifying vertical information."

Next steps

So where do P&G and Pace go from here? P&G is already reaping the rewards of PackML through lower machine-to-machine and business information system integration costs. The company continues moving forward with plans to broadly implement PackML on packaging platforms.

Pace Packaging's controls team is fully onboard with this PackML execution for their machine. Pace has embraced the PackML concept and intends to make it available to other customers and other machines they offer. Pace president Ken Regula sums it all up this way: "The benefits of the PackML standards are clear and will provide the end-user who embraces this system with readily measurable results. OEMs should not hesitate in starting to implement PackML as their programming standard." 

SPONSORED BY



EVOLVING STRATEGIES FOR PACKAGING MACHINERY SPECIFICATIONS



Machines have evolved to incorporate servos, high-speed networks and software smarts. Your methods for specifying them need to evolve, too. Here are three paths you can take.

By [Keith Campbell](#), Contributing Editor and Automation Blogger

Packaging machines have gone through three major periods of evolution, as have many end user's strategies for writing packaging machine specifications. Let's explore why you might consider changing your company's specification strategy.

First generation machines were all mechanical, perhaps powered by an electric motor. Second generation machines were electro-mechanical, containing add-on sensors and actuators that were controlled by relays, solid-state logic cards and eventually programmable logic controllers (PLCs). Today's third generation machines are mechatronic machines in which the electrical, control, computer, and mechanical aspects comprise a fully integrated package, designed from the ground up using a mechatronic paradigm.

The movements in these third generation machines are typically initiated by a series of servos that are coordinated in time and space using sophisticated software and high speed networks. It is not uncommon for such machines to see a reduction of up to 80% of its moving parts when compared to its second generation predecessor and for the intellectual property content of the machine to be more than 50% in software rather than 80% or more in mechanical design. These mechatronic machines have resulted in lower costs, shorter lead times, the use of less floor space, faster startups, higher operational effectiveness and efficiency (OEE), more agility and safer workplaces.

During the electro-mechanical period, it was reasonable for end users to create separate mechanical and electrical

SPONSORED BY





continued

Evolving strategies for packaging machinery specifications

PAIN KILLER



It's difficult to design packaging automation software that will last the lifetime of your machine without the right engineering capabilities.

DIAGNOSIS

You no longer have the luxury of a large engineering staff. True packaging automation expertise is increasingly rare. These limitations put projects at risk.

REMEDY

Combine Yaskawa's expert automation engineers with your staff to meet ever-increasing customer performance expectations. You will match specifications you never thought possible with our motion control technology and packaging application knowledge.



Lacking the time, resources and confidence to create elegant and reliable packaging solutions? Call Yaskawa today for just the relief you need.



YASKAWA AMERICA, INC.
DRIVES & MOTION DIVISION
1-800-YASKAWA | YASKAWA.COM



Get personal with Yaskawa.
Call Doug Meyer today.
1-847-887-7086

IT'S PERSONAL
YASKAWA

specifications for their machines, and many did so. These electrical specs included control requirements that were heavily influenced by the steadily increasing use of PLCs. At the time that many of these plant and corporate specifications were being developed, PLCs were relatively new, unproven in the eyes of many, and expensive. A memory card might cost \$2500 and be one of five or six such cards making up a processor chassis. There were scores of PLC manufacturers in multiple countries at a time when worldwide overnight express was not yet a reality, consensus standards were mostly regional or national in scope, and we had not yet entered into the global manufacturing economy. Specs at many sites solved these issues by calling for the use of a particular, regional preferred automation vendor.

Today we find ourselves in different circumstances and with three predominant strategies in play for preparing controls specifications for packaging machinery.

Strategy one is no specification at all -- just let it up to the machine builder. This strategy would have worked with first generation machines and MIGHT be ok for completely stand alone machines in operations that can tolerate low OEEs. I've seen this strategy in practice, sometimes resulting in machines




Evolving strategies for packaging machinery specifications

arriving at the dock that couldn't be used. I wouldn't recommend this strategy to anyone. It is like taking a trip without a destination in mind, and in that case, any road will get you there.

Strategy two is to continue with the specifications that have evolved over the years and that rely primarily on sticking with one automation partner in the hope that it will simplify parts, training and connecting systems together. I practiced this strategy during the 80's and 90's, but, in today's fast-paced world where hardware is so cheap, many of the reasons that led to adopting this strategy are no longer valid. The preferred legacy suppliers are slowed down by concerns with backward compatibility and by the extra effort involved in developing proprietary platforms and communications systems. One of the saddest statements I ever heard in my 35+ year career as an automation professional was when a colleague at a leading food company stated that his company had decided it was ok to forgo innovations until they could be brought to them by their preferred automation partner. The preferred supplier approach can easily limit innovation and increase purchase costs.

Just as generation three packaging machines start out with a clean sheet design, specification strategy three rewrites user

specifications based upon functional needs and accepted international standards. It takes advantage of the economies of scale and the speed of developments in consumer and IT markets by using PC platforms along with open-system standards for software and communications. It opens up the possibility of gaining early competitive advantage, but requires a certain level of competent knowledge within plant operations. I would recommend this strategy to companies today who aspire to be world-class.

Some standards, like OMAC's PackML may be employed in either proprietary or open control architectures. Other standards may only be available within open architectures. Those of us old enough to remember IBM during its mainframe days will remember the saying that "no one ever got fired for buying IBM". This meant buying IBM proprietary designs. But their closed architecture eventually led to the downfall of their hardware business as the industry moved on to open PC platforms. Now we have Apple making a move on the industry with their more closed design for phones and tablets, but on a global basis, more open Android platforms are gaining substantial ground. And so it goes. 

SPONSORED BY





NESTLÉ MAKES PACKAGING A STRATEGIC PRIORITY

Profitably proud of its controls technology in process automation, Nestlé now sees bottom-line opportunities by paying more attention to packaging line efficiency.

By **Pat Reynolds**, Packaging World Editor

A significant change in focus is underway at Nestlé, the world's largest food and beverage company. Packaging line efficiency, historically a responsibility left to outside sources and rarely viewed as an in-house strategic asset, is being carefully examined to see how it might make a better and more consistent contribution to the bottom line. And right smack dab in the middle of this entire initiative is our old friend PackML, or, as it is more officially known, the ISA-TR88.00.02 PackML Standard. Essentially a packaging machine programming language, it's a standard that Nestlé would like to see embraced by packaging machinery manufacturers around the world.

Bryan Griffen, electrical and automation engineering group manager who is based at Nestlé headquarters in Vevey,

Switzerland, is knee-deep in Nestlé's PackML initiative. He'll talk about what it all means in a presentation at the Packaging Automation Forum April 26 in Chicago (see sidebar). Here's a sneak preview.

Packaging World: What is your role at Nestlé HQ?

Bryan Griffen: Setting the direction and strategy for how Nestlé does electrical and automation engineering worldwide. We don't develop particular solutions. We develop methods for Nestlé divisions. Organized as Nestlé is into numerous different operating companies at market level, those operating companies have the final say in how they implement standards coming out of Switzerland. If they've got local regulations that come into play or financial reasons to go with a different solution on a particular project,

SPONSORED BY





Nestlé makes packaging a strategic priority

then that's their prerogative. But the general concept of how we do engineering comes out of our group in Switzerland.

Considering how visible and active consumer packaged goods companies like Procter & Gamble, Frito-Lay, and SAB Miller have been in promoting the benefits to be gained by embracing PackML, is it fair to ask why it's taken Nestlé a little longer to get on board?

It's a fair enough question, though I should point out that it isn't as if we haven't been involved in PackML at all. You'll see Nestlé names in some of the documents that have emerged from the OMAC Packaging Workgroup. What you have not seen is a strong and visible Nestlé involvement from the central head office. There are two reasons for that. First, we haven't had the people here at the central office to tackle things like PackML. But now we do. I have a couple of packaging engineers working for me, packaging automation engineers. We're starting to build up some mass that will allow us to do some things. The other reason is because Nestlé is a bit like an aircraft carrier. It doesn't turn very quickly or easily. But if it does make a turn, get out of its way because it's not about to stop. So it's just taken time for Nestlé to see the value in PackML and to start moving in that direction with support from Vevey.

It's interesting to hear you speak of new hires in packaging engineering when everywhere else we hear about staff reductions in that area. Sounds like you are bucking the trend.

It's probably true. We're bucking our own internal trend, too. Traditionally, Nestlé made its living off of process manufacturing. We're experts in process automation and PID-type control for process equipment. Our recipes have

» MPiec Family of Machine Controllers



- » Featuring Yaskawa's motion engine integrated with IEC61131-3 and PLCopen programming standards, these controllers provide full scalability from single to multi-axis applications within the MotionWorks IEC platform. The single axis MP2600iec is integrated with a Sigma-5 SERVOPACK, providing a compact single axis solution. The MP2300Siec and MP2310iec provide up to 16 axes of control while the latest addition, the MP3200iec, controls up to 62 axes.

[LEARN MORE »](#)

SPONSORED BY





Nestlé makes packaging a strategic priority

been our key differentiating factor—that and how we make products. Packaging has been another story. We've left packaging to a group we call Packaging Engineering, but they focus primarily on package design, the material used, how the package should be opened and re-closed, and so on. So they would come up with their design and hold it up and pretty much say, "Who can package this for us?" As a result, it was largely packaging machinery OEMs who were shaping our packaging operations. And then we'd hire an outside integrator to take all those disparate OEM machines and make them work together in a packaging line.

So what has changed?

We've come to realize that in order to improve the bottom line, in order to drive effectiveness and efficiency in our operations, we need to take a more active and direct role in defining what equipment from these OEMS looks like. So yes, we are bucking the trend in that we are building up a team that can give some specification and direction to OEMs on how to create the automation for the packaging systems. We haven't done that in the past.

Isn't there a downside when you wade into a packaging machinery OEM's design specs or dictate what their controls preferences should be?

I totally agree. What you have to understand is that Nestlé has had a 30-plus year relationship with Rockwell Automation when it comes to process automation. Through that Packaging Engineering group I mentioned earlier, which consists of package designers more than operations engineers, our process automation specs made their way into the hands of the packaging machinery OEMs of the world, who then assumed that the "Nestlé spec" for packaging machinery was Rockwell. So the packaging machinery OEMs would say, not all of them but some would say, "Okay, if you really want us to use Rockwell we will, but it will cost you twice as much and we can't guarantee the performance anymore." They reacted that way not because there was anything wrong with the Rockwell technology components, but simply because some of the packaging machinery OEMs had gotten accustomed to other ways of designing the controls architecture of their packaging machines and now we were asking them to turn their backs on a lot of the progress they felt they had been making.

SPONSORED BY



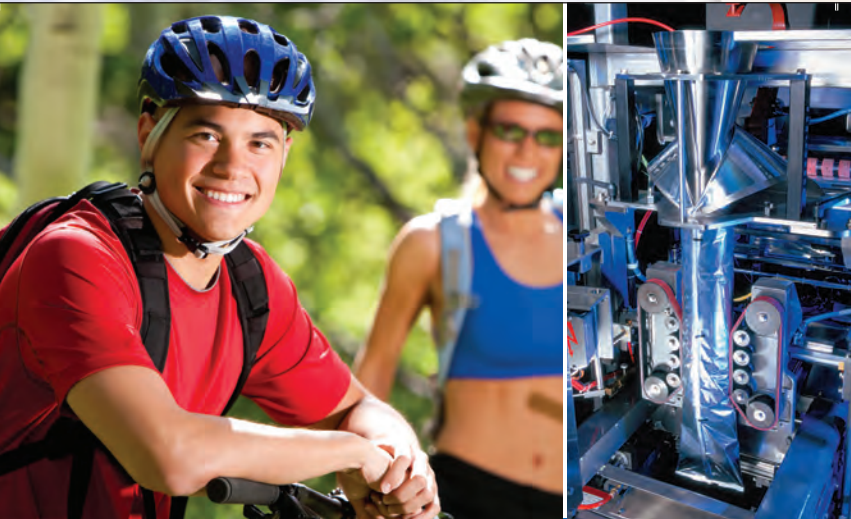


continued

Nestlé makes packaging a strategic priority

How would you describe the new attitude at Nestlé where controls platforms are concerned?

We want to make it clear that on the process side of things, where we own the technology, we will continue to be very dogmatic about whom we will use for process automation and how it will be used. But when it comes to packaging, it's the OEMs who build the machinery, not us. We are now saying look, we'd like you to use this international standard called PackML. We'd like to implement PackML in a way that lets us do two things. First, connect your machine with someone else's machine in one of our packaging lines. And second, provide us more visibility, with a way to get information out of our packaging machines. We have several approved PLC suppliers that you may use. You know better how to build your packaging machines, and you know if a Rockwell solution or a Siemens solution or a B&R or Schneider solution works better. Just make sure that you implement it with PackTags according to the PackML State Model. Our goal is to avoid exorbitant integration costs and to avoid having systems that don't let us move data readily into the MES layer.



QUALITY

ENJOY THE RIDE

Your business will coast along when you use Yaskawa's best-in-class servo systems. Our broad range of servos can be matched with single-axis to 62-axis motion controllers for a system solution with the highest quality and reliability in the business.

Take a ride to better performance. Call Yaskawa today to learn how we can put our innovation in motion for you.



MPieC Controller

Featuring Yaskawa's motion engine integrated with IEC61131-3 and PLC open programming standards, these controllers provide full scalability from single to 62-axis applications within a single programming platform.





Nestlé makes packaging a strategic priority

Is MES at Nestlé a home-grown solution or something a vendor provides out of the box?

A little of both. We use Siemens, Rockwell, and Wonderware as our defined MES providers. But the solutions they've provided have typically been customized. I guess you could say they've been "Nestlé-ized" in that there are certain KPIs and things we want to track in very specific ways. So where out-of-the-box solutions have been implemented, they've been modified to fit Nestlé requirements.

You mentioned the word "visibility." Is it any more important now than it was in the recent past?

What's newly important at Nestlé is bringing visibility to our packaging operations. Our MES solutions have been built around process automation. Now that those solutions are stabilized and we see the benefits we've gained, we'd like to see additional manufacturing benefits by moving those kinds of solutions into the packaging hall.

How are packaging machinery OEMs responding to all of this?

It's a mixed bag. You have the early adopters who took to it right away and are doing all they can to meet our wishes. But

others are a little more resistant. One of the messages I hope to deliver at the Packaging Automation Forum is that PackML offers substantial benefits to the OEMs. One in particular is that by adhering to a standard like this, they don't have to share as much intellectual property with the integrators of the world or with us, for that matter. We wouldn't have to care about the fine details of the programming they use, whether it's ladder logic, or function blocks, or whatever else they choose. All we see is the PackTag we want to see and where we are in the State Model.

Tell us a little about the demo you are working on.

It's a pilot application with several of our key technology suppliers. We are providing them with a standard specification on how to implement the PackML State Model on a simulated packaging machine likely to exist in a typical packaging line. We're also developing a communications protocol—it happens to be based on the Weihenstephaner protocol—and insisting that the components from each technology supplier be able to communicate machine states among themselves. And we want that communication to take place without needing a separate PLC but rather by implementing the Pack Tags correctly and using the communications protocol

SPONSORED BY





Nestlé makes packaging a strategic priority

as specified. It's been successful in simulation mode, so now we're moving to actual, physical implementation—a packaging robot, for example—at the next level. We did find some holes, and that's what we'll be working on in the near future with the OMAC Packaging Workgroup, trying to determine how we can plug those gaps or at least put some application tech notes or guidelines around the standard so that people understand where to be careful of the gaps and how the gaps have been filled by others.

What will this do for Nestlé's business?

It will allow us to write a detailed spec on how to implement PackML that can then be delivered to the OEMs stating clearly that these are the specific tags that we need to see in the system. It will clarify what we mean when we say we are in a specific state in the State Model, because what may be interpreted as a Hold state by one OEM may be interpreted as a Suspend state by another OEM. It will bring clarity to the whole task of interpreting and implementing the State Model. Then we won't have to be so dogmatic with the hardware selected by the OEMs. We can open the hardware possibilities available to the OEM to get their machine to do what it needs to do—as long as they implement it in such a way that it can integrate horizontally into the rest of the machine's and the rest of the packaging line's hardware components as well as provide the vertical links we need to our MES.

These business benefits you describe revolve around your dealings with packaging machinery OEMs. What about day-to-day operational gains in things like productivity, efficiency, and improved asset utilization?

Greater visibility into packaging operations and the ability to tie those operations

» Sigma-5 Servo Motors & SERVOPACKS



- » The Yaskawa Sigma-5 servo family features rotary, linear, and direct drive motors in sizes from 3W to 55kW. Sigma-5 SERVOPACKS feature "Tuning-less mode" which continually adjusts servo gains without software configuration or setup by the user. A vibration suppression feature utilizes standard 20-bit absolute position feedback to overcome mechanical resonance and reduce machine settling time. Options are also available for integrated IEC controls, EtherCAT, and enhanced functional safety.

[LEARN MORE »](#)

SPONSORED BY

 **YASKAWA™**

Nestlé makes packaging a strategic priority

into MES should lead to productivity gains. As for efficiency and asset utilization, they can only improve because the State Model gives us better ways to deal with faults and operator training.

As the newly designated co-chair of the OMAC Packaging Workgroup, what kind of goals do you have for the group?

One key thing that seems to be missing is a communications protocol. We have definitions of PackTags and of the State Model, but we don't have a way for those to be communicated consistently right now. We need a complete strategy for communicating machine-to-machine as well as for vertical

communication to MES. Also, as is the case with any spec, there is a little too much room for interpretation when it comes to PackML. I think it would be helpful to specify more tightly both how we interpret the PackTags and what they mean to us. That would lead to more consistency from OEM to OEM. And finally, I think the OMAC Packaging Workgroup needs to do a better job of defining what the benefits of PackML are to the OEMs. We need to sell them on this concept, because going forward we'll need to get more buy-in from them.

Last question. Is top management at Nestlé fully on board where PackML is concerned?

Yes.



SPONSORED BY

