Paper ID #41055

Implementing PackML in the Engineering and Technology Curriculum

Dr. Maged Mikhail, Purdue University Northwest

Dr. Maged B.Mikhail, Assistant Professor, Mechatronics Engineering Technology Ph.D., Electrical Engineering, Tennessee State University, Nashville, Tennessee, August 2013. Dissertation title: â€Development of Integrated Decision Fusion Software System

Implementing PackML in the Engineering and Technology Curriculum

Abstract

PackML (Packaging Machine Language) is an automation standard widely adopted within the packaging industry to establish standardized control protocols for packaging machinery. Development of the PackML standard was facilitated by the Organization for Machine Automation and Control (OMAC), garnering substantial recognition across the packaging sector. The integration of PackML has yielded notable benefits, including heightened operational efficiency, cost reduction, and increased machine adaptability.

The paper explores the conceptual foundations of PackML and the practical aspects of its implementation. It highlights four dedicated laboratory exercises created to empower students with hands-on experience in utilizing PackML with modern automation systems. These exercises aid in acquainting students with standardized and contemporary programming processes, often absent from university laboratory environments. Students familiar with industry processes are better positioned to secure higher-paying employment opportunities, thus elevating their professional prospects.

These outcomes are a direct result of the ongoing collaboration between the author and OMAC, driven by a shared commitment to promote the adoption of PackML within academic settings.

Introduction

Automated machine end-users commonly encounter difficulty in integrating equipment and packaging machines due to communication and compatibility issues. Purchasing new equipment or upgrading automation hardware requires large capital investment and results in a production halt. PackML presents a viable alternative wherein the programming and data interface of different machines are standardized as shown in Figure 1.

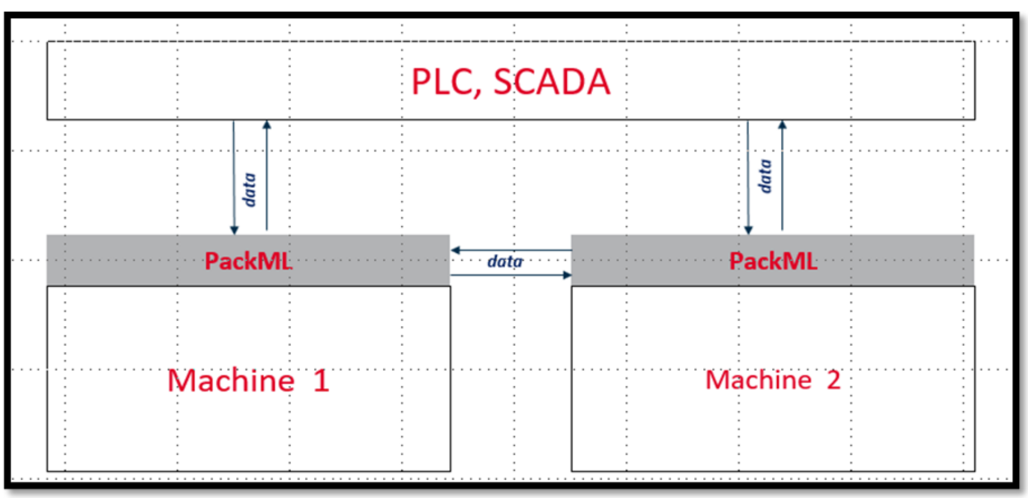


Figure 1: PackML provides a top-level data interface between machine peers.

This solution enables the integration of an older machine into digitalization concepts for which it is impossible to integrate machine data interfaces or for which there is no communication option with the other machines. This solution translates the machine's native data interface PackML tags into PLC controllers. The information necessary for this purpose is generated through a sensor system and physical signals are collected through the module periphery. No intervention in the existing automation system is necessary. Configuration is done exclusively through the Human Machine Interface (HMI) or web interface of the Control System. The outcomes associated with the implementation of PackML include:

- A consistent 'look and feel' to operators and technicians.
- Establishment of a foundation for vertical and horizontal integration.
- Standardization of information sent and received from a machine.
- Plug-and-play functionality.
- Drive consistent end-user specification.
- Elimination of barriers presented by the control platform.
- A modular approach to machine control.
- Lower total cost of ownership.
- Integration of machines from multiple vendors.
- Support for productivity-enhancing initiatives such as overall equipment effectiveness (OEE)

PackML defines machine states, modes of operation, and machine-to-machine communication. It provides a broadly recognized machine state model and standardized data model (PackTags) to help ensure that programmers are speaking the same language and using terminology consistently. This creates a predictable, reusable model when multiple programmers are involved during machine design. The PackML states flowchart is shown in Figure 2.

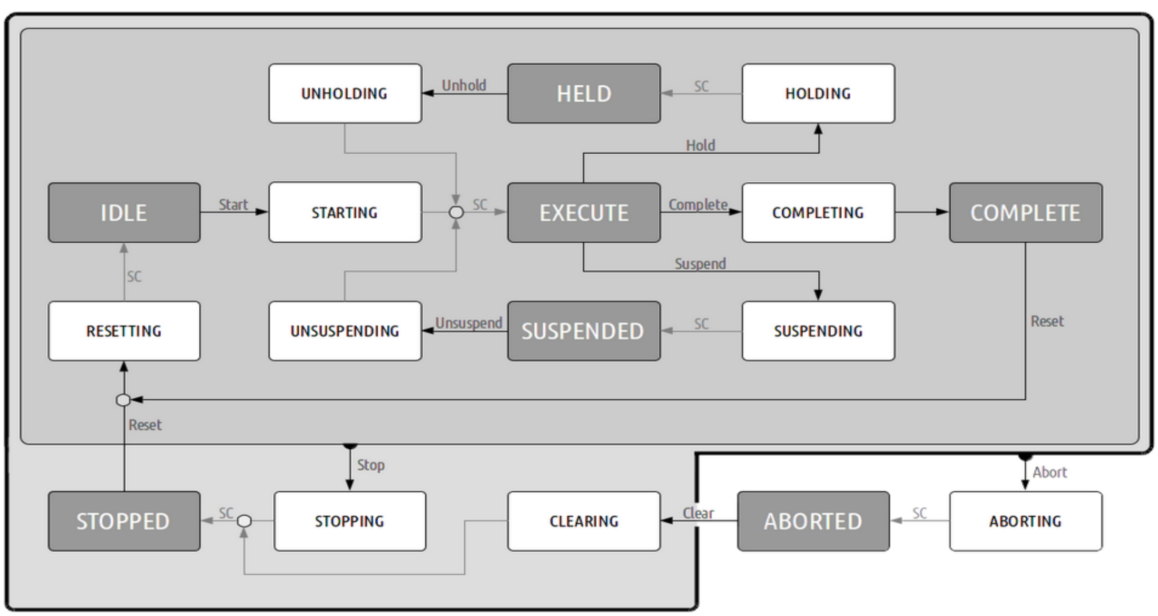


Figure 2: PackML State Model

Constraints to be considered when implementing PackML include:

- Programming skills and time.
- Memory space and limitations.
- HMI is required for each machine.

Context of the Exercises

The exercises contained herein are assigned as part of a senior-level course titled "Programmable Logic Controllers for Advanced Mechatronics Applications." This is a required course for the university's ABET-accredited Bachelor of Science in Mechatronics Engineering Technology program. The three-credit-hour class is currently limited to seventeen students with one section offered in the Spring semester. The class meets for five hours each week; a two-hour lecture at the beginning of the week and a three-hour lab in the latter part. A sophomore-level introductory course in PLCs is a prerequisite; both courses in PLCs use identical laboratory equipment, therefore allowing the students to refer to their reports in the second course.

Laboratory exercises are currently performed using Rockwell Automation products. The trainer chassis contains a ControlLogix 5570 controller with analog, digital, and relay input/output modules, a CompactLogix controller, EtherNet/IP-based local area network, variable frequency drives (VFD) and induction motors, servo drives and motors, and a 10.4" HMI. Rockwell's Studio 5000 software is used for programming; at present the curriculum covers ladder logic and structured text. FactoryTalk View Studio facilitates the development of the HMI.

Summary of the Exercises

Nine to ten laboratory exercises (labs) are assigned each semester, exclusive of a midterm and final project. The student will become This paper describes four exercises concerning PackML. The labs are organized to introduce and gradually develop an understanding of the PackML concepts. The outcome of completing these exercises is that the student will become familiar with state-based logic programming in concurrence with the development of a complimentary HMI runtime. The topics of the four PackML exercises are:

- Lab 1 – PackML Concept and Overview
- Lab 2 – PackML Trainer
- Lab 3 – Simplified PackML Cycle
- Lab 4 – Complete PackML HMI for 17 States

The basic format of the exercises provides background information concerning the topic. The information is then used to solve a stated problem. Discussion questions are posed to the student to evaluate comprehension of the topic. The student will develop a standardized report for each lab to create a reference document for their benefit; many students refer to their reports in other classes such as "Robotic System Integration" and their transition to industry.

Student Survey Response

The responses from students following the completion of the PackML project were overwhelmingly positive. Many students not only acknowledged the project's significance but also shared their newfound knowledge and experiences during job interviews. Remarkably, PackML was highlighted as a valuable skill, with employers expressing a preference for candidates with PackML experience.

A substantial majority of students demonstrated proficiency in implementing the PackML approach, reflecting the effectiveness of the project in imparting practical skills. The overall feedback from students has been exceptionally favorable, particularly when they integrated their PackML experience into their resumes. Students reported that this experience played a pivotal role in securing rewarding careers in the fields of packaging and machine integrations.

The positive outcomes of the PackML project, as reflected in the student's success stories, underscore its effectiveness in not only enhancing theoretical understanding but also in providing a tangible advantage in the competitive job market. The project's impact on students' career paths is a testament to the practical value and real-world applicability of the PackML approach."

Conclusion

In conclusion, this paper has extensively explored the PackML technique, providing a comprehensive overview, understanding, implementation, and evaluation of this approach. The incorporation of four detailed labs, elucidated step by step, serves as a valuable resource for instructors and faculty seeking to impart essential knowledge for implementing PackML. The labs, executed through Rockwell software, are versatile and can be adapted to different software platforms available in educational settings.

PackML presents numerous advantages to the packaging industry, such as enhanced operational efficiency, heightened flexibility, and cost reduction. Through the standardization of machine control, PackML streamlines the setup, operation, and maintenance of packaging machines, mitigating the risk of errors and inconsistencies in machine functionality. This standardized approach not only facilitates ease of use but also contributes to a clearer understanding for students.

The implementation of these labs plays a pivotal role in equipping students with both theoretical knowledge and hands-on experience in dealing with actual machine operations. Furthermore, it enables them to comprehend the intricacies of integrating production lines. These labs are instrumental in fostering a robust foundation for students to grasp the practical aspects of PackML and its application in the packaging industry.

Acknowledgment

The authors extend their sincere appreciation to OMAC with special recognition to Mr. Mike Hogan, the Sr. Director Global Quality and EHS and his dedicated OMAC team. We are profoundly grateful for their unwavering support, which has greatly benefited academia. PackML's commitment to providing numerous opportunities for educators and students, including workshops training, and grants, has made a significant and positive impact.

References

[1] M. G. Ishikawa, "What is PackML and how can it help packaging companies and OEMs?," *PackIOT*, Aug. 17, 2021. https://packiot.com/what-is-packml-and-how-can-it-help-packaging-companies-and-oems/

[2] A. Presher, "PackML with OPC UA solution for machine integration | Industrial Ethernet Book," *Industrial Ethernet Book*, Dec. 27, 2020. https://iebmedia.com/technology/industrial-ethernet/packml-with-opc-ua-solution-for-machine-integration/

[3] M. G. Ishikawa, "What is PackML and how can it help packaging companies and OEMs?," *PackIOT*, Aug. 17, 2021. https://packiot.com/what-is-packml-and-how-can-it-help-packaging-companies-and-oems/

[4] "Packaging Work Group." https://www.omac.org/packaging-work-group

[5] OMAC—Organization for Machine Automation & Control, "What is PackML?," *YouTube*, May 06, 2019. https://www.youtube.com/watch?v=CQS-R5COOLw

[6] Itxadmin, "Introduction to PackML: a Universal Machine Control Language," *E Tech Group*, Nov. 03, 2023. https://etechgroup.com/blog/manufacturing/packml-for-beginners/

Appendix Key

- Assigned Text: Normal
- Benchmark Response: *Italicized*

Appendix A – Lab #1

Lab #1

Introduction

PackML is an industrial standard for the control and operation of packaging machines. PackML stands for Packaging Machine Language; it is a roadmap for consistently developing equipment [1]. The core of PackML defines seventeen machine states (starting, executing, held, holding, resetting, etc.) that describe the operation of the machine. By deploying PackML to multiple, connected machines the flow of information becomes consistent throughout the installation. PackML provides benefits to all parties involved with the machine including operators, technicians, programmers, and managers [2]. These benefits include:

- A consistent look and feel for the entire production line.
- Inherent flexibility for various applications
- Simplified operation and maintenance
- Increased operational efficiency.
- Elimination of company-specific terminology

PackML is maintained by OMAC, the Organization for Machine Automation and Control. OMAC connects machine builders, machine users, and system integrators to develop and share best practices that improve industry efficiency while retaining their competitive advantages [3].

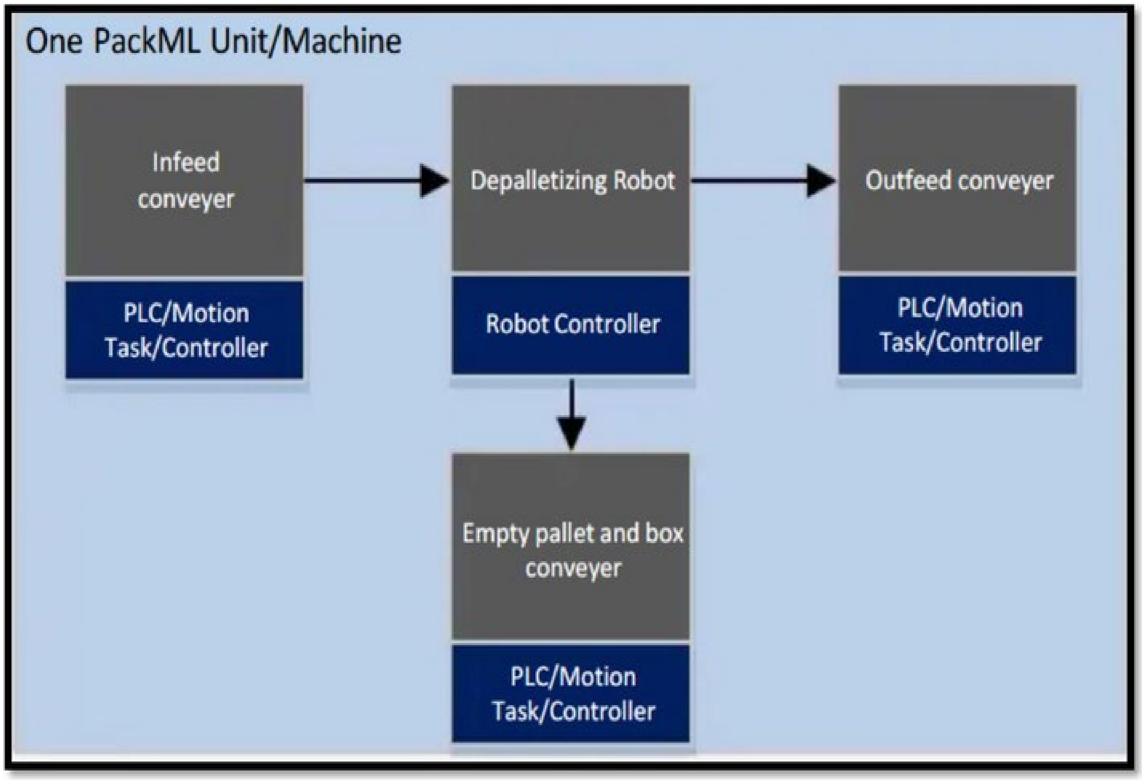


Figure 1. PackML Example [1]

Questions:

1. How can PackML be used to improve the scenario shown above?

2. What are the challenges you will face when you integrate this system?

Exercise Response

1.
    a. *The immediate observation from this example is that the system could utilize four different controllers (Rockwell, Siemens, Panasonic, and FANUC, for example.) This is not a bad thing per se. Still, for the sake of maintaining and repairing the machine, the workers must now be familiar with four different automation systems that are likely to use different names and conventions to describe their operation. (Wait vs WAIT vs WAITING vs WAIT_A could be describing the same overall function in the controller programs.) In a troubleshooting scenario, the technician must now learn to connect and translate terms which diverts their effort from repairing the system.*
    b. *The topography of the system demonstrates that the conveyor language is being translated for use by the robot, then re-translated and transmitted forward to the outfeed and empty container conveyors. There is no supervisory control in the diagram; there is no way to observe and evaluate the system at large. If we*

presume that there are multiple automation vendors, there will likely be four different HMIs in use which may confuse inexperienced operators.

c. *If PackML is implemented in this system an opportunity is provided wherein the language of operation can be synergized. PackTags can describe the same operation throughout the system (Starting, Executing, Completing, Completing) rather than the in-house standard for each programming provider.*

d. *The opportunity to introduce a supervisory system allows for greater insights into the process. This can include data keeping, data visualization, and a reasonable determination of operational efficiency.*

2.

a. *The ease of integrating this system will depend on how the controllers were originally programmed and whether the controllers can support the PackML standard. Many hardware vendors provide software packages for this purpose. There may not be packages for older or more obscure controllers so one will have to develop their setups and test them for compatibility to the standard.*

b. *It will also need to be determined whether the mechanical devices have the correct sensors to determine what state they are in or to collect data for supervisory use. It may be that a conveyor in this scenario is always moving and does not observe for part presence for instance.*

Questions

1. What is the concept of PackML?
   a. *Packaging Machine Language (PackML) is an automation standard intended to make machine data easier to transfer from machine to machine. PackML provides benefits to all parties making or using the machine.*
2. How does PackML bring a common "look and feel" between multiple machines?
   a. *PackML implementation is dependent on how information is expressed through HMIs. Ideal presentations clearly define what the individual machine is doing, but in a way that can be easily compared to what another machine is doing; waiting on machine 1 looks just like waiting on machine 2 and looks just like waiting on machine 3.*
3. What are the differences between machine states and machine modes in the context of PackML?
   a. *States: the status or condition of a machine in question*
   b. *Modes: the way that the machine is used*
4. What are the advantages and requirements for implementing PackML?
   a. *Advantages: PackML standardizes machine control. This reduces the amount of work done to maintain multiple systems. Additionally, the system provides stability in scalability; the size of the system is easily increased using PackML.*
   b. *PackML requires hardware and software that can communicate data (including PackTags) across different hardware and software installations. It also requires memory space for managing data. The people maintaining or programming the systems will need to be familiar with and understand PackML.*
5. In what way has PackML found applications beyond packaging, especially in discreet manufacturing?
   a. *Because PackML provides a way to standardize communication, it may be able to be used in instances where any industrial equipment needs to be integrated with*

other equipment that is not necessarily plug-and-play. For instance, when developing small batches, prototype production of items. In this instance, the goal is to achieve harmony in the system at all costs as soon as possible.

6. Can you provide examples of the type of data that can be transmitted from a machine to its supervisor?
    a. *PackTags (Machine Status)*
    b. *Commands/Recipe parameters (Temperature, Speeds)*
    c. *Current Operating Mode*
    d. *Alarms*
    e. *Machine Cycle Count*
7. What are some examples of data that can be exchanged between machine peers?
    a. *Machine Status*
    b. *Machine Mode*
    c. *Communication Status*
8. Besides packaging or producing products, what are some other intended uses for a packaging machine? Can you give examples of packaging machine modes?
    a. *It may be necessary for a machine to simply operate as a conveyor and no operation is performed on the object. In this instance, the machine mode would need to be defined as PASSTHROUGH or BYPASS. Execute would be redefined as "moving the item to the next conveyor stage" from "performing the usual operation."*
9. Why is it crucial to establish naming conventions for data within the PackML data interface?
    a. *Harmonizing naming conventions makes the system easier to approach regardless of job title. Within each business there exists unique terminology and ways of writing. When a stranger encounters these unfamiliar terms, time must be spent to translate and relearn, which detracts from their original goal of operating or repairing the machine.*
    b. *On the factory floor, a common language will assist interactions between operators, technicians, engineers, and managers. Some groups may not be as familiar with the machine as others or may be familiar in a different way (programming vs operating.)*
10. Are there any other conventions or requirements related to the data interface that should be considered in the context of PackML?
    a. *One should ensure that the units for given data (velocity, acceleration, etc.) are uniform across the integrated system. INT 1 = INT 1, but 1 [ft/s] does not equal 1 [m/s]*

Conclusion

It appears that PackML provides a clear method for developing control systems for many, if not all manufacturing environments. Standardization is appealing in a world that is moving towards larger, totally connected systems; PackML simplifies otherwise com.

Appendix B – Lab #2

Lab #2

Introduction

The PackML Trainer allows the user to observe the PackML states as they are manipulated. The trainer was developed to provide hands-on experience with PackML without the bulk of an actual machine. It shows all 17 states and the transitions between them through animation [4]. The trainer has multiple modes to simulate various levels of control over the system (Base and OEM) [5].

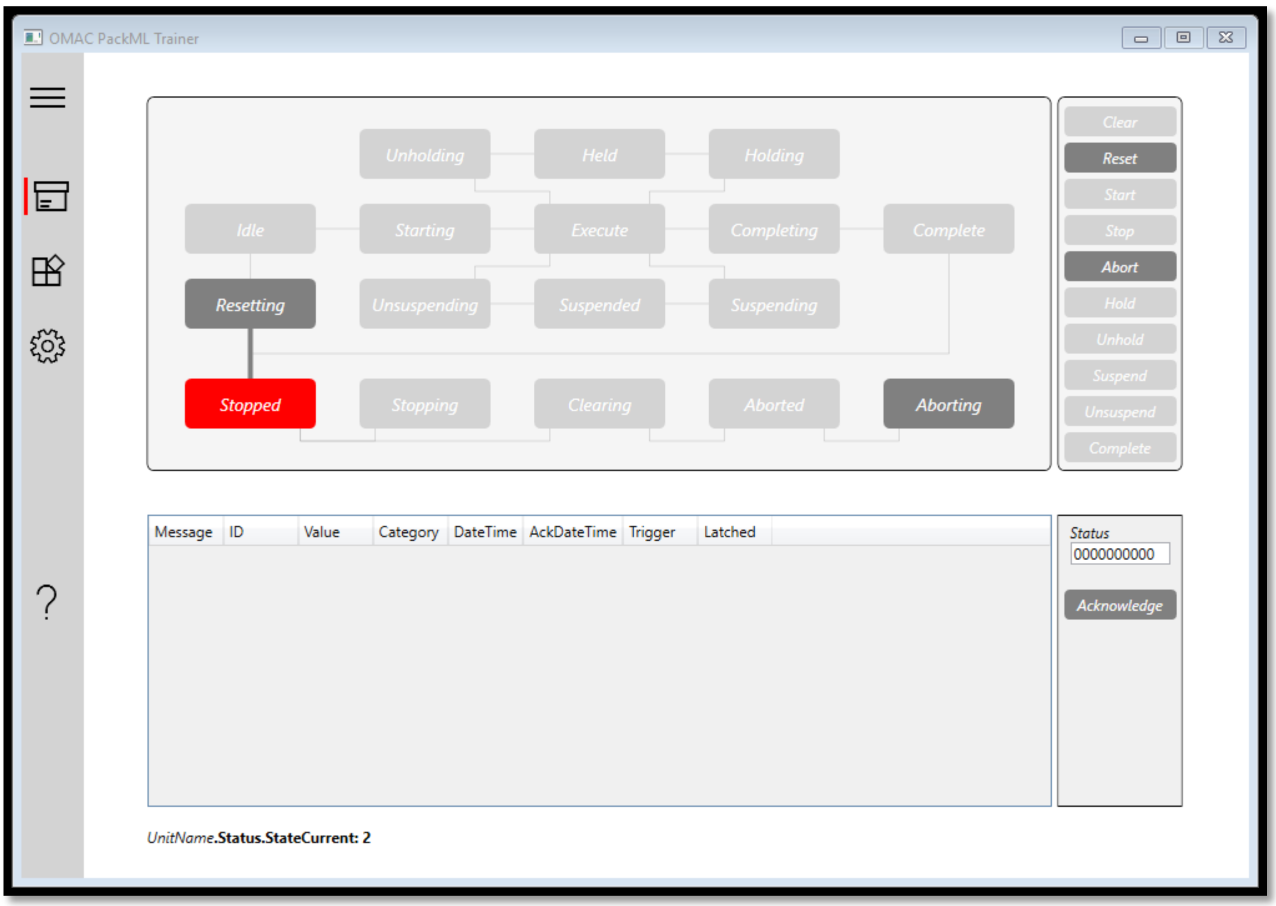


Figure 1 PackML Simulation Trainer on Startup [6]

Lab Activity

Figure 1 shows the trainer as it starts up in the “Simple PackML Workflow” mode. The only options are to either reset or abort. Aborting leads to Clearing > Stopping > Stopped as shown in Figure 2.

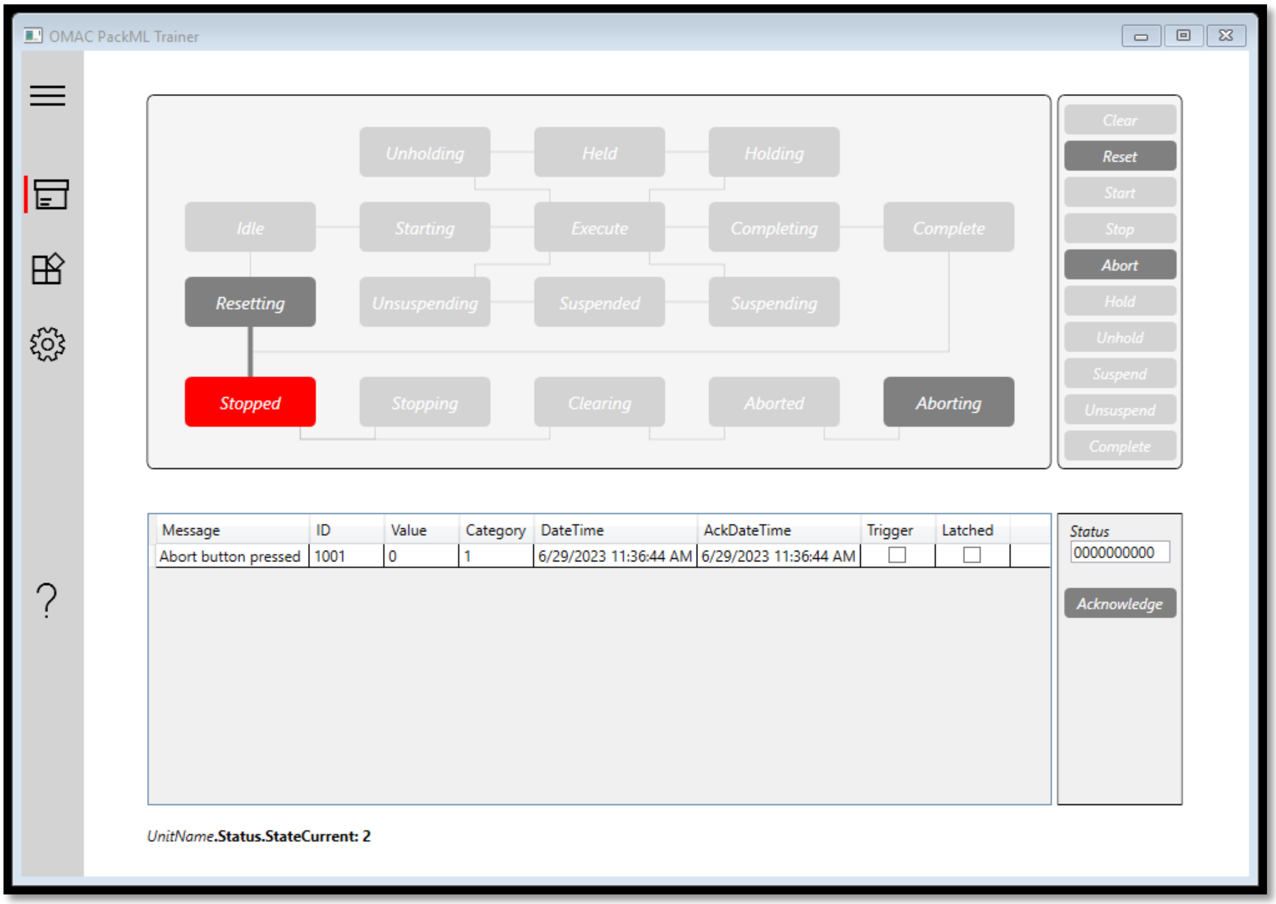


Figure 2. PackML Simulation Trainer on Stopped

After resetting, the system enters Idle where it waits for Start, Stop, or Abort commands in Figure 3.

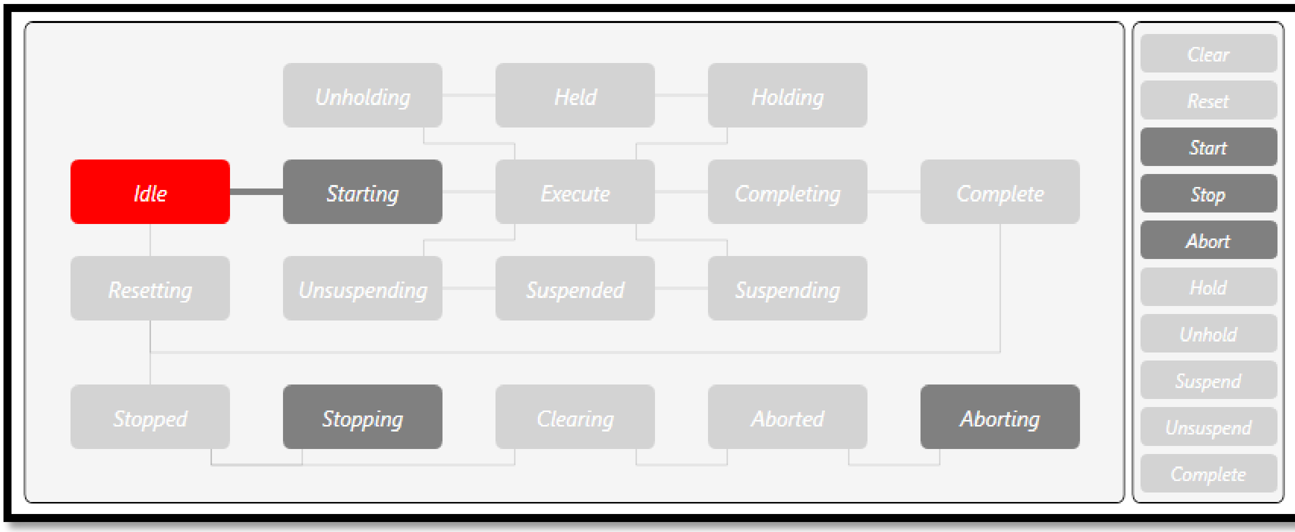


Figure 3. PackML Simulation Trainer on Startup

If the user switches to the OEM/Typical User Experience mode, there is an immediate change in the available inputs. Figure 4 is at the same stage as Figure 3, but the Abort button has been replaced/integrated into the Reset button.

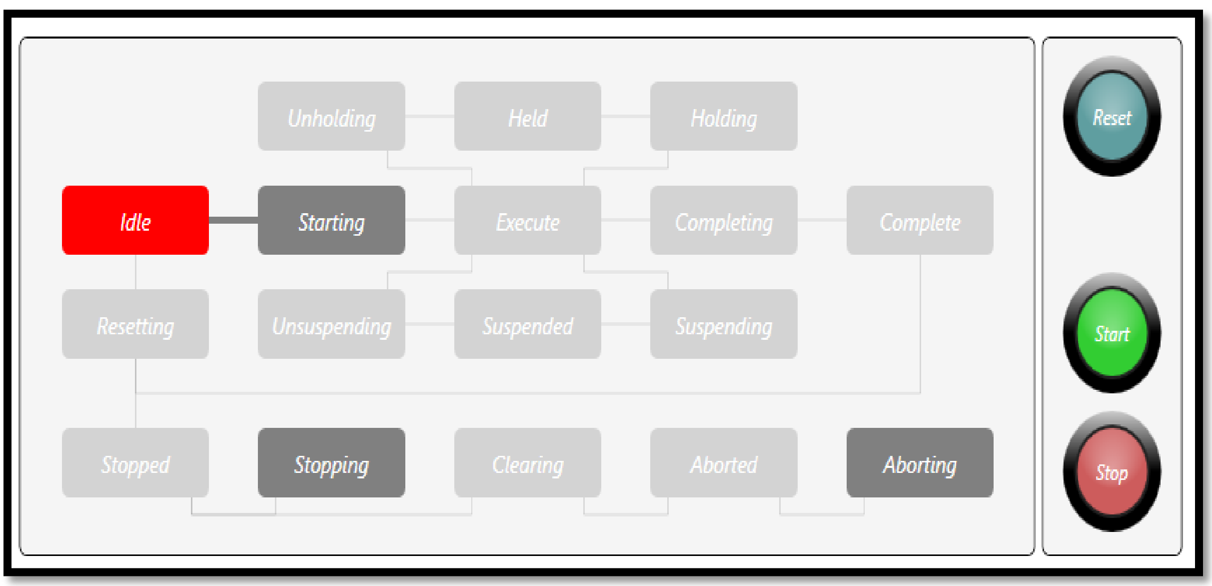
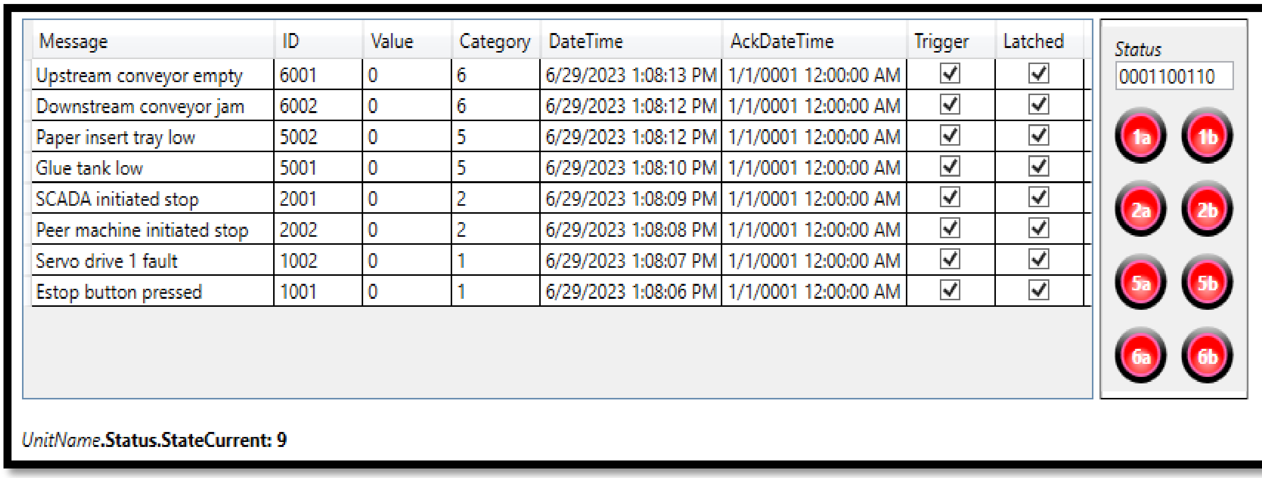


Figure 4. PackML Simulation Trainer on Idling in OEM

OEM limits the interface to what one may expect to see on a production machine (Start, Stop, Reset). It is important to understand that the workflow mode is representative of what is occurring inside the controller, whereas the OEM mode represents what an operator without a programming terminal will see. It is useful to anticipate what other people's interactions with the machine will look like.



Message	ID	Value	Category	DateTime	AckDateTime	Trigger	Latched
Upstream conveyor empty	6001	0	6	6/29/2023 1:08:13 PM	1/1/0001 12:00:00 AM	☑	☑
Downstream conveyor jam	6002	0	6	6/29/2023 1:08:12 PM	1/1/0001 12:00:00 AM	☑	☑
Paper insert tray low	5002	0	5	6/29/2023 1:08:12 PM	1/1/0001 12:00:00 AM	☑	☑
Glue tank low	5001	0	5	6/29/2023 1:08:10 PM	1/1/0001 12:00:00 AM	☑	☑
SCADA initiated stop	2001	0	2	6/29/2023 1:08:09 PM	1/1/0001 12:00:00 AM	☑	☑
Peer machine initiated stop	2002	0	2	6/29/2023 1:08:08 PM	1/1/0001 12:00:00 AM	☑	☑
Servo drive 1 fault	1002	0	1	6/29/2023 1:08:07 PM	1/1/0001 12:00:00 AM	☑	☑
Estop button pressed	1001	0	1	6/29/2023 1:08:06 PM	1/1/0001 12:00:00 AM	☑	☑

Figure 5. OEM Mode Alarms/Massages

The OEM mode also allows the use of eight errors/alarms/messages. These can be triggered to demonstrate how the system reacts to them. When the red button corresponding to each error is lit the system cannot clear the error. The red button controls the trigger function, whereas the latched function must be unchecked in the check box. This represents the operator physically clearing the obstruction or issue.

Questions

1. What is the difference between mode and state?
    a. *State: seventeen states describe what the machine is doing at any given time.*
    b. *State command: the action that triggers a change from one state to another e.g., a button press.*
    c. *Mode: a set of allowed states for a given process e.g., Production, Cleaning, Calibration*
2. Describe some machine modes.
    a. *Production: all actuators and processes are utilized to make products or perform operations*
    b. *Calibration: does not move products through, merely tests the operations of the machine so that an operator or technician can measure something about them*
    c. *Maintenance: does not produce product, may move pieces of the machine away for ease of access*
    d. *Bypass: moves product through the machine without operating on the products themselves*
3. Held vs Suspended.
    a. *Held: transition from this state occurs based on the change of internal conditions or by operator input*
    b. *Suspended: transition from this state usually occurs based on the change of external conditions usually without operator intervention. Relevant process parameters e.g., %completion must be retained.*
4. Why are some states "out of the border" like aborting and stopping?

    a. *Stopping and Aborting (a severe fault or Estop) exist outside the normal running routine of the machine. They will only occur when a Stop or Abort state change occurs through the intervention of external forces.*

5. Internal vs External Conditions
    a. *Internal: Conditions that require immediate operator intervention like Low Fluid, Supply Empty. These conditions exist in the context of one machine.*
    b. *External: Conditions that exist upstream or downstream of one machine. A downstream blockage or lack of components in the upstream conveyor would be considered external. They do not originate within the machine.*
6. What actions can be taken if a machine spends lots of time in a Held state?
    a. *Held states originate from internal conditions i.e., conditions that the operator is responsible for preventing or alleviating. The specific causes need to be evaluated in real-time production and evidence collected. It may be necessary that additional operators are required to run the process, that more supervision is needed, or that improvements be made to the quantity of available materials (fluids, placed components, etc.) It is unwise to simply blame the operator and ask them to work faster.*

Conclusion

The PackML trainer is a well-made tool for demonstrating the workflow of a PackML machine. It is helpful to understand the potential reaction of an input to the system. That said, it is difficult to visualize how the physical machine operates going off this trainer alone. It would be helpful if an animation of a machine process were displayed alongside the flowchart.

Appendix C – Lab #3

Lab #3

Create and design a simple PackML on the HMI screen.

<u>Input: Push Buttons</u>

1. Start – to start the process.
2. Stop – to stop the process.
3. Reset – to reset a fault before restarting the system.
4. Infeed Jam – this simulates a jam on the infeed side of the machine.
5. Outfeed Jam – this simulates a jam on the outfeed side of the machine.
6. Internal Failure – this simulates a failure (electrical or mechanical) within the machine.

<u>Output:</u>

1. Running – indicates the process is running properly.
2. Idle – indicates that the process is not running.
3. Held – indicates that the machine is on hold due to an infeed/outfeed jam.
4. Suspended – indicates that the process is suspended due to machine failure.

<u>Transition state:</u>

They stay on only for a certain time showing the transition in between the input and output.

1. Starting
2. Stopping
3. Resetting
4. Holding
5. Suspending

Create a ladder logic using Studio 5000 and the above buttons on the HMI screen to simulate the scenario discussed using the hint below in Figure 1.

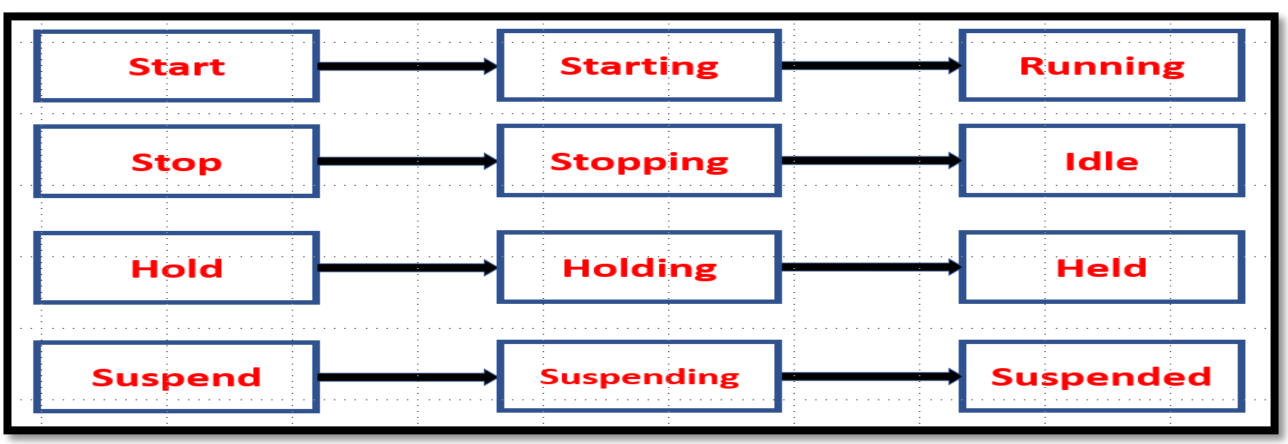


Figure 1. Flowchart of the State Transitions

1. The "START" option is only available after "RESET" is pressed and the system finishes resetting – use the button visibility option in FactoryTalk View.

2. "RESET" should clear the "HELD" and "SUSPENDED" conditions.

Appendix D – Lab #4

Lab # 4

The objectives of this lab are to:

- Develop a PackML Trainer with 17 states using Studio 5000 software.
- Design an HMI to show the state transitions.
- Test and troubleshoot the system.
- Develop the PLC program with your choice of PLC language such as Ladder logic or Structure Test, etc.

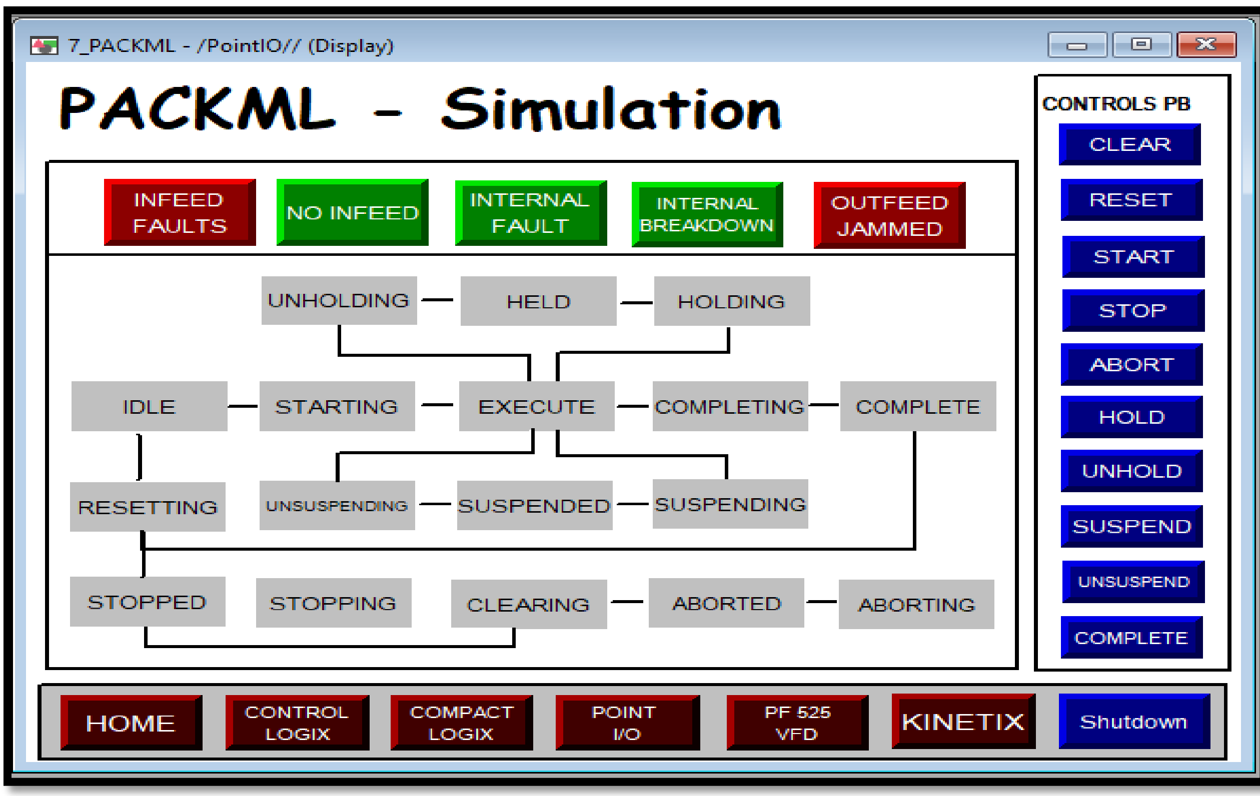
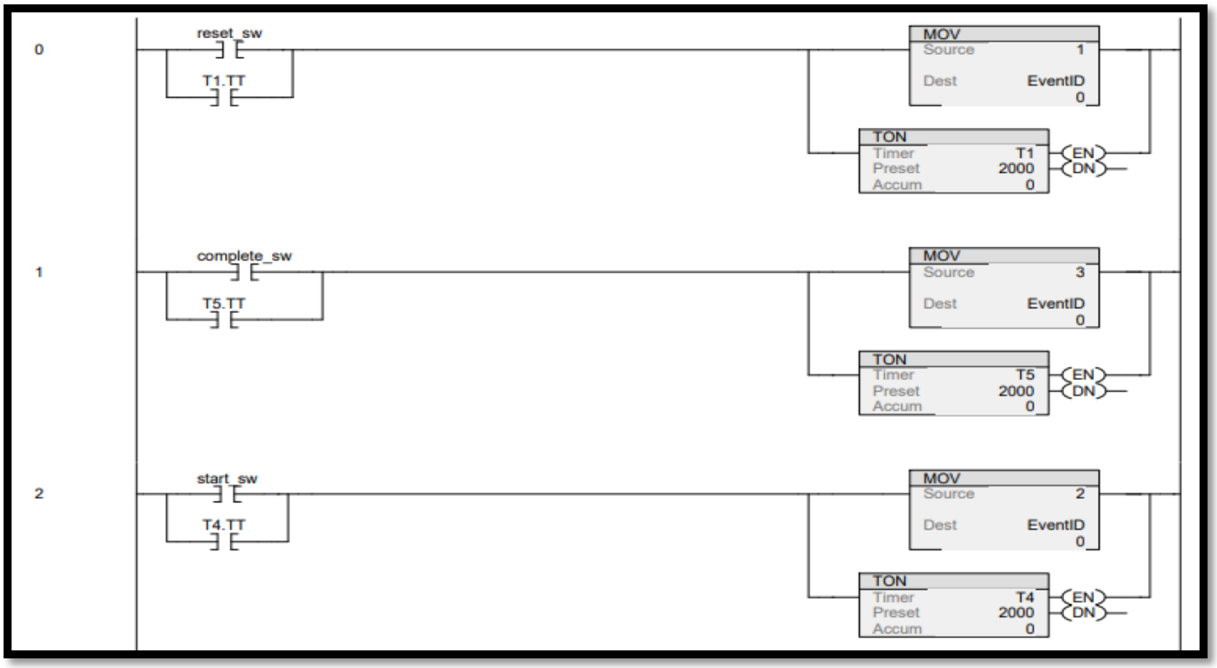


Figure 1 PackML Full (17) States




```
Held :=0;
Unholding :=0;
Suspended :=0;
Unsuspending :=0;

case EventID of

      1:
          //reset
          if T1.TT then
              resetting :=2;
              idle :=1;
              stopping :=1;
              aborting :=1;
          else
              resetting :=0;
              idle :=2;
              stopping :=1;
              aborting :=1;
              starting :=1;
          end_if;

      2:
          //start
          if T4.TT then
              starting :=2;
              executing :=1;
              stopping :=1;
              aborting :=1;
          else
              starting :=0;
              executing :=2;
              holding :=1;
              completing :=1;
              suspending :=1;
              stopping :=1;
              aborting :=1;
          end_if;
```

```
          else
              suspending :=0;
              Suspended :=2;
              Unsuspending :=1;
              aborting :=1;
              stopping :=1;
          end_if;

      9:
          //unsuspend
          if T10.TT then
              Unsuspending :=2;
              executing :=1;
              stopping :=1;
              aborting :=1;
          else
              Unsuspending :=0;
              executing :=2;
              holding :=1;
              completing :=1;
              suspending :=1;
              aborting :=1;
              stopping :=1;
          end_if;

      10:
          //clear
          if T11.TT then
              clearing :=2;
              stopped :=1;
              aborting :=1;
          else
              clearing :=0;
              stopped :=2;
              resetting :=1;
              aborting :=1;
          end_if;

end_case;
```

Figure 2 Sample Code in Ladder Logic and Structured Text