Michel Baudin's Blog

Ideas from manufacturing operations

FEATURED

Yet Another Post About Poka-Yoke

A week ago, José Roberto Rolim Nunes started a new discussion on the TPS Principles and Practice discussion group on LinkedIn by asking "What is Poka-Yoke?" As of today, it has had 42 contributions, including several from Sid Joynson, Jerry O'Dwyer, and Peter Winton. Sid recounts personal communications from Shigeo Shingo, Jerry adds a semiconductor industry perspective, and Peter discusses Poka-Yoke with current production machinery versus what it was in Shingo's days. Discussion groups often revisit the same topics, but with different participants exploring different details. On Poka-Yoke, I have posted the following before:

1. Key details on Poka-Yoke/Mistake-Proofing (11/22/2011). A compilation of my inputs to a discussion of Poka-Yoke in the AME discussion group on LinkedIn. Several sections of this post are repeated here, with illustrations added.
2. Poka-Yoke at Toyota: the Current State (3/10/2013). A review of Mikiharu Aoki's 2013 book on the subject.

The featured image above is of my favorite Poka-Yoke. The press punches a hole in a fiberboard with no metal parts. At a later step, a metal bracket is mounted in this hole. One day, an operator mistakenly loaded into the press a board that already had a bracket in it, which put the press die out of commission for 36 hours. The device mounted in front of the press is a permanent magnet, which sucks up any board already containing a bracket.and preventing it from going in. It was designed and implemented by Hormoz Mogarei's team a few years ago.

The topics covered here are as follows:

Definition, applicability and boundaries

A Poka-yoke is a device integrated in a manufacturing operation to prevent *human* error. As an approach to quality improvement, it is therefore relevant where human error is the main cause of defects, which means that your process is capable and that discrete malfunctions, such as tool breakage, either stop the line automatically or are promptly detected thanks to one-piece flow.

Once you have a capable process and one-piece flow, human error percolates to the top of the Pareto chart of defect causes, and Poka-Yoke can take you to the next level. One feature of Poka-Yoke that is often missing in the literature is that the defect-prevention devices must not add labor.

The reason this is vital is that, otherwise, operators will stop using them under pressure and they will be ineffective. For example, validating picks by reading bar codes adds labor if an operator has to wave a reader and is therefore not a Poka-Yoke. If the ID Is read automatically while the part travels on a belt, then it is a Poka-Yoke.

Poka-Yoke and Inspections

A common misconception is that only devices that prevent defects being created qualify as Poka-Yoke. Poka-Yoke inventor Shigeo Shingo, however, disagrees. To Shigeo Shingo, “inspection” was not a dirty word. In ZQC, p.92, he wrote: “Poka-Yoke systems involve carrying out 100% inspections and requiring immediate feedback and action when errors or defects occur.”

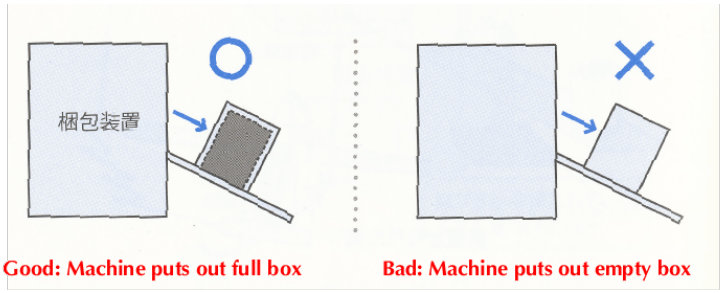
Although he does not say this explicitly, what I read between his lines is that, since Poka-Yoke exist to prevent human error, they do not prevent defects from being created by machines, but they ensure that these defects are detected and acted upon immediately when the workpiece is unloaded.

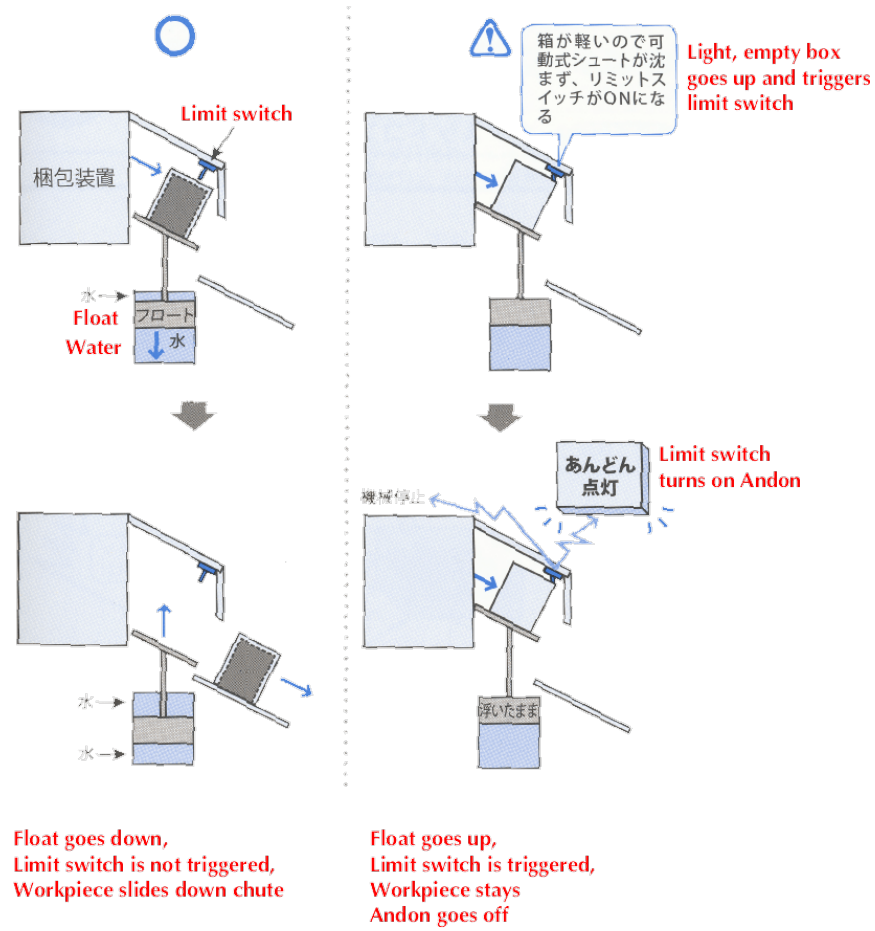
We should remember that Poka-Yoke is about prevention of *human* error, not *machine malfunction*. Even in the best processes, machines will occasionally malfunction, in ways that will damage a workpiece and *preventing* this from happening is out of the scope of Poka-Yoke.

On the other hand, it is a *human* mistake to let this defective part escape and proceed to the next operation, and a device that keeps this from happening performs mistake-proofing and is therefore a Poka-Yoke, as Shingo defined it. It should also be noted that Lean/TPS includes other forms of 100% inspections as well, such as go/no-go gauge checking on every part, or successive inspection, in which each operator on an assembly line starts by touching every component that was supposed to be installed at the previous station.

Nobody likes inspections, but, if they find defects, they are necessary. An inspection step that fails to uncover a single defect in a million consecutive units is probably unnecessary. When you find that 100% inspection is unnecessary, you don’t replace it with sample inspection, because drawing samples would disrupt the production routine. Instead, you go to first and last piece checking on a production run. And the next step is complete elimination of the inspection step.

An inspection system that just filters bad parts is not a Poka-Yoke, because it does nothing to prevent the generation of more bad parts. If you have a process that is out of control and routinely produces 5% of defectives or more, human error is *not* the problem, and Poka-Yoke *not* the solution. You need to fix the process. The following is an example of a Poka-Yoke to detect empty packages coming out of a machine, from Mikiharu Aoki’s Poka-Yoke at Toyota: the Current State (pp. 168-169):





Poka-Yoke example from Mikiharu Aoki

It does not prevent the machine from putting out an empty box, but it prevents it going forward. This Poka-Yoke could have been built in the 1960s. For this purpose, a engineer's first instinct today would have been to use an electronic check-weigher, and program it to trigger the Andon as needed.

This mechanism in this example can be implemented by a shop floor team with mostly mechanical skills. An even simpler device that has been used to filter empty boxes on a conveyor is to blast air at them: the empties fly off, while the full ones are unaffected; it does not, however, trigger the andon.

Poka-Yoke and Statistics

About Poka-Yoke versus Statistical Methods, following are two quotes from Shigeo Shingo's ZQC book and Mikel Harry's Six Sigma, that I posted a few days ago in another thread on the TPS+1 subgroup:

Shigeo Shingo: "When I first heard about statistics in 1951, I firmly believed it to be the best technique around, and it took me 26 years to be completely free of its spell." ZQC, p. 54

Mikel Harry:"We believe that statistical knowledge is to the information age what fossil fuel was to the industrial age. In fact, the future of industry depends on an understanding of statistics." Six Sigma, p. 24

Shigeo Shingo's and Mikel Harry's perspectives seem diametrically opposed. Neither Shingo nor Harry, however, takes the trouble to specify the context of their remarks. Shingo's world is primarily automotive; Harry's, electronics. My take on this is that they are both right, in non-overlapping universes.

My first experience in manufacturing was in the semiconductor industry, where lack of process capability, at the level of the whole 500+ step wafer process, is the main cause of defects, and is fought by armies of yield enhancement engineers who use statistical design of experiments. In this context statistical tools are indispensable. Later, working in automotive, it was not the case.

When I last looked at this, the semiconductor industry, along with pharmaceuticals, was the largest industrial user of statistical software products. The motivation in pharmaceuticals is compliance with legal mandates for drug approval; the motivation in semiconductors is internal, technical needs.

Poka-Yoke and Usability Engineering

Usability engineering is also an approach that, while falling short of Poka-Yoke, goes a long way towards reducing human errors by leveraging intuition and pre-existing habits. The idea is to make the human interfaces of tools and machines so intuitive that people naturally tend to use them correctly. It does not mistake-proof processes, but it makes mistakes unlikely and reduces training needs. The following picture shows features of the alarm clock I used before switching to a smartphone app. It included both usability engineering and mistake-proofing.

Swiss Army alarm clock

Usability engineering is applied to the clock face. It is analog, and consistent with the style developed over hundreds of years, that is now a cultural constraint, even for digital clocks. The hours are marked in bold numbers with high contrast and a consistent orientation. The hour and minute hands have markedly different lengths, and the hour-hand has an arrow. These hands will not be confused, and they glow in the dark.

The alarm and clock time settings, on the other hand, are mistake-proof. On traditional alarm clocks, you set the alarm time by turning a button on the back while looking at the clock face in the front, and often turn the wrong button, changing the clock time instead. The mistake-proofing device here is a cover that you have to open if you want to access the clock time knob. With the cover closed, you can only change the alarm time.

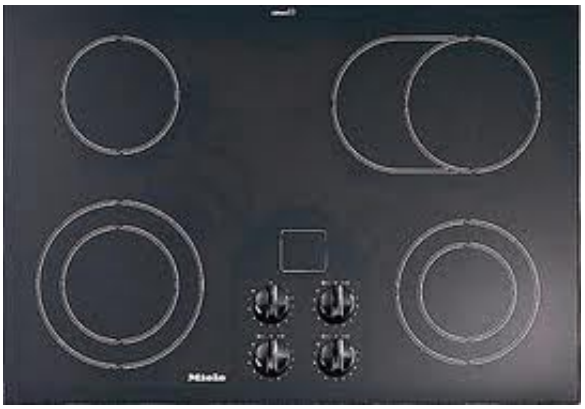
Mistake-proofing and usability engineering are two different disciplines, with overlapping goals. By making user interfaces intuitive, usability engineering makes mistakes unlikely but does not prevent them. Conversely, you could have a fully mistake-proof machine that would require you to study a manual in order to operate it.

In a car, for example, you could prevent starting the engine while in gear — which is mistake-proofing — and still make the controls unintelligible by confusing captions or unusual locations. In the Ford Edsel, for example, you shifted gears by pressing a button in the center of the steering wheel, which violated a cultural constraint on shifter location established over 70 years of car making.

The push-button shifter on the Ford Edsel

The reason you can rent a car from any brand and drive it off without opening a manual is usability engineering, not mistake-proofing. Usability engineering is widely used in consumer goods, which are bought their end-users, but not in production machines, which aren't. The principles of Lean equipment design include usability engineering, even though it is not explicitly referenced. Production machinery should have both; most of it currently has neither.

The best source on usability engineering is Don Norman's Design of Everyday Things. For its application to airliner cockpits, see Asaf Degani's Taming HAL. The application of usability engineering to production machinery is discussed in Part I of Working with Machines (pp. 9-84). In is also discussed in an earlier post on Avoiding Lean Wallpaper. Here is an example of Don Norman's recommendations for a stovetop design:



Natural mapping of knobs to burners

The relative positions of the knobs match the relative positions of the burners they control. It is not a Poka-Yoke, because it doesn't *prevent* you turning the wrong knob; in practice, however, it is the only layout in which nobody does.

Poka-Yoke in Production versus Household Appliances

Why is production machinery so lacking in mistake-proofing and usability engineering? I think the suppliers are just catering to their customers, the people who select equipment and make purchasing decisions. In most companies, the operators are out of this loop, and their needs are not addressed. Household appliances are loaded with mistake-proofing and usability engineering features because they are bought by their end-users.

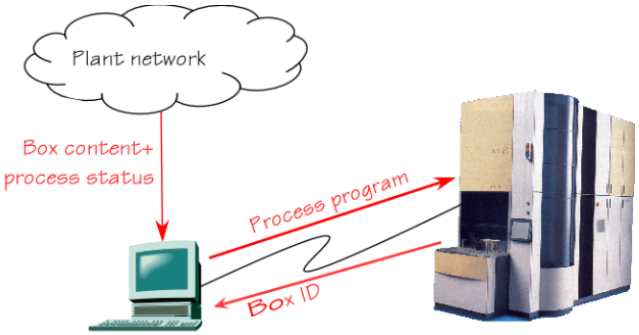
As a consequence, manufacturing Poka-Yoke are usually retrofits introduced as part of continuous improvement. To the extent that they are specific to the plant's use of the machines, this is inevitable. Machine suppliers, however, can help by building in Poka-Yokes that apply to all uses of their equipment, and by making it easy for users to add custom Poka-Yokes.

Poka-Yoke, Jidoka, and Technology

Logically, Poka-Yoke belongs under the Jidoka column of the TPS house. There are only two columns, the other one is Just-In-Time, and Poka-Yoke is not part of Just-In-Time…

The engineers who program control systems to prevent using the wrong recipe in semiconductor wafer processing or car engines from starting while in gear are definitely mistake-proofing. What the literature has encouraged us to do, however, is to think of Poka-Yoke as small, cheap devices developed by the people who do the work as part of continuous improvement. I don't see anything wrong with applying the term to the work of design engineers, as long as we don't forget about the Poka-Yoke from the floor.

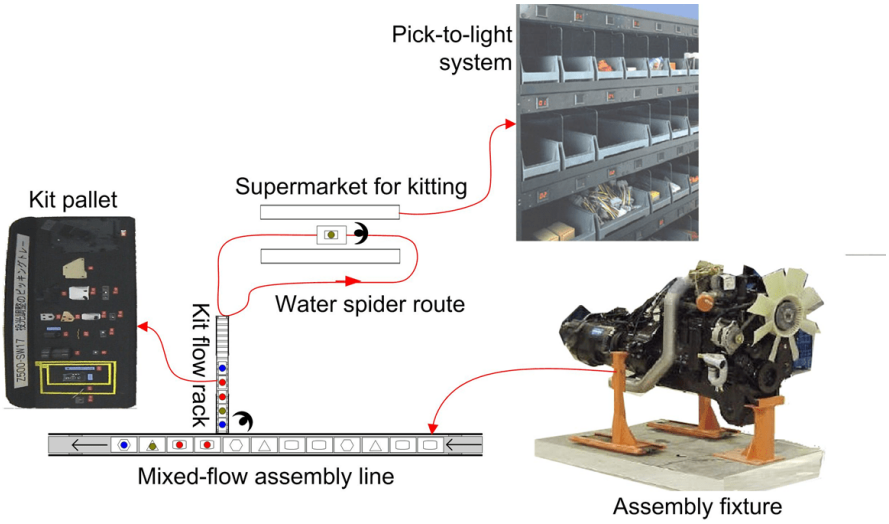
The Poka-Yoke concept relevant in semiconductor manufacturing even though process capability issues are dominant, because operator errors have catastrophic consequences. An operator who starts the wrong recipe on a diffusion furnaces may lose an entire load of 200 wafers, worth about $250K. The prevention of such errors is mistake-proofing, but it is pursued through high-technology methods by engineers specialized in computer systems to control process equipment. The following is an illustration of how this was accomplished in the 1990s:



The control systems embedded in the production machines were themselves powerful computers, and communicated with outside computers using industry-specific protocols. Using commercially available packages, in-house computer engineers programmed an external computer to communicate with both the embedded controller on one side and with the plant-wide Manufacturing Execution System to prevent the wrong process program being executed.

The literature on Poka-Yoke gives the impression that Poka-Yoke devices are always low-tech, but that is because they use examples from the 1960s. Relying on size differences between products to make them trigger or not trigger switches is fine as long as these differences exist. If, however, you are building differently configured computers in the same cases, the outer dimensions are identical, but you can use RFID tags to achieve the same result.

Imagine a mixed-flow assembly line as in the following picture:

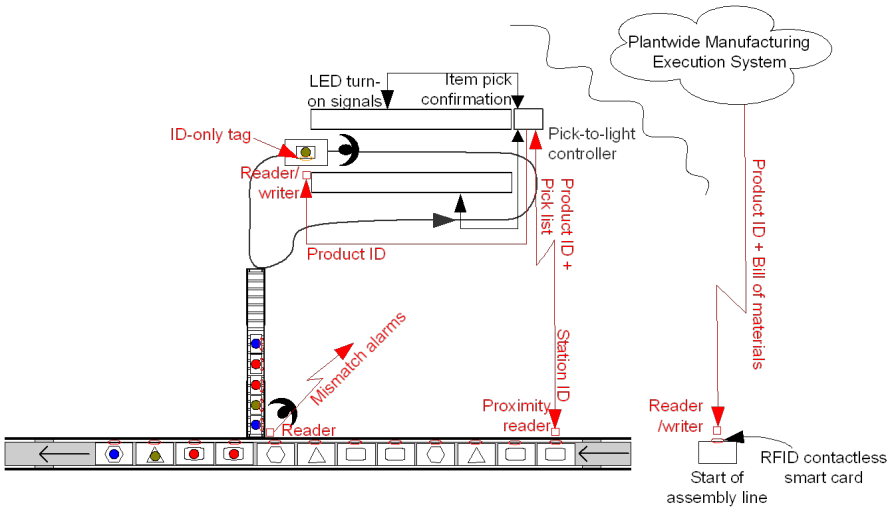


Mixed-flow assembly line

This picture focuses on one assembly station within a mixed-flow assembly line. The line makes a variety of different configured products. At a supermarket just off the line, a water spider picks kits of parts for the work done at this station and sequences them on a gravity flow rack a few minutes before they are used.

The pick-to-light system itself is not a Poka-Yoke, because it does not physically prevent picking the wrong parts or the wrong quantities. Systems that pop up lids to make parts accessible only in some bins are closer, because they only make the right parts accessible, but they still don't control the picking quantity. Pick-to-Light systems are popular because they are cheaper than automatic dispensers, increase picker productivity, and reduce, if not eliminate, picking errors. But the pick-to-light controller has to know which bins to light up.

How do you go about making sure it does, and the water spider delivers the kits in the proper sequence to the flow rack, knowing that you cannot rely on the outer dimensions of the product to pinpoint its configuration? Following is a possible solution based on RFID technology:



Mistake-proofing configuration with RFID

The entire bill of materials for the product configuration is loaded onto a high-capacity RFID tag attached the fixture at the start of the line. Past that point, all the data is locally held, and the operation of the line is decoupled from the central information system of the plant. An RFID proximity reader is located a few stations ahead. Through it, the product sequence and pick lists are fed to the pick-to-light controller just early enough for the water spider to pick the kits in time for assembly.

Poka-Yoke Implementation

When a Poka-Yoke is a simple device, as described in Shingo's ZQC book, in Productivity Press's big red book

of 240 examples, in Hinckley's more recent "Make no mistake" (2001), or in Aoki's "Poka-Yoke at Toyota (2012), it is implemented differently from when it is a larger-scale project. For these classical Poka-Yoke, the challenge is the idea, and it usually comes from a production operator or a technician.

The implementation cost is petty cash, and it takes a few hours to do, as part of continuous improvement. It does not require a formal economic justification or business case. The barrier is low. If you kind of think it the current setup is a mistake waiting to happen, and a $50 change to the operator work station will prevent it, you do it.

Preventing a diffusion furnace in semiconductors from running the wrong recipe is a different story. With the technology as it was when I was involved, you were talking about a $50K retrofit to the local control system, with integration to the overall plant system, so you had to look at the economics. In one case, there had been recipe mistakes about once a quarter, resulting each time in a total loss do a load of wafers, worth at this stage about $250K, which worked out to about $1M/year. Investing $50K to save $1M/years sounds like a good investment, but, for this kind of Poka-Yoke, you need to have that conversation, and it needs to be documented.

Related

Key details on Poka-Yoke/Mistake-Proofing
November 22, 2011
In "Technology"

What Poka-Yoke Are And Are Not, And How To Sustain Them
November 30, 2015
In "Technology"

Poka-Yoke at Toyota: the Current State
March 10, 2013
In "Book reviews"

August 8, 2013

11 Replies

« Previous

Next »

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Website

Post Comment

Notify me of follow-up comments by email.

Notify me of new posts by email.

Kent Vincent on August 9, 2013 at 7:20 am

I like seeing the jidoka concept brought into the discussion. On a client's engine parts line I worked there was one error proofing step involving a vision system to detect imperfections in "staking" a rocker arm axle in a valve train by way of an automated load cell mallet action on the end of the axle where softer malleable metal was positioned. Suspicious looking end-on views of the axle caused the rocker arm assembly to be diverted to an inspection and rework path to guard against loose axles that would release needle bearings into an engine cyclider resulting in a roadside breakdown. Further down the production line another vision system detected missing bolts in the final assembly that an operator failed to put in place prior to automated tightening. It could be argued that the latter step had more poke yoke elements to it than the first step which drew on "autonomation" concepts and not much in the way of operator error. The important thing is that both measures fall squarely in the jidoka "pillar" as it were and do not need to be caught up in the semantics of whether or not poke yoke is the terminology to be applied. The broader nomenclature also avoids any "search for the guilty human" aspects to any failures that occur. There are many cases where "poke yoke projects" ought to be labeled "jidoka projects" to keep overall objectives clear, relegating poke yoke to the status of one of possibly several implementation approaches.

↪ Reply

John Hunter on August 10, 2013 at 6:13 pm

For my own thinking I think of “mistake proofing” as best and different from “mistake making less easy” (visual indications of a problem for example, but no physical block to making the mistake). And I don’t think of “mistake proofing” as different for person v. a machine.

But in communicating with others I have to be much more verbose as the understanding of what poka-yoke means doesn’t fit the understanding I have in my head. Making mistakes harder and making them more visible is good. Preventing them is even better.

Reply

Michel Baudin on August 11, 2013 at 6:14 am

I agree that mistake-*proofing* — or Poka-Yoke — is different from making mistakes less likely. Mistake-proofing is primarily about improving quality, but it has the side effect of reducing stress for operators. If the work you are doing is mistake-proof, you have less to worry about.

Usability engineering is not a substitute for mistake-proofing but a complement to it. By making mistakes less likely it also improves quality, but the primary goal is to make processes easy to learn and execute. In routine operations, it reduces training and documentation costs, and increases productivity.

In emergency response, it saves lives. When an airliner pilot has to assume manual control of the aircraft, you cannot, and should not, constrain possible actions, as mistake-proofing would. But the cockpit’s usability is critical to the correct interpretation of the information it provides, and therefore to the outcome.

Mistakes are human; machines fail in other ways. They wear down and break, but they are not distracted or driven by habits. Preventing mistakes by operators is what Poka-Yoke is about, both etymologically and practically. Plenty of Poka-Yoke are about human-machine interfaces, but the control of what happens *inside* machines is outside the scope of Poka-Yoke.

Reply

Juergen Boenisch Ph.D. on August 22, 2013 at 10:03 am

Comment in the *TPS Principles and Practice* discussion group on LinkedIn:

An inspection system after a machine is not a Poka Yoke!

You cannot inspect quality into a system (Deming). It already happened and is after the fact.

Poka Yoke devices eliminate the root cause for making errors at the fist place. Hence, they need to be installed at the root cause of problems or defect.

Is it always possible to install Poka Yoke?
Well, most likely not, but it at least makes you think about the root cause and how to eliminate/minimize defects.
And if the people think hard enough about a problem, ‘Yes’, you will find a Poke Yoke device.

Reply

Michel Baudin on August 22, 2013 at 10:05 am

To Shigeo Shingo, “inspection” was not a dirty word. In ZQC, p.92, he wrote: “Poka-Yoke systems involve carrying out 100% inspections and requiring immediate feedback and action when errors or defects occur.”

Although he does not say this explicitly, what I read between his lines is that, since Poka-Yoke exist to prevent human error, they do not prevent defects from being created by machines, but they ensure that these defects are detected and acted upon immediately when the workpiece is unloaded.

It should also be noted that Lean/TPS includes other forms of 100% inspections as well, such as go/no-go gauge checking on every part, or successive inspection, in which each operator on an assembly line starts by touching every component that was supposed to be installed at the previous station.

Nobody likes inspections, but, if they do occasionally find defects, they are necessary. An inspection step that fails to uncover a single defect in a million consecutive units is probably unnecessary.

When you find that 100% inspection is unnecessary, you don’t replace it with sample inspection, but it would disrupt the production routine. Instead, you go to first and last piece checking on a production run. And the next step is complete elimination of the inspection step.

Reply

William Ryan on August 22, 2013 at 10:10 am

Comment in the *TPS Principles and Practice* discussion group on LinkedIn:

In engine final assembly we had top, bottom ,right and left side inspectors that also performed other tasks. This was value added for us. Nothing is allowed to get past done wrong, not even the way a rubberband is put on. We eliminated black lighting inspections for leaks and we eliminated all the hot testing of every engine that was done for years.
I performed all kinds of inspections and poke yokes over the years and it is always about cutting waste and cost and being able to add customer value at the same time. Today inspections are still used as back ups to process breakdowns where control plans require them to be done manually on every part until the process is fixed. the old quality control station at the end of every line went bye bye.
Blinking red lights means the line is on hold by the operator but solid red lights can mean bigger problems. Three in a row red means the line will stop automatically for further investigation. !00% manual inspection for most processes is obsolete but identifying, tagging, coding and grouping become more important for proper containment.

Reply

Christian Bardoux on August 22, 2013 at 10:15 am

Comment in the *TPS Principles and Practice* discussion group on LinkedIn:

Do you want zero defects with no compromise, or do you have a reduction target ?
Also, what is the severity of that defect, if delivered to a customer (next operations or final user) ?
My philosophy on that is that I have a Poka Yoke when the design makes it impossible to have a defect.
Therefore, inspection is not a Poka Yoke (and maybe I am wrong).
When it is not possible I use the FMEA approach to evaluate the issue and the solution.

↳ Reply

Michel Baudin on August 22, 2013 at 10:16 am

A Poka-Yoke must make a defect impossible for a human to generate or to pass on if generated by a machine, in a way that doesn't add any labor. What we usually call "inspections" don't qualify on multiple counts. First, they never catch all the defects; there are always a few that escape. Second, they add labor.

Not all but most Poka-Yokes are simple enough that it is cheaper to implement them than to conduct an FMEA to figure out whether you should. In this kind of situation, you don't go by "it it ain't broke, don't fix it," but by "when in doubt, fix it."

↳ Reply

John Sprovieri on October 31, 2013 at 2:45 pm

Ideally, poka-yoke techniques ensure that the right conditions exist to make a good assembly, before a joining process is actually executed. Thus, there should be only one way two parts can be joined before they are snapped, welded, bonded or fastened together. Where this is impossible, poka-yoke techniques detect defects as soon as they are made, preventing faulty assemblies from being passed to the next station. http://bit.ly/1aJFur3

To err is human, to prevent errors is the manufacturing engineer's job: http://bit.ly/16QGibh

↳ Reply

Pingback: Jidoka isn't just about "stop and fix" | Michel Baudin's Blog

Pingback: Poka-Yoke » The Lean Presentation

Search

Lean Leadership Summit 2016

About the author

Michel Baudin

Fascinated with the art of making things, Michel is working to improve it. Trained in engineering and applied math, he got his feet wet in production in the early 1980s, and later apprenticed under master Japanese consultant Kei Abe for eight years, starting his own group in 1996. He has been consulting since 1987, teaching courses and writing technical books. He intends to keep working with like-minded partners in the Takt Times Group and contributing improvements in the management and technology of manufacturing as a consultant, trainer, and writer.

Personal Links

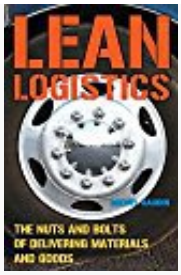
The takt times group
Blog about manufacturing
Press and blog clippings about Lean

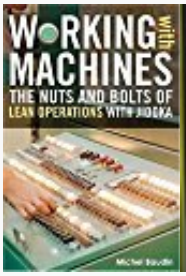
Verified Services

View Full Profile →

My books

Shopping Cart





My Bookstore | Shopping Cart

To find out more:

Name (required)

Email (required)

Phone number

Website

Your question (required)

Submit »

Videos

"Introduction to Lean 2013"
by Michel Baudin

Follow Blog via Email

Enter your email address to follow this blog and receive notifications of new posts by email.

Join 2,494 other subscribers

Email Address

Follow

Most popular posts and pages

What is Karakuri Kaizen?
Safety Stocks: Beware of Formulas
Why "Smart" part numbers should be replaced with keys and property lists
Deming's Point 3 of 14 - Cease dependence on inspection to achieve quality...
Why 5S fails

Recent Posts

Is SPC Obsolete? (Revisited)
Now It's Humans Assisting Robots | Sheelah Kolhatkar | The New Yorker
Why do we call it "value stream"?
How Hard Is It To Create Flow?
More Sophisticated Graphics In Today's New York Times

Categories

Announcements
Answers to reader questions
Asenta selection
Blog clippings
Blog reviews
Book reviews
Data science
Deming
Events
History
Information Technology
Laws of nature
Management
Metrics
News
Organization structure
Personal communications
Policies
Polls
Press clippings
Technology
Tools
Uncategorized
Web scrapings

Follow me on Twitter

My Tweets

My tags

5S Automation Autonomation Cellular manufacturing Continuous improvement Deming ERP Ford Government Health care industrial engineering Industry 4.0 Information technology IT jidoka Kaizen Kanban Lean Lean assembly Lean Health Care Lean implementation Lean Logistics Lean management Lean manufacturing Logistics Management Manufacturing Manufacturing engineering Metrics Mistake-Proofing Poka-Yoke Quality Six Sigma SMED Standard Work Strategy Supply Chain Management Takt Takt time Toyota Toyota Production System TPS Training VSM WCM

Michel Baudin's feed

RSS - Posts

RSS - Comments

View Full Site

Proudly powered by WordPress

Follow

Follow Michel Baudin's Blog

Get every new post delivered to your Inbox

Join other followers:

Email Address

Sign me up!