43

Total Product Development and Taguchi's Quality Engineering

43.1. Introduction

Quality in product development began with attempts to inspect quality into products or services in the process domain, the design domain, or the customer domain. The evolution of quality involves a significant change in thinking from reacting to inspection events to utilizing process patterns in engineering and manufacturing to build quality into the product. The use of pattern- and structure-level tools, such as the Taguchi system of quality engineering, TRIZ, and axiomatic design, provide a foundation for world-class product development.

43.2. The Challenge

Although the challenge is worldwide and exists in every industry, perhaps nowhere is the challenge so readily and clearly visible as in the U.S. automotive industry. The oil embargo of 1973 was a shocking introduction to the energy crisis, with consumers experiencing scarce gasoline and fuel oil and correspondingly high prices. In 1980, with a hostage crisis in Iran and soaring domestic gas prices, U.S. purchasers suddenly shifted to small cars and trucks, catching U.S. automakers

Taguchi's Quality Engineering Handbook. Genichi Taguchi, Subir Chowdhury and Yuin Wu

with bulging inventories of full-sized carryover models. These automobile manufacturers found themselves in a full-blown economic recession, with consumers purchasing significant numbers of smaller, more-fuel-efficient, imported automobiles. Consumers discovered that these imported vehicles also had superior quality, and the challenge began.

From 1975 to 1979, total auto sales in the United States were 51.8 million, with U.S.-made vehicles accounting for 42.3 million. From 1985 to 1989, 53.3 million vehicles were sold in the United States. Compared to 1975–1979, U.S.-made vehicles dropped 10%, to 38.1 million, of which 3.3 million were from U.S. plants built by importers, while imported vehicles rose 60%, to 15.2 million [1]. In 1984, Louis Ross of Ford Motor Company said [2]: "Imports in 1984 took 26 percent of the U.S. car market. So here in our home market domestic auto manufacturers—and their supply base—are playing in the World Series, and in order to score we have to be able to meet or beat the world's best in quality, in cost, and in productivity, and in product content."

This challenge continues today. So far, much of the work in improving quality and productivity has been in implementing lean manufacturing and assembly systems. The player's scorecard is market share [3]. Following is a summary of the U.S. automotive market share (%):

	1988	1995	2001
GM	36	33	28
Ford	25	26	23
DaimlerChrysler	15	15	14
Japanese	20	23	27
European	2	2	4
Korean	2	1	4

The challenge will be won by the companies that can build quality and productivity into the design (product and process) during the product development process.

43.3. Model of New Product Development

To understand how quality may be built into the design during product development, consider the model shown in Figure 43.1. George Box, professor emeritus at the University of Wisconsin, has observed: "All models are wrong, but some are useful." So although this model does not match reality completely, it is useful.

This model was developed based on the ideas of Nam Suh [4] and Peter Senge [5], both of MIT. Nam Suh created the domain model of product development, with the belief that great products or services are a result of proper mappings between various domains. The first domain involves understanding the customer and defining desired customer attributes. These customer attributes are then mapped from the customer domain to functional requirements in the functional

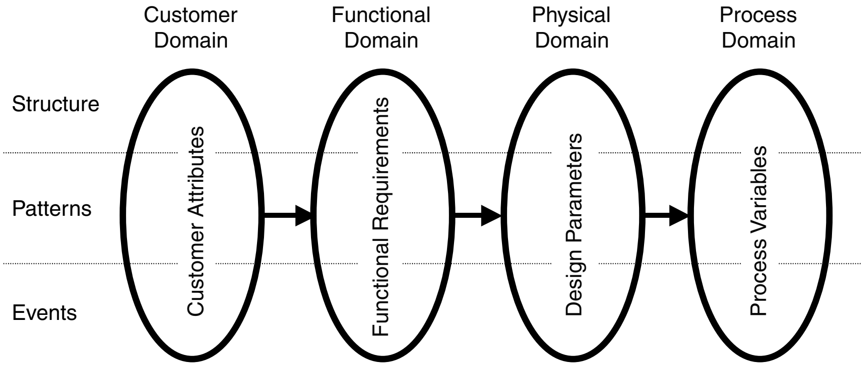


Figure 43.1
Senge's levels of thinking overlaid on Suh's domain model of product development

domain. Functional requirements are then mapped to design parameters in the physical domain. Finally, design parameters are mapped to process variables in the process domain. The better or more complete the mappings between domains, the better the resulting product or service.

Peter Senge defined different levels of thinking based on events, patterns, and structure. *Event thinking* is at the lowest level. Here, one simply waits for something to happen, then reacts to the result. *Pattern thinking* involves monitoring process trends, then managing these trends before an unpleasant event occurs. *Structural thinking* deals with establishing the fundamental structure of the system so that the process patterns can work more effectively. Figure 43.1, an overlay of Peter Senge's patterns of thinking onto Nam Suh's domain model of product development, is useful in understanding the history of quality in new product development.

43.4. Event Thinking in New Product Development

After World War II, the primary way of assuring quality to customers was inspection after the process domain. Parts were produced, and then these parts were checked to see whether they were good enough to ship. If a part was not good, an *event* had occurred, resulting in rework or scrap and problem solving. Event thinking also occurred in the physical domain. Many engineers simply threw a design together and then tested it, expecting the design to fail. The failure of a design verification test is an event that the engineers answered with a sequence of build–test–fix cycles.

Build–test–fix is actually a method used today by many designers to "inspect" quality into the product or service. Supposedly, the design will be "Band-Aided" enough so that it will function properly before the product or service reaches the customer. Otherwise, the inevitable result is consumer complaints and warranty cost in the customer domain.

Unfortunately, many companies today still depend on event thinking to assure quality to customers. These companies learn about customers through analysis of warranty cost, try to assure design integrity via batteries of expensive reliability tests, and rely on checks after assembly to assure that the product is good enough to ship. At a company like GE, the cost of quality associated with event thinking in 1996 was estimated to be as high as $10 billion each year in scrap, reworking of parts, correction of transactional errors, inefficiencies, and lost productivity [6]. Problem-solving opportunities associated with event thinking in product development are illustrated in Figure 43.2.

43.5. Pattern Thinking: Big Ideas Make Big Differences

Serious pattern-level thinking was introduced to product development when W. Edwards Deming, Joseph M. Juran, and others were invited to Japan shortly after World War II. In 1950, Ichiro Ishikawa, president of the Union of Japanese Scientists and Engineers, arranged for Deming to meet with 21 top management executives of Japanese industry and lecture about quality.

Deming began by introducing some ideas he had learned from Walter Shewhart, specifically the plan–do–study–act cycle of learning and statistical process control (SPC). SPC, a pattern-level quality method in the process domain, focuses on patterns or trends in process data so that the process can be adjusted before an inspection event occurs. When Japanese companies began to implement SPC, quality improved dramatically. For the first time, product or design engineers knew that the parts they had designed were being manufactured according to print.

Use of pattern-level thinking in the other design domains began when Kaoru Ishikawa, known for Ishikawa diagrams and formalization of quality circles, noticed that even though parts were being made to print, customers were still unhappy with the products [7]. Specifications and tolerance limits were stated in the drawings. Measurements and chemical analyses were being performed. Standards existed for all these things and the standards were being met, but these standards were created without regard to what the customer wanted.

Ishikawa wrote: "When I ask the designer what is a good car, what is a good refrigerator, and what is a good synthetic fiber, most of them cannot answer. It is obvious that they cannot produce good products." You simply cannot design a good product or service if you do not know what "good" means to a customer. Ishikawa encouraged people to think at the pattern level in the customer domain

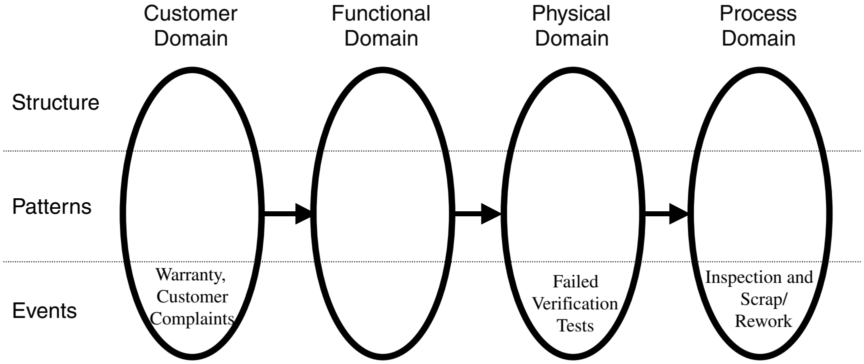


Figure 43.2 Problem-solving opportunities associated with event thinking in the various design domains

instead of just reacting to a warranty event. He said that if you don't know what a good product is, ask your customers. Customers will give you what Ishikawa called the *true quality characteristics.*

The problem with true quality characteristics, or customer attributes (CAs), is that the designer often cannot use them directly. For example, a customer may want the steering of an automobile to be "comfortable." An engineer cannot write on a drawing, "Make the steering comfortable." The engineer must find substitute quality characteristics, dimensions, or characteristics of the design that are correlated with customer desires but have meaning to an engineer [engineering characteristics (ECs)].

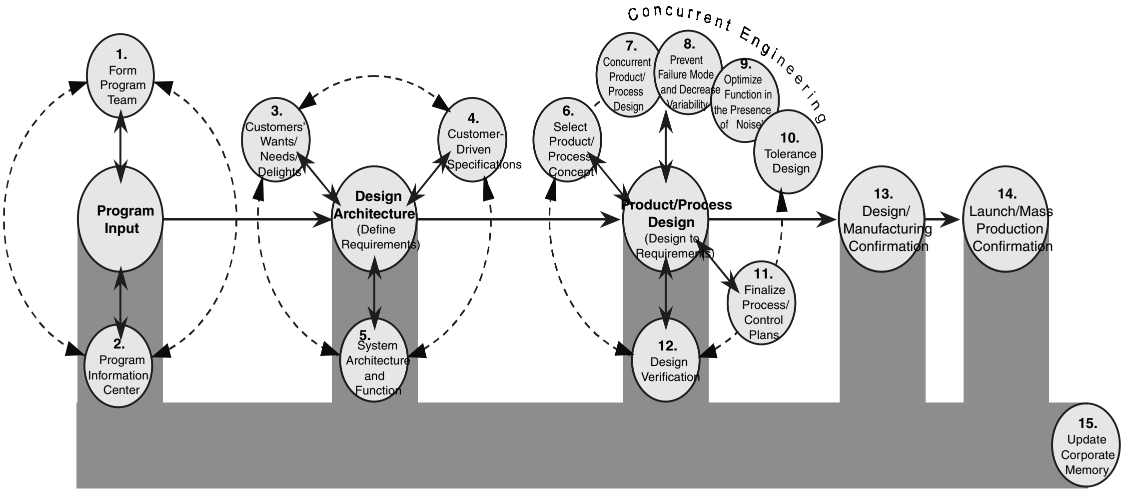
Therefore, Ishikawa said that the designer must create a map that moves from the "world of the customer" to the "world of the designer." He used a tree diagram to create such a map and called these maps *quality tables.* The Kobe Shipyard of Mitsubishi Heavy Industries created the first quality table in 1972. Once the quality table was completed, Ishikawa felt that the designer had a customer-driven definition of a good product or service. This definition of quality could then be deployed into the product development activity. Thus, quality function deployment (QFD) was born [8].

In subsequent years, about 120 different quality tools and methods were created at the pattern level for designers to manage product development process trends, so that inspection "events" would become more of a nonevent. Some of the most popular and powerful of these tools include:

- ❑ Failure mode and effects analysis (FMEA) for both the product and process domains
- ❑ Taguchi's quality engineering, including the methods of parameter design (for the product and process domains) and tolerance design (for the product domain)
- ❑ Design for assembly (DFA) and design for manufacturing (DFM), which improve the mapping from the product to the process domain
- ❑ System engineering, value analysis (VA), and value engineering (VE) in the functional domain

Rather than mentioning over 120 tools, it is easier and more effective to consider the fundamental ideas behind the tools and emphasize these ideas. For example, the main idea behind the tool of FMEA for both design and process is that after the design of the product (process) has been roughed out, it makes sense to ask "What can go wrong?" and "How can we modify the design and the process to prevent that from happening?" It is the implementation of this big idea behind the tool that makes big differences in results. Six tools have been developed to access this big idea in various situations. Besides FMEA, the best known include fault tree analysis and process decision program charts.

All together, there are 15 big ideas that make big differences at the pattern level of thinking. These ideas are illustrated in Figures 43.3 to 43.7, and a few are shown in the domain model of Figure 43.8. Among these 15 ideas, concept selection is the work of Stuart Pugh [9], and Taguchi contributed the ideas on parameter design and tolerance design.



Note: Dashed line denotes iteration.

Figure 43.3
Pattern-level big ideas that make big differences

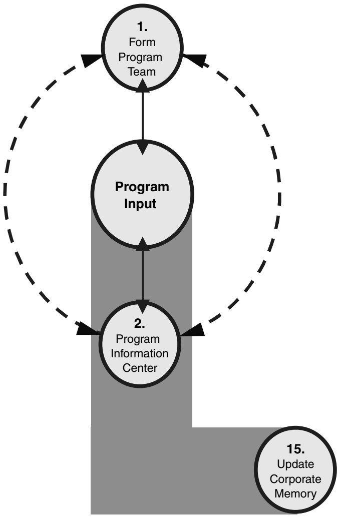
Purpose

Step 1. People Work as a Program Team, Share a Vision

Establish and maintain a program development team (for both product and process) that has a shared vision.

Step 2. Create a Program Information Center

Establish and maintain a program information center and understand program, social, and institutional lessons learned.



Breakthrough Idea

Step 1. High-performance teams emerge and grow through systematic application of learning organization and team disciplines to foster synergy, shared direction, interrelationships, and a balance between intrinsic and extrinsic motivation factors.

Step 2. Create an information environment with networks to foster program/product knowledge via: prior lessons learned (corporate memory), best practices (institutional, technical, and social), and inter- and intradisciplinary communication and collaboration.

Figure 43.4
Pattern-level big ideas associated with program input

Purpose

Step 3. Establish/Prioritize Customers' Wants/Needs/Delights

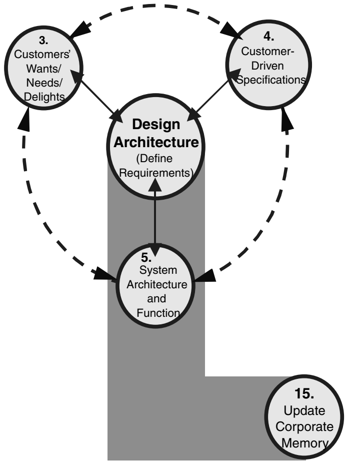
Identify customers and create opportunities for team members to establish/prioritize customer wants, needs, delights, usage profiles, and demographics.

Step 4. Derive Customer-Driven Specifications

Translate customer, corporate, and regulatory requirements into product/process specifications and engineering/test plans.

Step 5. Define System Architecture and Function

Define system architecture, inputs/outputs, and ideal function for each of the system elements, and identify interfaces.



Breakthrough Idea

Step 3. Foster intense customer engagement to identify base expectations as well as distinctive opportunities which differentiate and characterize a winning product.

Step 4. Establish the foundation (maximum potential) for customer satisfaction by systematically translating the customer definition of a "good" product into engineering language and competitive targets.

Step 5. Lay the foundation for analytical optimization of function, cost, quality, and performance by gaining understanding of how the system and system elements function ideally, and by gaining understanding of the interfaces and interactions between functional system elements.

Figure 43.5
Pattern-level big ideas associated with design architecture

Purpose

Step 6. Select Product/Process Concept

Create/establish alternative product design and manufacturing process concepts and derive enhanced alternatives for development.

Step 7. Concurrent Product and Process Design

Design and model product and process concurrently using low-cost tolerances and inexpensive materials.

Step 8. Prevent Failure Modes and Decrease Variability

Improve product and process through reduction of failure modes and variability.

Step 9. Optimize Function in the Presence of Noise

Optimize product and manufacturing/assembly process functions by testing in the presence of anticipated sources of variation, or noise.

Step 10. Tolerance Design

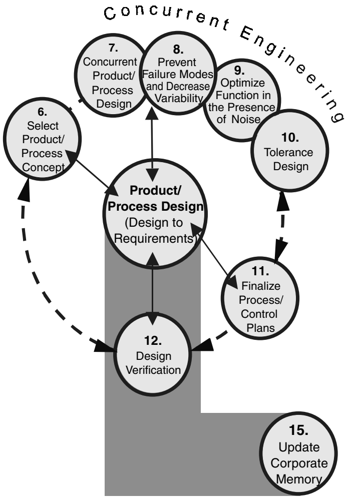
Selectively tighten tolerances and upgrade materials to achieve desired performance (with cost/benefit trade-offs). Identify key characteristics for manufacturing control and variability reduction.

Step 11. Finalize Process/Control Plans

Finalize process and establish tooling, gauges, and control plans.

Step 12. Design Verification

Integrate and verify design and manufacturing process functions with production-like hardware/software.



Breakthrough Idea

Step 6. Derive a concept to meet or exceed customer expectations through systematic exploration of many alternatives.

Step 7. Achieve superior performance through simultaneous integration of engineering, manufacturing, and delivery functions.

Step 8. Improve product and process by asking: "What can go wrong?" and "Where can variation come from?" Revise design and process to prevent occurrence and reduce variation.

Step 9. Improve performance against customer targets, up front in the development process, by adjusting controllable parameters to minimize deviations from the intended/ideal function.

Step 10. Achieve functional targets at lowest cost by selectively tightening tolerances and upgrading materials only where necessary. Demonstrated customer-sensitive characteristics are chosen for ongoing variation reduction.

Step 11. The manufacturing and assembly processes, tooling, gauges, and control plans are appropriately designed to control and reduce variation in characteristics that influence customer satisfaction.

Step 12. Improve quality and reduce time-to-market by enabling a single prototype build/test/fix cycle.

Figure 43.6
Pattern-level big ideas associated with product/process design

Purpose

Step 13. Design/Manufacturing Confirmation

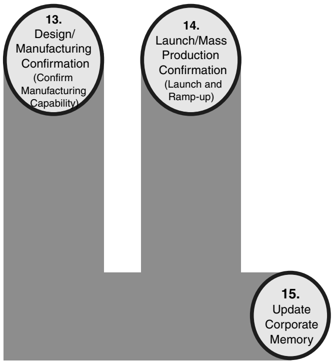
Confirm manufacturing and assembly process capability to achieve design intent.

Step 14. Launch/Mass Production Confirmation

Launch the product, ramp-up, and confirm that mass production delivers function, cost, quality, and performance objectives.

Step 15. Update Corporate Memory

Update corporate knowledge database with technical, institutional, and social lessons learned.



Breakthrough Idea

Step 13. Enable rapid, smooth confirmation of production and assembly operations with minimal refinements.

Step 14. Implementation of robust processes and up-front launch planning, with just-in-time training, will promote a smooth launch and rapid ramp-up to production speed. Final confirmation of deliverables enables team learning with improved understanding of cause/effect relationships.

Step 15. Retain whats been learned. Create and maintain capability for locating, collecting, and synthesizing data and information into profound knowledge. Communicate in a timely, user-friendly manner all technical, institutional, and social lessons learned.

Figure 43.7
Pattern-level big ideas associated with prelaunch and launch

43.6. Taguchi's System of Quality Engineering

One of the most significant achievements associated with designing quality into a product is Taguchi's system of quality engineering [10]. As shown in Figure 43.8, this system spans the physical and process domains and consists of:

☐ Parameter design in the physical domain
☐ Tolerance design in the physical domain
☐ Process parameter design in the process domain
☐ On-line quality control in the process domain

Use of these methods can make improvements in quality by factors of 10 and also improve both cost and productivity. Taguchi's intent is to produce the highest-quality product at the lowest possible cost. The big idea behind parameter design is the concept of hitting the initial product (and/or process) design with "noise"

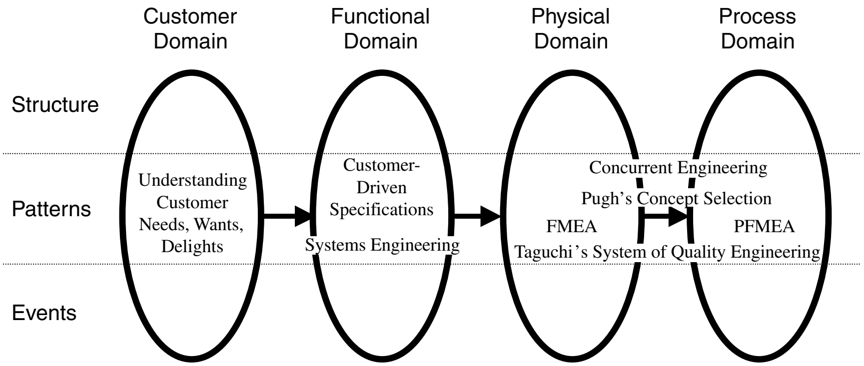


Figure 43.8
Examples of pattern thinking for problem prevention in various design domains

and finding a combination of factors that can be adjusted in the design systematically to make its functional performance insensitive to noise. Here noise is defined as variation the engineer either cannot control or chooses not to control, but which may affect product (and/or process) performance.

For example, environmental conditions are a noise. An automotive engineer cannot control whether a vehicle will be required to start in the freezing arctic cold or in a very hot desert, or whether the humidity will be very dry or very moist, but the vehicle must start and perform in all conditions. User variation is a noise. The driver may be conservative or extremely aggressive, yet the vehicle must function as intended. Less extreme variation in environment or operators may also affect process performance. Variation in material and/or part characteristics is also noise. So is functional deterioration over time.

Prior to the creation of parameter design, the best an engineer could do was to understand what is important to the customer in terms of product and process characteristics, find the target or set point, and work to reduce variation around this target point. With parameter design, one can make the variation in a characteristic insensitive from the customer point of view and open up tolerances to achieve high quality at low cost. This idea is a significant improvement to the product development process—so much so that if your competition is doing this and you are not, you will not be able to compete.

Similarly, tolerance design is a big idea that can improve quality and reduce cost. Too often, tolerances are chosen because "We have always done it that way" or "This is the standard tolerance for this situation." Taguchi's huge contribution involves using a designed experiment (orthogonal arrays) to understand the cause-and-effect relationships between tolerances and/or material choices and functional performance. With this knowledge, large tolerances and low-cost materials may be used for characteristics whose variation does not affect customers, achieving low cost. When customers are sensitive to a characteristic's variation, the characteristic can then be processed to reduce this variation with a tolerance that meets functional targets, thus achieving high quality. Of course, this exercise should be done after parameter design, because the number of customer-sensitive characteristics will then be reduced.

On-line quality control involves checking and managing manufacturing and assembly process trends in the most efficient and practical way possible so that subsequent process inspection events will not occur. One must not underestimate the impact that Taguchi's ideas can have on product and process performance. Nevertheless, as powerful as these ideas are, they will not matter if the engineer does not understand or design for the customer, if the fundamental physics of the design is inappropriate, or if equipment is not maintained properly. Structure thinking makes a difference.

43.7. Structure Thinking in New Product Development

Thinking at a level of fundamental structure offers even higher-leveraged opportunities to create products and services that not only function as intended, but also deliver unprecedented customer satisfaction. When the foundational structure of design is properly established, the methods at the pattern level are much more effective. When pattern-level methods work well, the event outcomes become world-class.

In the evolution of quality, two very powerful design methods have emerged at the structural level: axiomatic design and TRIZ (see Figure 43.9).

Axiomatic design is the result of work by Nam Suh [11]. In the late 1970s, he asked himself the question: Can the field of design be scientific? Suh wanted to establish a theoretical foundation for design by identifying the underlying principles or axioms that every great design must have in common. He knew that he could never prove that these principles were true, but could he find a set of principles for which no one could find a counterexample? After a decade of work, two principles emerged. From these two principles, theorems and corollaries could be derived that together form a scientific basis for the structure of great design.

The first principle that Suh discovered was the principle of independence. Consider Suh's domain model shown in Figure 43.1. Mapping between domains represents a mapping of "what's" to "how's." The *independence axiom* states that in great designs, the "how's" are chosen such that the "what's" maintain independence; that is, design parameters must be chosen in such a way that functional requirements maintain independence. For example, consider the water faucet designs shown in Figure 43.10.

The functional requirements for a water faucet are to control the flow rate and the water temperature. As functional requirements, these are independent by definition. A design that obeys the independence axiom will maintain this independence.

The faucet on the left of Figure 43.10 has two design parameters: a hot water knob and a cold water knob. What is the relationship between these design parameters and the functional requirements? When the hot water knob is turned, temperature is affected, but so is flow. Turning of the cold water knob also affects temperature and flow. Therefore, this design is coupled and the independence of the functional requirements has not been maintained. If a consumer has optimized flow rate, then turns one of the knobs to optimize temperature, the flow rate is changed and is no longer optimal. Designs of this type can eventually satisfy customers only by iterating between the two design parameters.

Consider the design on the right of Figure 43.10. This faucet has one handle and the design parameters are to lift the handle to adjust flow and to move the

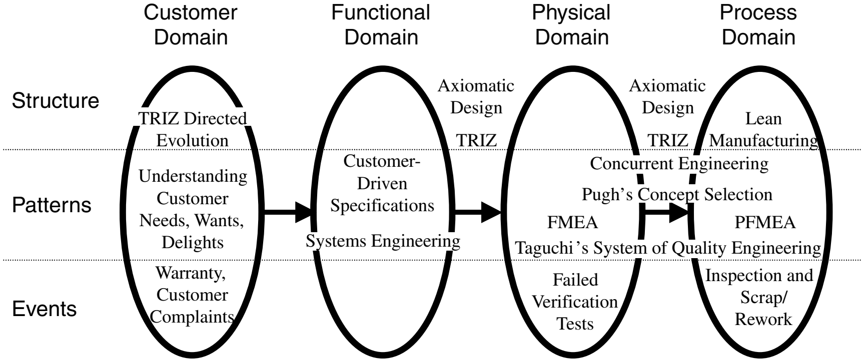
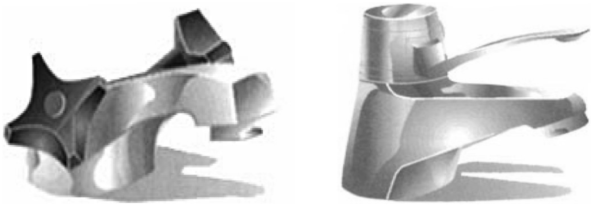


Figure 43.9
Quality evolution in the various design domains



Functions for water faucet:
FR_1 = control flow rate FR_2 = control temperature

Design1:
DP_1 = hot water knob
DP_2 = cold water knob

Design 2:
DP_1 = handle lifting
DP_2 = handle moving side to side

Figure 43.10
Water faucet example

handle from side to side to adjust temperature. In this design, adjusting temperature does not affect flow, and adjusting flow does not affect temperature. From an axiomatic design point of view, this design is superior because it maintains the independence of the functional requirements.

Imagine what happens when a designer is working in a situation with a dozen or more functional requirements. If the design is coupled, optimization of one function may adversely affect several other functions. When these functions are fixed, the original function no longer works well. The designer is always tuning and "Band-Aiding" such a design and the customer will never be completely happy. However, if the design is created in such a way that each of the functional requirements is handled independently by the design parameters, each function of the design can easily be optimized with pattern-level tools. Taguchi's additive model allows us to reach the same conclusion. Functional independence (Suh) or the absence of control factor interactions (Taguchi's additive model) provides the opportunity to design and manufacture products that will attenuate all forms of unwanted variation, including variation due to conditions of customer use.

The independence axiom can be used to evaluate how good a design will be when the design is still on paper! But suppose that you have two design alternatives that both follow the independence axiom. Which one is better? Suh's second principle states that the better design will minimize the information content necessary for implementation; that is, the better design has a higher probability of meeting the required functional performance [4].

This second principle is related to Taguchi's application of the loss function. When a manufacturing process is composed of parameters with broad, shallow loss functions implying wide tolerances, the process will function with minimum effort or controls and required information content is low. If the process has many parameters with steep, narrow loss functions, implying very tight tolerances, it must be operated with many rules or constraints and must be tuned constantly. Required information content is high.

When a product design, like a copy machine, consists of parameters with broad, shallow loss functions, the design will function with minimum effort or controls over a wide range of conditions (paper type, environmental temperature, etc.) so the information content required is low. On the other hand, if the copy machine has many parameters with steep, narrow loss functions, it must be operated with many rules or constraints, has a very high downtime, and must be tuned continually by a repairman. Required information content is high. Designs that have a solid axiomatic foundation simply work better than designs that do not.

Suppose the designer cannot find a set of design parameters that maintain the independence of all of the functional requirements. In this situation, improving one function typically degrades another. An example in automotive steering is steering "road feel" and "parking efforts." When the steering efforts are high, the customer experiences good "road feel." However, high efforts can make it difficult for customers to park. Adjusting efforts to make the vehicle "easy to park" will result in degraded "road feel." A typical approach to resolve this situation is compromise—trade off customer functionality and hope for the best.

Another example of compensation was in the early introduction of front-wheel-drive vehicles. These vehicles tended to steer to the right as the accelerator was compressed. The driver had to compensate by applying a negative steering wheel angle. This approach is clearly undesirable here and must also be avoided in the design stage if we wish performance to stay on target.

This is where TRIZ is most helpful. TRIZ, a Russian acronym for *theory of inventive problem solving*, is the result of over 45 years of research by Genrich Altshuller and colleagues [12]. Altshuller hated compromise. He called the situation where functions oppose each other contradictions and developed a methodology where design teams could systematically innovate and find design parameters that resolved contradictions, thereby creating "win–win" functional situations.

The methodology began by identifying all possible contradictions that existed in patent databases and identifying how these contradictions were resolved. Altshuller found that only a few particular principles of resolution have ever been widely used to resolve certain pairs of functional contradictions. For example, suppose that the functions of weight and reliability contradict. When the design is changed to improve reliability, weight increases. When weight is decreased, reliability degrades. Altshuller found that for the most part, there are only four principles that have been commonly used to resolve this contradiction [12]. He created a matrix of contradictions and resolution principles and used this information to guide design teams so that they could brainstorm in areas that are likely to lead to "win–win" solutions. Altshuller also believed in minimizing information content; this he called the *principle of ideality*. Later, TRIZ was expanded to include an entire database of innovation techniques, including the study of system evolution.

Altshuller found that systems tend to evolve along specific laws and lines of evolution. By studying system evolution for the past and present (for the supersystem, system, and subsystem), a designer can identify the current stage of system evolution. By applying laws and lines of evolution, design teams can predict what the next developments of the system will be. This is a huge competitive advantage. Companies that operate at the event level obtain information about the customer through warranty data. Companies at the pattern level interact with customers and find out what customers believe is important today. No customer can tell the designer what will be important tomorrow. At the structural level, TRIZ-directed evolution predicts what will be important to customers tomorrow!

So if TRIZ works at the structural foundation of design, why does the name imply problem solving? The answer is simply that higher-level thinking can always be used as a methodology to solve lower-level problems. The fact that problems exist in the event realm means that the original design process had serious flaws—pattern or structural work that should have been done is either missing or incomplete. The designer can always go back and complete this work at any time. This is why completing work associated with pattern or structural tools can quickly lead to problem resolution. A good example of this is six sigma, which addresses problems created by event-level thinking, but the methodology of six sigma utilizes pattern-level tools such as FMEA.

43.8. Synergistic Product Development

The world has yet to see what can happen when all the big ideas associated with structure, pattern, and event thinking are implemented iteratively and synergistically. As product development technology moves in this direction, consumers can look forward to truly robust, reliable products and services—the big ideas are applicable to both—with unprecedented levels of customer satisfaction.

Great designs must start with a solid foundation. This means using TRIZ-directed evolution to understand how the system is evolving. In addition, principles of axiomatic design and TRIZ must be used to assure that an adequate architectural structure is established so that design parameters are chosen such that functional requirements maintain independence.

With this foundation, the tools at the pattern level of thinking will work so much better. With a proper understanding of system evolution, the results of consumer research can be interpreted correctly so that the planned product or service is something that consumers will want. Using structural principles of design and TRIZ, system engineering can be deployed effectively by Suh's method of zigzagging between the functional and physical domains, creating tree diagrams of functions and related design parameters. Now the stage is set to use quality engineering.

How does this up-front structural work make it easier to utilize Taguchi's system? First, one of the things engineers struggle with in parameter design is the question of what to measure. An effective measure is one that is related to the underlying physics and energy of the system. Doing the up-front structural work will make engineers better understand the fundamental physics of the system, which is extremely useful in selecting effective performance measures for optimization.

Second, if the foundation of the design is poor, it is impossible to optimize all the functions of the system effectively. However, if the design parameters are chosen such that the functional requirements maintain independence, different control factors may be used to optimize different functions, so that parameter design optimization can be effective and efficient. Efforts to get the structural foundation of the design right must first be done up-front.

How does quality engineering enable activities at the event level of thinking to go well? Imagine a process that is not robust, that is, a process sensitive to noise. Such a process rarely runs well for long intervals. Parts are OK, OK, OK, and then suddenly a part is out of specification. Something changed, a noise hit the process, and now tuning is needed to bring the parts back to specification. Operators scramble with adjustments until everything is running well again. Then, suddenly, the parts are out of print in the other direction of the specification. Such a process never works well and requires a great deal of management attention and energy. A robust process does not care about changes in noise, and parts track on target.

Imagine a product that is robust—a product that always performs close to target, no matter the circumstances. Such products will cause customers to brag about this performance to their neighbors. Deming used to say that the best thing we could do is to make products (or services) that cause customers to brag to their friends and neighbors. When it comes time to purchase another product, not only do they come back to buy yours, but they bring friends and neighbors with them. The industry challenge will finally be won by those companies that integrate and synergistically implement the big ideas that make big differences into the way they design products and services. People will go out of their way to find such products and services.

43.9. Conclusions

Quality tools and methods have evolved utilizing three stages of thinking (event, pattern, and structure) across four domains associated with product development. Taguchi's system of quality engineering plays a central role at the pattern or process level of thinking. Implementation of axiomatic design and TRIZ at the structure level will enhance the effectiveness of pattern-level processes. When the pattern-level processes work to their capability, the customer-related events are world-class. Companies that wish to accelerate development of their own quality programs can utilize the concepts (big ideas) explained in this chapter to understand their current level of evolution and to implement focused actions that can move them quickly past their competition.

ACKNOWLEDGMENT

The authors would like to thank Dr. Carol Vale of the Ford Motor Company for her assistance in editing this document and making the content more readable.

References

1. *Ward's Automotive Yearbook, 1979, 1985, 1990.* Southfield, MI: Ward's Communications.
2. *Ward's Automotive Yearbook, 1985*. Southfield, MI: Ward's Communications.
3. *Time*, January 14, 2002, p. 43.
4. Nam P. Suh, 2001. *Axiomatic Design: Advances and Applications.* New York: Oxford University Press.
5. Peter M. Senge, 1990. *The Fifth Discipline.* New York: Doubleday.
6. M. Harry and R. Schroeder, 2000. *Six Sigma.* New York: Doubleday.
7. K. Ishikawa, 1988. Quality Analysis. *ASQC Conference Transactions—Philadelphia*, pp. 423–429.
8. Dan Clausing and John R. Hauser, 1998. *House of Quality*. Cambridge: Harvard Business School Press.
9. Stuart Pugh, 1991. *Total Design*. Reading, MA: Addison-Wesley.
10. Genichi Taguchi, 1993. *Taguchi on Robust Technology Development: Bringing Quality Engineering Upstream.* New York: ASME Press.
11. Nam P. Suh. 1990. *The Principles of Design.* New York: Oxford University Press.
12. G. Altshuller. 1998. *40 Principles: TRIZ Keys to Technical Innovation.* Worchester, MA: Technical Innovation Center.

This chapter is contributed by Larry R. Smith and Kenneth M. Ragsdell.